

OpenEye
SCIENTIFIC

Grapheme TK – C#

Release 2026.1.0

Cadence Design Systems, Inc.

August 06, 2026

CONTENTS

1 Introduction **1**

2 Theory **3**
- 2.1 Annotating Atoms and Bonds . . . 3
- 2.2 Drawing a Molecule Surface . . . 9
- 2.3 Drawing a Ligand-Protein Complex Surface . . . 20
- 2.4 Depicting Property Maps . . . 27

3 Examples **33**
- 3.1 Grapheme Examples . . . 33
- 3.2 Python Cookbook Examples . . . 57

4 API **59**
- 4.1 OEGrapheme Classes . . . 59
- 4.2 OEGrapheme Basic Arc Drawing Classes . . . 143
- 4.3 OEGrapheme Constants . . . 177
- 4.4 OEGrapheme Functions . . . 207
- 4.5 OEGrapheme Basic Circle and Arc Drawing Functions . . . 258

5 Preliminary API **291**
- 5.1 Preliminary OEGrapheme Classes . . . 291
- 5.2 Preliminary OEGrapheme Functions . . . 300

6 Release History **307**
- 6.1 Grapheme™ TK 1.5.5 . . . 307
- 6.2 Grapheme™ TK 1.5.4 . . . 308
- 6.3 Grapheme™ TK 1.5.3 . . . 308
- 6.4 Grapheme™ TK 1.5.2 . . . 308
- 6.5 Grapheme™ TK 1.5.1.3 . . . 308
- 6.6 Grapheme™ TK 1.5.1.0 . . . 308
- 6.7 Grapheme™ TK 1.5.0 . . . 308
- 6.8 Grapheme™ TK 1.4.7 . . . 308
- 6.9 Grapheme™ TK 1.4.6 . . . 309
- 6.10 Grapheme™ TK 1.4.5 . . . 309
- 6.11 Grapheme™ TK 1.4.4 . . . 309
- 6.12 Grapheme™ TK 1.4.3 . . . 309
- 6.13 Grapheme™ TK 1.4.2 . . . 309
- 6.14 Grapheme ™ TK 1.4.1 . . . 310
- 6.15 Grapheme™ TK 1.4.0 . . . 310
- 6.16 Grapheme™ TK 1.3.7 . . . 312
- 6.17 Grapheme™ TK 1.3.6 . . . 313

6.18 Grapheme™ TK 1.3.5 . . . 314
6.19 Grapheme™ TK 1.3.4 . . . 315
6.20 Grapheme™ TK 1.3.3 . . . 316
6.21 Grapheme™ TK 1.3.2 . . . 318
6.22 Grapheme TK 1.3.1 . . . 319
6.23 Grapheme TK 1.3.0 . . . 321
6.24 Grapheme TK 1.2.3 . . . 323
6.25 Grapheme TK 1.2.2 . . . 324
6.26 Grapheme TK 1.2.1 . . . 325
6.27 Grapheme TK 1.2.0 . . . 325
6.28 Grapheme TK 1.1.3 . . . 326
6.29 Grapheme TK 1.1.2 . . . 327
6.30 Grapheme TK 1.1.1 . . . 328
6.31 Grapheme TK 1.1.0 . . . 328
6.32 Grapheme TK 1.0.6 . . . 330
6.33 Grapheme TK 1.0.5 . . . 330
6.34 Grapheme TK 1.0.4 . . . 331
6.35 Grapheme TK 1.0.3 . . . 332
6.36 Grapheme TK 1.0.2 . . . 332
6.37 Grapheme TK 1.0.1 . . . 332
6.38 Grapheme TK 1.0.0 . . . 333

7 Copyright and Trademarks **335**

8 Sample Code **337**

9 Citation **339**
9.1 Orion® . . . 339
9.2 Orion Floes . . . 339
9.3 Toolkits and Applications . . . 339
9.4 Publications for Bibliographies . . . 341
9.5 OpenEye MMDS Web Service . . . 342

10 Technology Licensing **343**
10.1 GCC . . . 415

Index **431**

CHAPTER

ONE

INTRODUCTION

2D structure diagrams can be considered to be the "natural language" of chemists, since the graphical representation allows molecules to be instantly conceivable.

Historically, 2D representations have mainly been used to visualize the connection table of molecular graphs. However, projecting 3D information into the 2D layout and displaying various atom and bond properties on the molecular diagram open up a novel way to present information to chemists.

The **Grapheme TK** provides several representation schemes that allow visualization of complex molecular interactions and properties in a clear and coherent 2D format that is the most natural to chemists.

Oxford dictionary definition

> grapheme /'grafēm/
>
> The smallest meaningful contrastive unit in a writing system.

Wiki definition

> A grapheme is the smallest semantically distinguishing unit in a written language, analogous to the phonemes of spoken languages. A grapheme may or may not carry meaning by itself, and may or may not correspond to a single phoneme.

Graphemes include alphabetic letters, typographic ligatures, Chinese characters, numerical digits, punctuation marks, and other individual symbols of any of the world's writing systems. The word grapheme is derived from Greek gráphō ("write"), and the suffix -eme, by analogy with phoneme and other names of emic units.

Note

This manual provides examples about molecule annotations, drawing 2D molecule surfaces and displaying atom properties on a property map. However, the basic concepts of molecule depiction and image handling are discussed in the **OEDepict TK** manual. Therefore, studying the **OEDepict TK** manual is highly recommended prior to reading this documentation.

CHAPTER

TWO

THEORY

2.1 Annotating Atoms and Bonds

The **Grapheme TK** provides a framework to annotate atoms and bonds *i.e.* mark them based on their properties. The following *Listing 1* example shows how to mark sp^2 and sp^3 hybridized atoms of depicted molecules. After preparing the molecule for 2D depiction and calculating the atom hybridizations, the molecule display object is created (OE2DMolDisplay) that stores the depiction information of a molecule The *OEAddGlyph* function is then called to add *glyph* to those atoms for which the given OEIsAtomHybridization functor returns *true*. In this example, the built-in *OEAtomGlyphCircle* class is used to mark atoms by drawings a circle around them with a specific style (*OECircleStyle*). Finally, the image is written out to a file by calling the `OEWriteImage` function. The image created by *Listing 1* example is shown in *Figure: Example of atom annotation*.

Listing 1: Example of using a built-in atom annotation style

```
public class AnnotateAtomPredicate
{
    public static int Main(string[] args)
    {
        OEGraphMol mol = new OEGraphMol();
        OEChem.OESmilesToMol(mol, "c1cc(N)cc(S(=O)(=O)O)c1");
        OEChem.OEAssignHybridization(mol);
        OEDepict.OEPrepareDepiction(mol);

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(350, 250, OEScale.
↪AutoScale);
        opts.SetTitleLocation(OETitleLocation.Hidden);
        OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

        OEPen penSP2 = new OEPen(OEChem.OEWhite, OEChem.OEBlueTint, OEFill.Off, 1.5);
        OEAtomGlyphCircle glyphSP2 = new OEAtomGlyphCircle(penSP2, OECircleStyle.Sun, 1.
↪2);
        OEGrapheme.OEAddGlyph(disp, glyphSP2, new OEIsAtomHybridization(OEHybridization.
↪sp2));

        OEPen penSP3 = new OEPen(OEChem.OEWhite, OEChem.OEPinkTint, OEFill.Off, 1.5);
        OEAtomGlyphCircle glyphSP3 = new OEAtomGlyphCircle(penSP3, OECircleStyle.Eyelash,
↪ 1.2);
        OEGrapheme.OEAddGlyph(disp, glyphSP3, new OEIsAtomHybridization(OEHybridization.
↪sp3));

        OEDepict.OERenderMolecule("AnnotateAtomPredicate.png", disp);
```

(continues on next page)

(continued from previous page)

```
        return 0;
    }
}
```

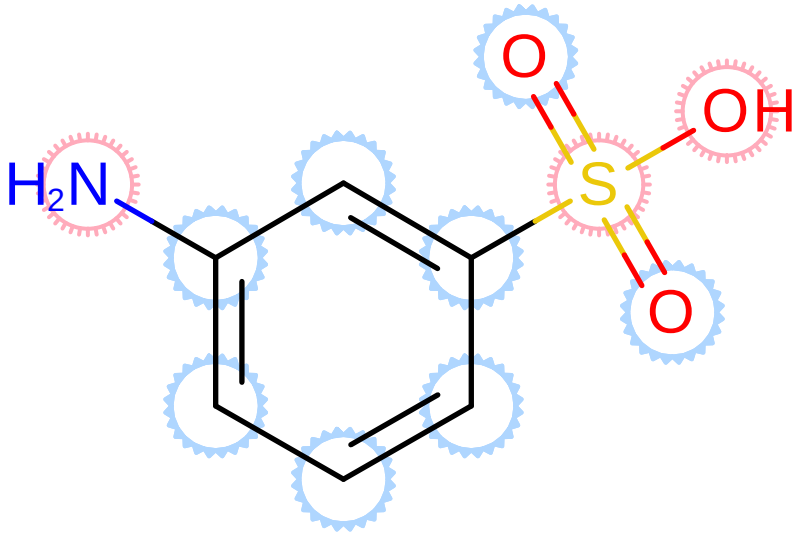


Fig. 1: **Example of atom annotation**

See also

- *OEAddGlyph* function
- *OEAtomGlyphCircle* class
- *OECircleStyle* namespace
- OE2DMolDisplay class and `OEWriteImage` function in the **OEDepict TK** manual

Similarly, the following example shows how to annotate bonds. The bonds being highlighted are specified by the bond predicate passed to the *OEAddGlyph* function. For each bond the predicate returns `true`, *i.e.* if it is a rotatable bond, the *OEBondGlyphArrow.RenderGlyph* method is invoked that draws an arrow across the middle of the specific bond. The image created by *Listing 2* is shown in *Figure: Example of bond annotation*.

Listing 2: Example of using a built-in bond annotation style

```
public class AnnotateBondPredicate
{
    public static int Main(string[] args)
    {
        OEGraphMol mol = new OEGraphMol();
        OEChem.OESmilesToMol(mol, "c1cc(NCC)cc(CS(=O)(=O)O)c1");
```

(continues on next page)

(continued from previous page)

```
        OEDepict.OEPrepareDepiction(mol);

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(400, 250, OEScale.
↪AutoScale);
        opts.SetTitleLocation(OETitleLocation.Hidden);
        OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

        OEPen pen = new OEPen(OEChem.OEDarkPurple, OEChem.OEDarkPurple, OEFill.Off, 2.0);
        OEBondGlyphArrow glyph = new OEBondGlyphArrow(pen, 0.5);
        OEGrapheme.OEAddGlyph(disp, glyph, new OEIsRotor());

        OEDepict.OERenderMolecule("AnnotateBondPredicate.png", disp);
        return 0;
    }
}
```

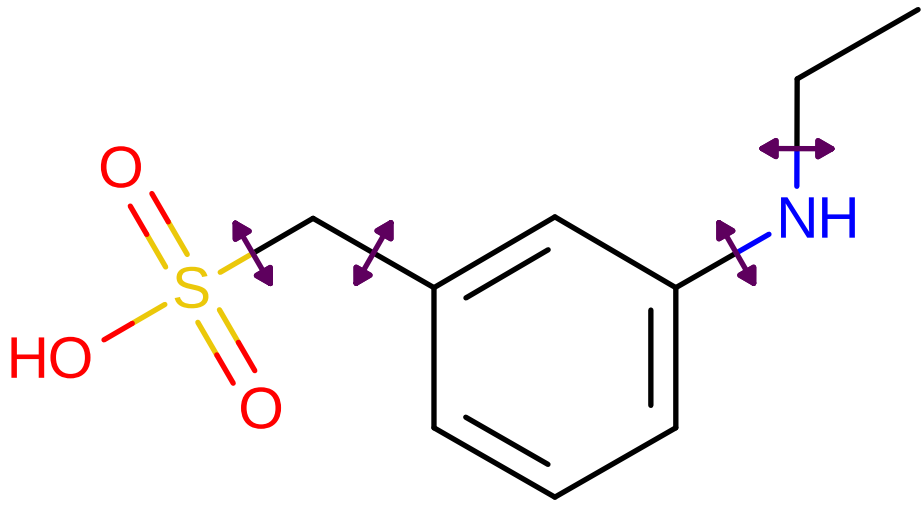
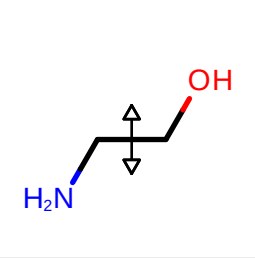
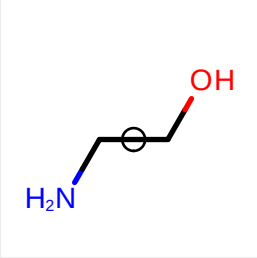
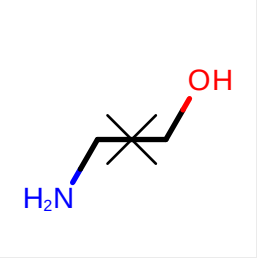
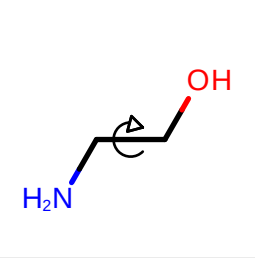
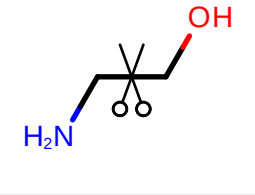
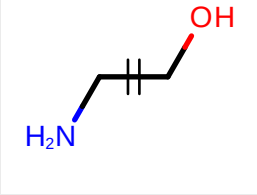
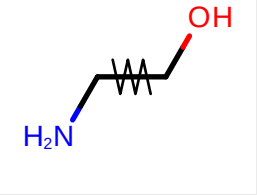


Fig. 2: **Example of bond annotation**

The following table lists the customizable bond glyphs that are currently available in **Grapheme TK**.

Table 1: **Supported bond glyphs**

			
OEBondGlyphArrow	*OEBondGlyphCircle*	*OEBondGlyphCross*	*OEBondGlyphCurvedArrow*
			
OEBondGlyphScissors	*OEBondGlyphStitch*	*OEBondGlyphZigZag*	

See also

- *OEAddGlyph* function

The last example shows how to depict *non-boolean* properties by implementing a user-defined annotation style. User-defined atom and bond annotations can be implemented by deriving from the *OEAtomGlyphBase* and the *OEBondGlyphBase* abstract classes and implementing the *OEAtomGlyphBase.RenderGlyph* and the *OEBondGlyphBase.RenderGlyph* methods, respectively.

In the *Listing 3* example, after calculating the MMFF charges, the atoms are annotated by their charge using the OELinearColorGradient class that interpolates colors. Atoms with negative and positive charge are highlighted by red and blue colors, respectively. The image created by *Listing 3* is shown in *Figure: Example of user-defined annotation*.

Listing 3: Example of user-defined annotation

```
public class AnnotatePartialCharge
{
    private class ColorCharge : OEAtomGlyphBase
    {
        private OELinearColorGradient colorg;

        private ColorCharge() {}
        public ColorCharge(OELinearColorGradient cg)
        {
            colorg = new OELinearColorGradient(cg);
        }
        public override bool RenderGlyph(OE2DMolDisplay disp, OEAtomBase atom)
        {
```

(continues on next page)

(continued from previous page)

```
            OE2DAtomDisplay adisp = disp.GetAtomDisplay(atom);
            if (adisp == null || !adisp.IsVisible())
                return false;

            double charge = atom.GetPartialCharge();
            if (charge == 0.0)
                return true;
            OEColor color = colorg.GetColorAt(charge);

            OEPen pen = new OEPen();
            pen.SetForeColor(color);
            color.SetA(100);
            pen.SetBackColor(color);
            pen.SetFill(OEFill.On);
            double radius = disp.GetScale()/2.5;

            OEImage layer = disp.GetLayer(OELayerPosition.Below);
            OEGrapheme.OEDrawCircle(layer, OECircleStyle.Simpson, adisp.GetCoords(),␣
↪radius, pen);

            return true;
        }
        public override OEAtomGlyphBase CreateCopy()
        {
            ColorCharge copy = new ColorCharge(colorg);
            copy.ReleaseOwnership();
            return copy;
        }
    }

    public static int Main(string[] args)
    {
        OEGraphMol mol = new OEGraphMol();
        OEChem.OESmilesToMol(mol, "Cc1cc(cc(c1[N+](=O)[O-])F)[N+]#C");
        OEChem.OEMMFFAtomTypes(mol);
        OEChem.OEMMFF94PartialCharges(mol);

        OEDepict.OEPrepareDepiction(mol);

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(350, 250, OEScale.
↪AutoScale);
        opts.SetAtomColorStyle(OEAtomColorStyle.WhiteMonochrome);
        opts.SetTitleLocation(OETitleLocation.Hidden);
        OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

        OEColorStop coloranion = new OEColorStop(-1.0, OEChem.OEDarkRed);
        OEColorStop colorcation = new OEColorStop(+1.0, OEChem.OEDarkBlue);
        OELinearColorGradient grad = new OELinearColorGradient(coloranion, colorcation);
        grad.AddStop(new OEColorStop(0.0, OEChem.OEWhite));

        ColorCharge colorcharge= new ColorCharge(grad);
```

(continues on next page)

(continued from previous page)

```
        OEGrapheme.OEAddGlyph(disp, colorcharge, new IsTrueAtom());

        OEDepict.OERenderMolecule("AnnotatePartialCharge.png", disp);
        return 0;
    }
}
```

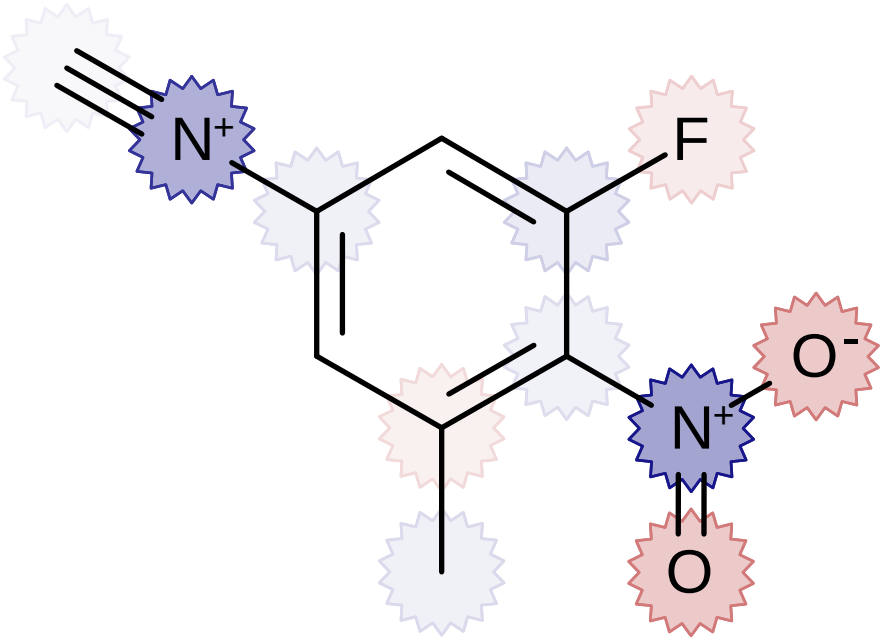


Fig. 3: **Example of user-defined annotation**

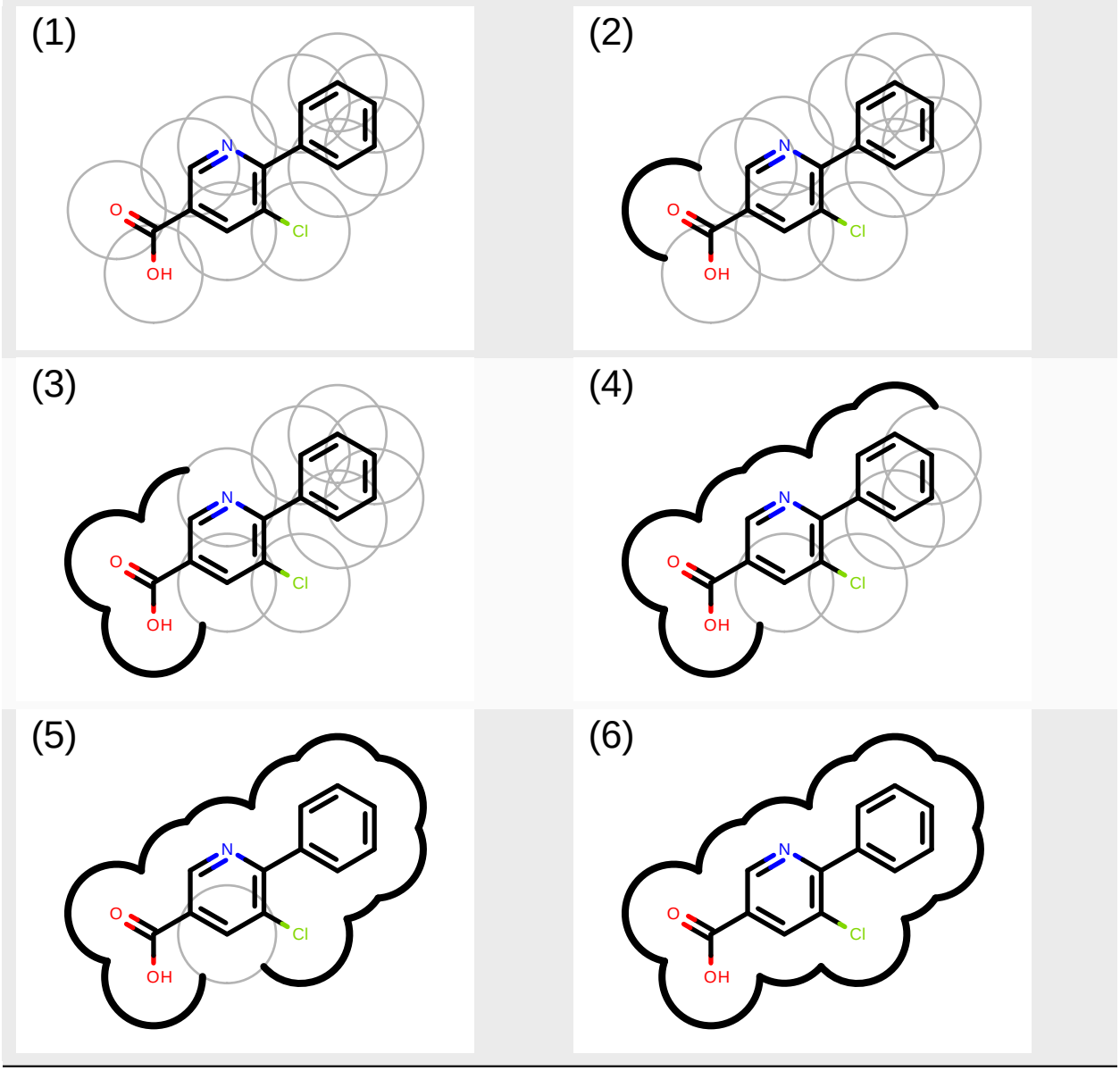
See also

- *OEAtomGlyphBase* class
- *OEBondGlyphBase* class
- `OEAddGlyph` function
- OELinearColorGradient class

2.2 Drawing a Molecule Surface

The 2D representation of the molecule surface is calculated by first drawing circles around each atom of the molecule and then identifying the external arc segments that define a continuous surface around the molecule. This process is illustrated in the table *Table: Calculation of the 2D representation of a molecule surface*

Table 2: **Calculation of the 2D representation of a molecule surface**



The following *Listing 1* example shows how to the display a molecule surface. After creating an image object and preparing the molecule for 2D depiction, an arc drawing functor is added to each atom of the molecule by using the *OESetSurfaceArcFxn* function. The surface then can be drawn by calling the *OEDraw2DSurface* function that calculates the arc segments that form a continuous curve around the molecule. The image created by *Listing 1* is shown in *Figure: Example of surface drawing*.

Listing 1: Example of surface drawing

```
double scale  = OEScale.AutoScale;
OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(width, height, scale);
opts.SetTitleLocation(OETitleLocation.Hidden);
opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts));
```

(continues on next page)

(continued from previous page)

```
OEEyelashArcFxn arcfxn= new OEEyelashArcFxn(new OEPen(OEChem.OEGrey, OEChem.OEGrey));

foreach (OEAtomBase atom in mol.GetAtoms())
{
    OEGrapheme.OESetSurfaceArcFxn(mol, atom, arcfxn);
}

OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);
OEGrapheme.OEDraw2DSurface(disp);
```

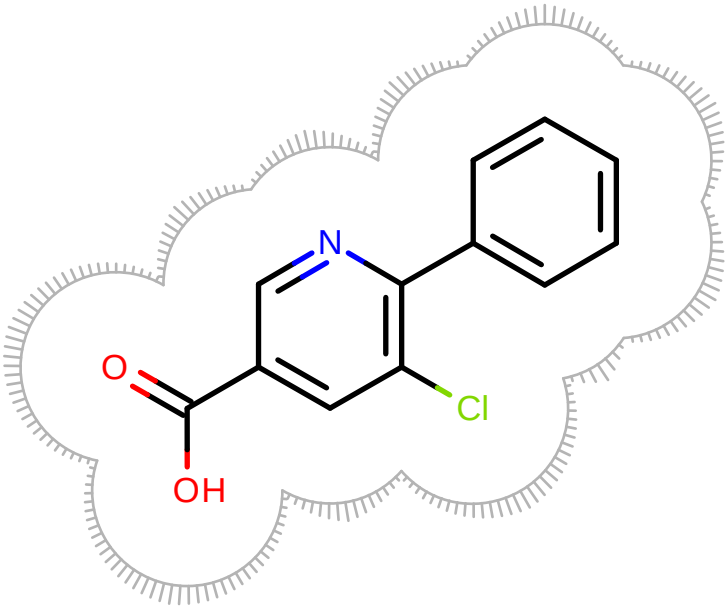
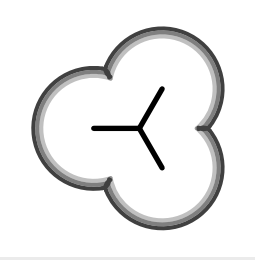
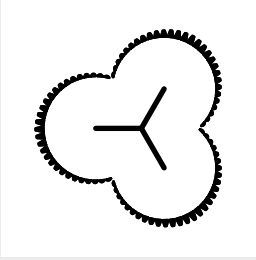
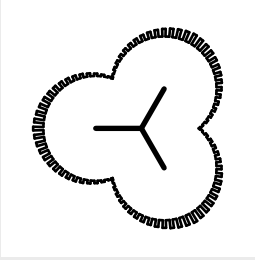
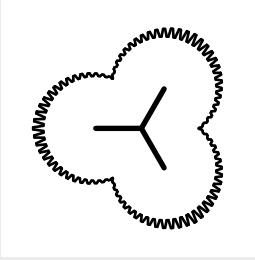
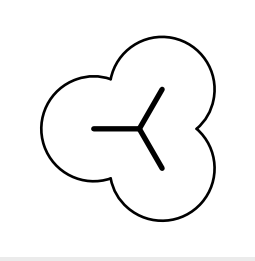
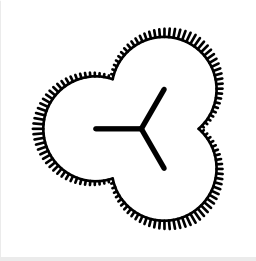
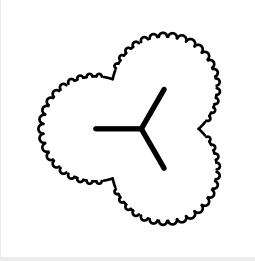
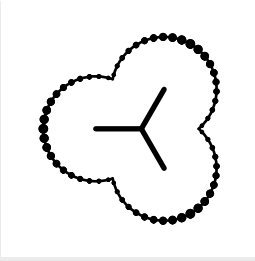
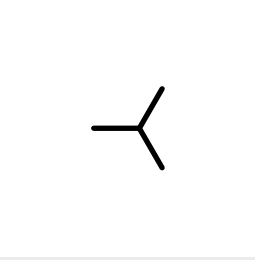
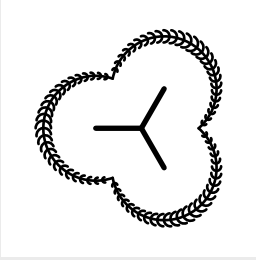
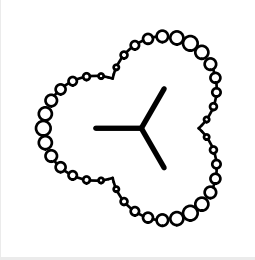
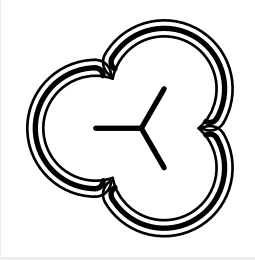
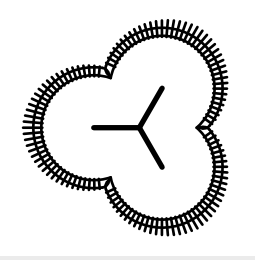
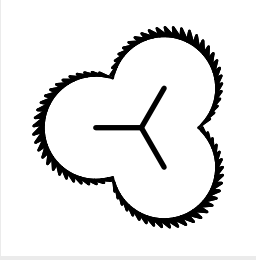
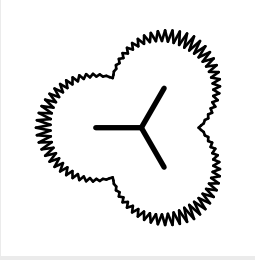
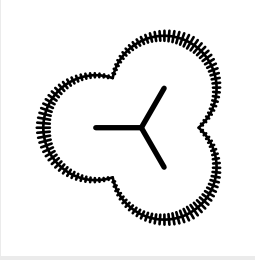
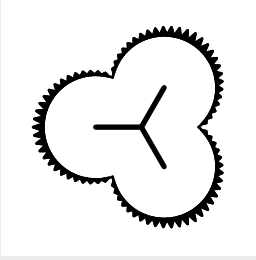
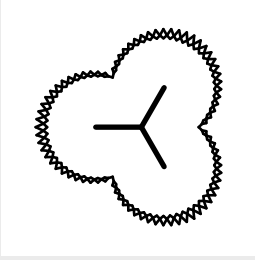


Fig. 4: **Example of surface drawing**

The *Listing 1* example uses the *OESurfaceArcStyle.Eyelash* style to draw the molecule surface. The following table lists all surface drawing styles that are available in **Grapheme TK**.

Table 3: **Supported surface drawing styles**

			
OEAlphaRainbowArcFxn	*OEBrickRoadArcFxn*	*OECastleArcFxn*	*OECogArcFxn*
			
OEDefaultArcFxn	*OEEyelashArcFxn*	*OEFlowerArcFxn*	*OENecklaceArcFxn*
			
OENullArcFxn	*OEOliveBranchArcFxn*	*OEPearlsArcFxn*	*OERaceTrackArcFxn*
			
OEStitchArcFxn	*OESunArcFxn*	*OEWreathArcFxn*	*OERailroadTrackArcFxn*
			
	OESawArcFxn	*OESimpsonArcFxn*	

User-defined surface drawing functors can be implemented by deriving from the *OESurfaceArcFxnBase* abstract base class. In the `Listing 2` example, a user defined arc drawing functor is implemented that colors the *eyelash* arcs according to the atom they belong to. The image created by `Listing 2` is shown in *Figure: Example of user-defined surface drawing*.

Listing 2: Example of user-defined surface drawing

```
public class AtomColorArcFxn : OESurfaceArcFxnBase
{
    public override bool Call(OEImageBase image, OESurfaceArc arc)
    {
        OE2DAtomDisplay adisp = arc.GetAtomDisplay();
        if (adisp == null || !adisp.IsVisible())
            return false;

        OEPen pen = new OEPen();
        OEColor color = adisp.GetLabelFont().GetColor();
        pen.SetForeColor(color);

        OE2DPoint center = arc.GetCenter();
        double radius    = arc.GetRadius();
        double bgnAngle  = arc.GetBgnAngle();
        double endAngle  = arc.GetEndAngle();

        OEGrapheme.OEDrawEyelashSurfaceArc(image, center, bgnAngle, endAngle, radius,␣
↪pen);
        return true;
    }
    public override OESurfaceArcFxnBase CreateCopy()
    {
        AtomColorArcFxn copy = new AtomColorArcFxn();
        copy.ReleaseOwnership();
        return copy;
    }
}
static void draw2DSurface(OEImageBase image, OEMolBase mol) {

    OEDepict.OEPrepareDepiction(mol);

    double scale = OEScale.AutoScale;
    OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(image.GetWidth(), image.
↪GetHeight(), scale);
    opts.SetTitleLocation(OETitleLocation.Hidden);
    opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts));

    AtomColorArcFxn arcfxn = new AtomColorArcFxn();
    foreach (OEAtomBase atom in mol.GetAtoms())
    {
        OEGrapheme.OESetSurfaceArcFxn(mol, atom, arcfxn);
    }

    OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);
    OEGrapheme.OEDraw2DSurface(disp);
    OEDepict.OERenderMolecule(image, disp);
}
```

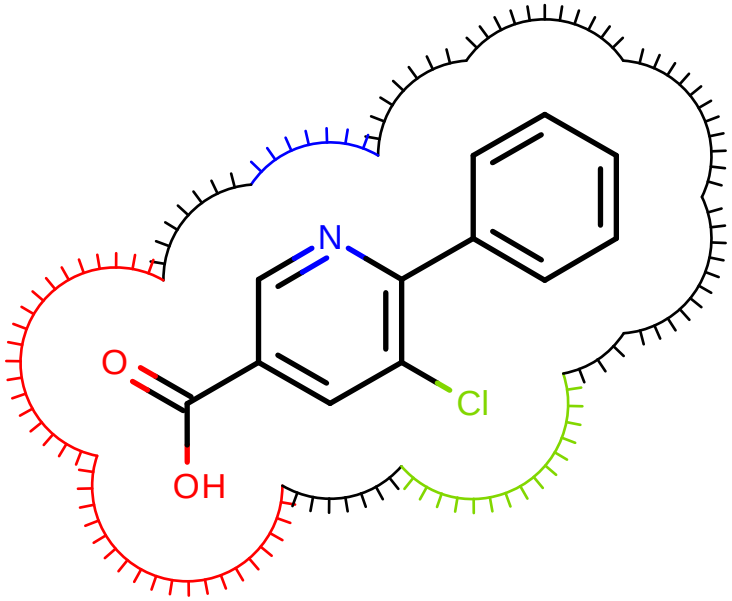


Fig. 5: **Example of user-defined surface drawing**

The *Listing 3* example shows how to project atom properties, such as partial charges into the molecule surface. The image created by *Listing 3* is shown in *Figure: Example of depicting atom properties*.

Listing 3: Example of depicting atom properties on the molecule surface

```
private class AtomPartialChargeArcFxn : OESurfaceArcFxnBase
{
    private OELinearColorGradient colorg;
    private AtomPartialChargeArcFxn() {}
    public AtomPartialChargeArcFxn(OELinearColorGradient cg)
    {
        colorg = new OELinearColorGradient(cg);
    }

    public override bool Call(OEImageBase image, OESurfaceArc arc)
    {
        OE2DAtomDisplay adisp = arc.GetAtomDisplay();
        if (adisp == null || !adisp.IsVisible())
            return false;

        OEAtomBase atom = adisp.GetAtom();
        if (atom == null)
            return false;

        double charge = atom.GetPartialCharge();
        if (charge == 0.0)
            return true;
        OEColor color = colorg.GetColorAt(charge);
```

(continues on next page)

(continued from previous page)

```
        OEPen pen = new OEPen();
        pen.SetForeColor(color);
        pen.SetLineWidth(2.0);

        OE2DPoint center = arc.GetCenter();
        double radius = arc.GetRadius();
        double bAngle = arc.GetBgnAngle();
        double eAngle = arc.GetEndAngle();

        double edgeAngle = 5.0;
        uint dir = OEPatternDirection.Outside;
        double patternAngle = 10.0;
        OEGrapheme.OEDrawBrickRoadSurfaceArc(image, center, bAngle, eAngle, radius,
                                             pen, edgeAngle, dir, patternAngle);
        return true;
    }

    public override OESurfaceArcFxnBase CreateCopy()
    {
        AtomPartialChargeArcFxn copy = new AtomPartialChargeArcFxn(colorg);
        copy.ReleaseOwnership();
        return copy;
    }
}

static void draw2DSurfacePartialCharge(OEImageBase image, OEMolBase mol) {

    OEDepict.OEPrepareDepiction(mol);

    OEChem.OEMMFFAtomTypes(mol);
    OEChem.OEMMFF94PartialCharges(mol);

    double scale  = OEScale.AutoScale;
    OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(image.GetWidth(), image.
↪GetHeight(), scale);
    opts.SetTitleLocation(OETitleLocation.Hidden);
    opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts));

    OEColorStop coloranion = new OEColorStop(-1.0, OEChem.OEDarkRed);
    OEColorStop colorcation = new OEColorStop(+1.0, OEChem.OEDarkBlue);
    OELinearColorGradient colorg = new OELinearColorGradient(coloranion, colorcation);
    colorg.AddStop(new OEColorStop(0.0, OEChem.OEWhite));

    AtomPartialChargeArcFxn arcfxn = new AtomPartialChargeArcFxn(colorg);

    foreach (OEAtomBase atom in mol.GetAtoms())
    {
        OEGrapheme.OESetSurfaceArcFxn(mol, atom, arcfxn);
    }

    OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);
    OEGrapheme.OEDraw2DSurface(disp);
```

(continues on next page)

(continued from previous page)

```
    OEDepict.OERenderMolecule(image, disp);
}
```

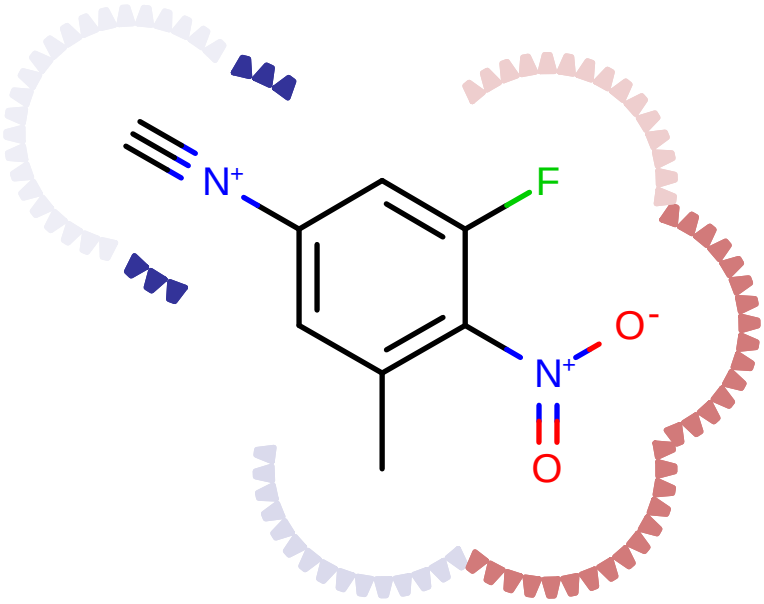


Fig. 6: **Example of depicting atom properties**

See also

- OELinearColorGradient class in the **OEDepict TK** manual

It is also possible to draw multiple 2D surfaces arc-by-arc. In the *Listing 4* example below the arcs returned by the *OEGet2DSurfaceArcs* function for the given atom display and radius are directly rendered below the molecule diagram. In this case it is important to call the *OEGetMoleculeSurfaceScale* function with the largest radius scale of the 2D surfaces drawn. This reduces the scaling of the molecule in order to able to fit the molecule diagram and all of the arcs of the 2D surfaces into the image. The image created by *Listing 4* is shown in *Figure: Example of drawing multiple 2D surfaces*.

Listing 4: Example of drawing multiple 2D surfaces

```
static void draw2DSurface(OE2DMolDisplay disp,  OEUnaryAtomPred atompred,
                          double radius,  OEColor color)
{

    OEPen penA = new OEPen(color, color, OEFill.Off, 2.0);
    OEDefaultArcFxn arcfxnA = new OEDefaultArcFxn(penA);

    OEColor grey = new OEColor(OEChem.OELightGrey);
    OEPen penB = new OEPen(grey, grey, OEFill.Off, 2.0, OEStipple.ShortDash);
```

(continues on next page)

(continued from previous page)

```
    OEDefaultArcFxn arcfxnB = new OEDefaultArcFxn(penB);

    OEImage layer = disp.GetLayer(OELayerPosition.Below);

    foreach (OE2DAtomDisplay adisp in disp.GetAtomDisplays())
    {
        foreach (OESurfaceArc arc in OEGrapheme.OEGet2DSurfaceArcs(disp, adisp, radius))
        {
            if (atompred.Call(adisp.GetAtom()))
                arcfxnA.Call(layer, arc);
            else
                arcfxnB.Call(layer, arc);
        }
    }
}

static void draw2DSurfaces(OEImageBase image, OEMolBase mol)
{

    OEDepict.OEPrepareDepiction(mol);

    double scale =  OEScale.AutoScale;
    OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(image.GetWidth(), image.
↪GetHeight(), scale);
    opts.SetTitleLocation(OETitleLocation.Hidden);
    opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts, 1.50));

    OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

    draw2DSurface(disp, new OEHasAtomicNum(OEElemNo.C), 1.00, OEChem.OEBlack);
    draw2DSurface(disp, new OEHasAtomicNum(OEElemNo.N), 1.25, OEChem.OEDarkBlue);
    draw2DSurface(disp, new OEHasAtomicNum(OEElemNo.O), 1.50, OEChem.OEDarkRed);

    OEDepict.OERenderMolecule(image, disp);

}
```

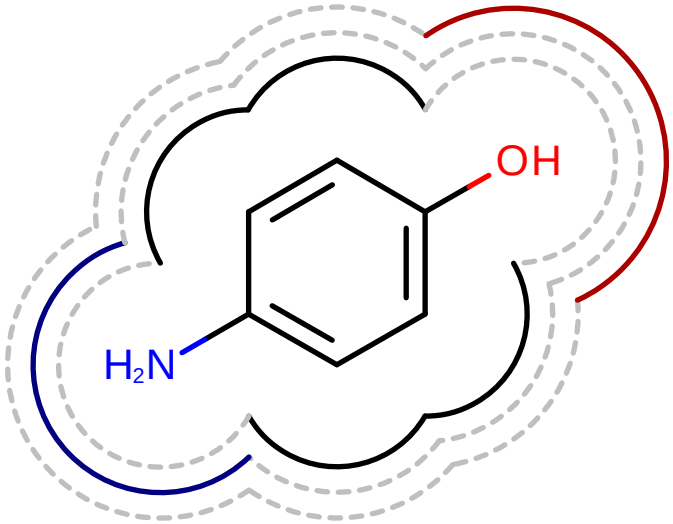


Fig. 7: **Example of drawing multiple 2D surfaces**

See also

OESurfaceArcScale namespace

The last *Listing 5* example shows how to draw a 2D surface with various radii. It draws two surfaces one with a minimum radius scale and one with various radius depending on the covalent radius of the atoms. The *OEGetMoleculeSurfaceScale* function has to be called in this example too with the largest radius scale in order to able to fit the molecule diagram and all of the arcs of the 2D surfaces into the image. The image created by *Listing 5* is shown in *Figure: Example of drawing 2D surface with various radii*.

Listing 5: Example of drawing 2D surface with various radii

```
public static void DrawSurfaces (OEImageBase image, OEMolBase mol)
{
    OEChem.OEAssignCovalentRadii(mol);
    double minradius = OEChem.OEGetCovalentRadius(OEElemNo.H);

    OEDoubleVector radiusScales = new OEDoubleVector((int)mol.GetMaxAtomIdx());

    double maxrscale = Double.MinValue;

    foreach (OEAtomBase atom in mol.GetAtoms())
    {
        double rscale = (atom.GetRadius() - minradius) + OESurfaceArcScale.Minimum;
        radiusScales.Insert((int)atom.GetIdx(), rscale);
        maxrscale = Math.Max(maxrscale, rscale);
```

(continues on next page)

(continued from previous page)

```
        }

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(image.GetWidth(),
                                                               image.GetHeight(),␣
↪OEScale.AutoScale);
        opts.SetTitleLocation(OETitleLocation.Hidden);
        opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts, maxrscale));

        OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

        OEImage layer = disp.GetLayer(OELayerPosition.Below);

        OEPen penA = new OEPen(OEChem.OELightGrey, OEChem.OELightGrey,
                               OEFill.Off, 2.0, OEStipple.ShortDash);
        OEDefaultArcFxn arcfxnA = new OEDefaultArcFxn(penA);

        foreach (OESurfaceArc arc in OEGrapheme.OEGet2DSurfaceArcs(disp,␣
↪OESurfaceArcScale.Minimum))
        {
            arcfxnA.Call(layer, arc);
        }

        OEPen penB = new OEPen(OEChem.OEGrey, OEChem.OEGrey, OEFill.Off, 2.0);
        OEDefaultArcFxn arcfxnB = new OEDefaultArcFxn(penB);

        foreach (OESurfaceArc arc in OEGrapheme.OEGet2DSurfaceArcs(disp, radiusScales))
        {
            arcfxnB.Call(layer, arc);
        }

        OEDepict.OERenderMolecule(image, disp);
    }
```

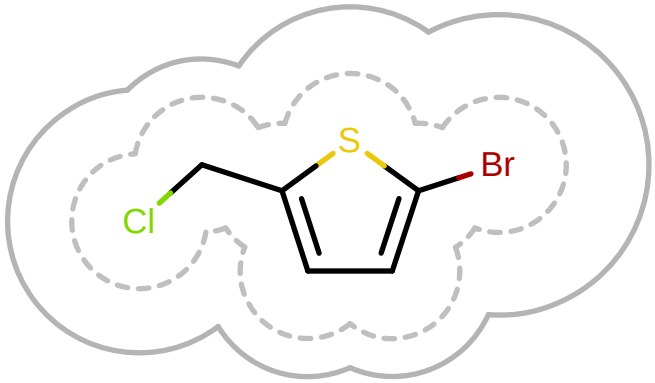


Fig. 8: **Example of drawing 2D surface with various radii**

See also

`OEGetCovalentRadius` function in the **OEChem TK** manual

2.3 Drawing a Ligand-Protein Complex Surface

The previous chapter demonstrates how to draw and customize 2D molecule surfaces. **Grapheme TK** also provides a method that depicts a ligand-protein complex in which the arcs of the 2D surface of the molecule are annotated based on the distance between the accessible surfaces of the ligand and the surrounding protein (`OEMakeAccessibleSurface`). Based on this calculation, the atoms of the ligand are divided into four categories which are then inherited by the corresponding arcs when the 2D molecule surface of the ligand is drawn. These four types are the following:

1. **Solvent** (*OEDefaultSolventArcFxn*)

   A surface arc is depicted using the *solvent* style if the corresponding atom is accessible to a solvent. (See *Figure: Example of the 'solvent' arc style*)



Fig. 9: **Example of the 'solvent' arc style**

2. **Cavity** (*OEDefaultCavityArcFxn*)

A surface arc is depicted using the *cavity* style if in the region of the corresponding atom, a cavity is detected between the ligand and the receptor molecule surfaces. The cavity is large enough to allow expanding the ligand in this region without bumping into the receptor. (See *Figure: Example of the cavity arc style*)

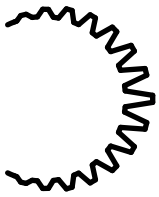


Fig. 10: **Example of the 'cavity' arc style**

3. **Void** (*OEDefaultVoidArcFxn*)

   A surface arc is depicted using the *void* style, if in the region of the corresponding atom a small interstice is detected between the ligand and the receptor molecule surfaces. (See *Figure: OEDefaultBuriedArcFxn*)

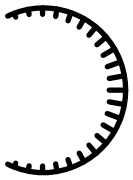


Fig. 11: **Example of the 'void' arc style**

4. **Buried** (*OEDefaultBuriedArcFxn*)

   A surface arc is depicted using the *buried* style, if in the region of the corresponding atom the ligand is tightly fit to the receptor. (See *Figure: Example of the buried arc style*)

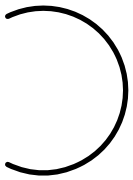


Fig. 12: **Example of the 'buried' arc style**

Note

Even though, **Spicoli TK** is used to generate the accessible surfaces of the ligand and the receptor, no **Spicoli TK** license is required to call the `OEAddComplexSurfaceArcs` function.

The following `Listing 1` example show how to display the surface of the ligand-protein complex. After importing the ligand and the receptor structures, the `OEAddComplexSurfaceArcs` function is called to generate the accessible molecule surfaces (using **Spicoli TK**) and assign the surface drawing styles (*solvent*, *buried*, *void*, or *cavity*) for the

atoms of the ligand. These styles are the used to draw the arcs of the molecule surface when the *OEDraw2DSurface* function is invoked. The *OEGetMoleculeSurfaceScale* function is used to reduce the scaling of the molecule in order to be able to fit both the molecule diagram and the molecule surface into the image.

Listing 1: Example of complex surface drawing

```
public class DrawComplexSurface
{
    public static OEGraphMol ImportMolecule (String filename)
    {
        oemolistream ifs = new oemolistream();
        OEGraphMol mol = new OEGraphMol();
        if (!ifs.open(filename))
        {
            OEChem.OEThrow.Fatal("Unable to open " + filename + " for reading");
        }

        if (!OEChem.OEReadMolecule(ifs, mol))
        {
            OEChem.OEThrow.Fatal("Unable to read molecule in " + filename);
        }

        OEChem.OEAssignBondiVdWRadii(mol);
        OEChem.OESuppressHydrogens(mol);

        return mol;
    }

    public static int Main(String[] args)
    {
        if (args.Length != 2)
        {
            OEChem.OEThrow.Usage("DrawComplexSurface <receptor> <ligand>");
        }

        OEGraphMol receptor = ImportMolecule(args[0]);

        OEGraphMol ligand = ImportMolecule(args[1]);

        uint width  = 450;
        uint height = 350;

        OEGrapheme.OEAddComplexSurfaceArcs(ligand, receptor);

        OEGrapheme.OEPrepareDepictionFrom3D(ligand);
        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(width, height, OEScale.
↪AutoScale);
        opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(ligand, opts));

        OE2DMolDisplay disp = new OE2DMolDisplay(ligand, opts);
        OEGrapheme.OEDraw2DSurface(disp);

        OEDepict.OERenderMolecule("DrawComplexSurface.png", disp);
```

(continues on next page)

(continued from previous page)

```
        return 0;
    }
}
```

The following snapshot (*Figure: Dihydrofolate reductase*) displays the ligand of the *2w3a* PDB complex in VIDA with the accessible surface of the protein. The image shown in the *Figure: The Grapheme representation of the 2w3a complex* is created by `Listing 1` code for the same ligand-protein complex.

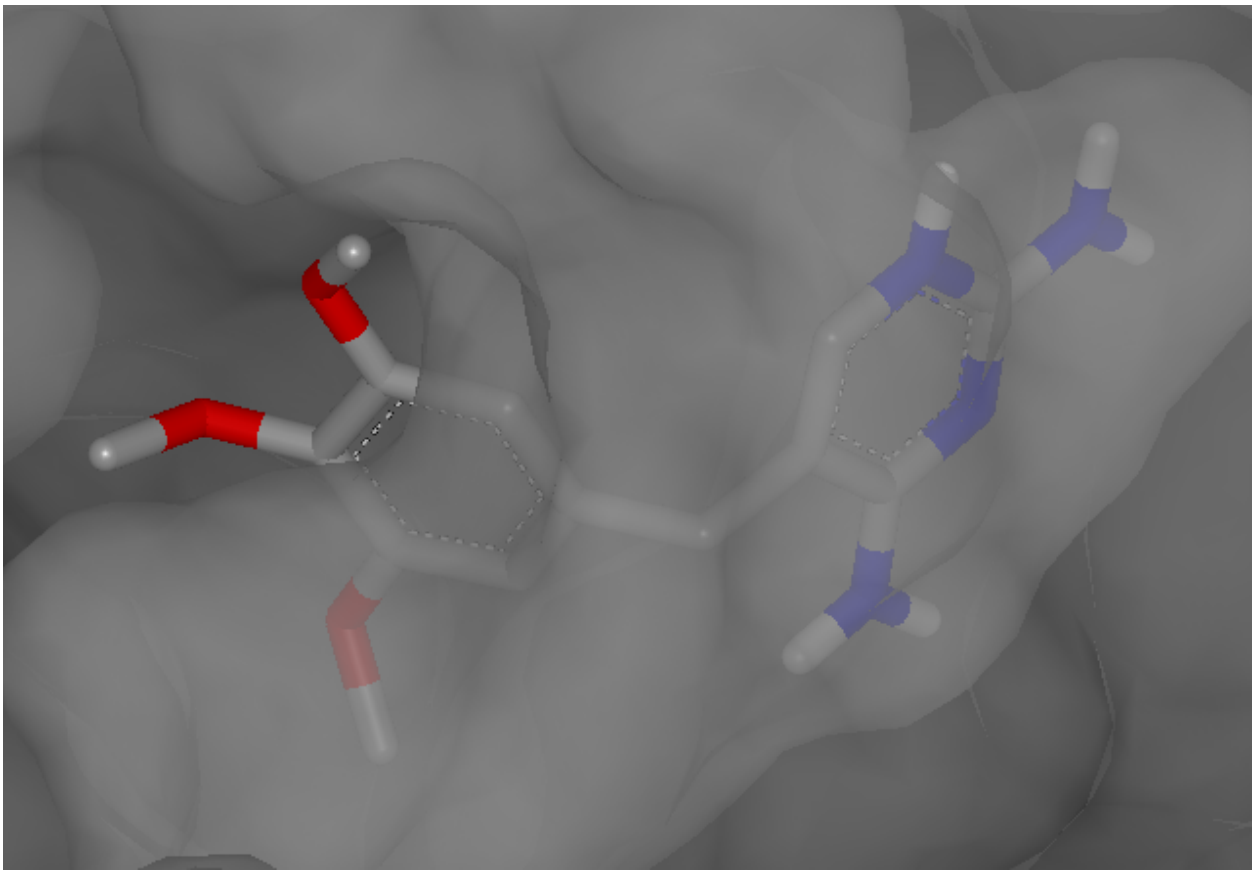


Fig. 13: **Dihydrofolate reductase complexed with trimethoprim in VIDA (PDB 2w3a)**

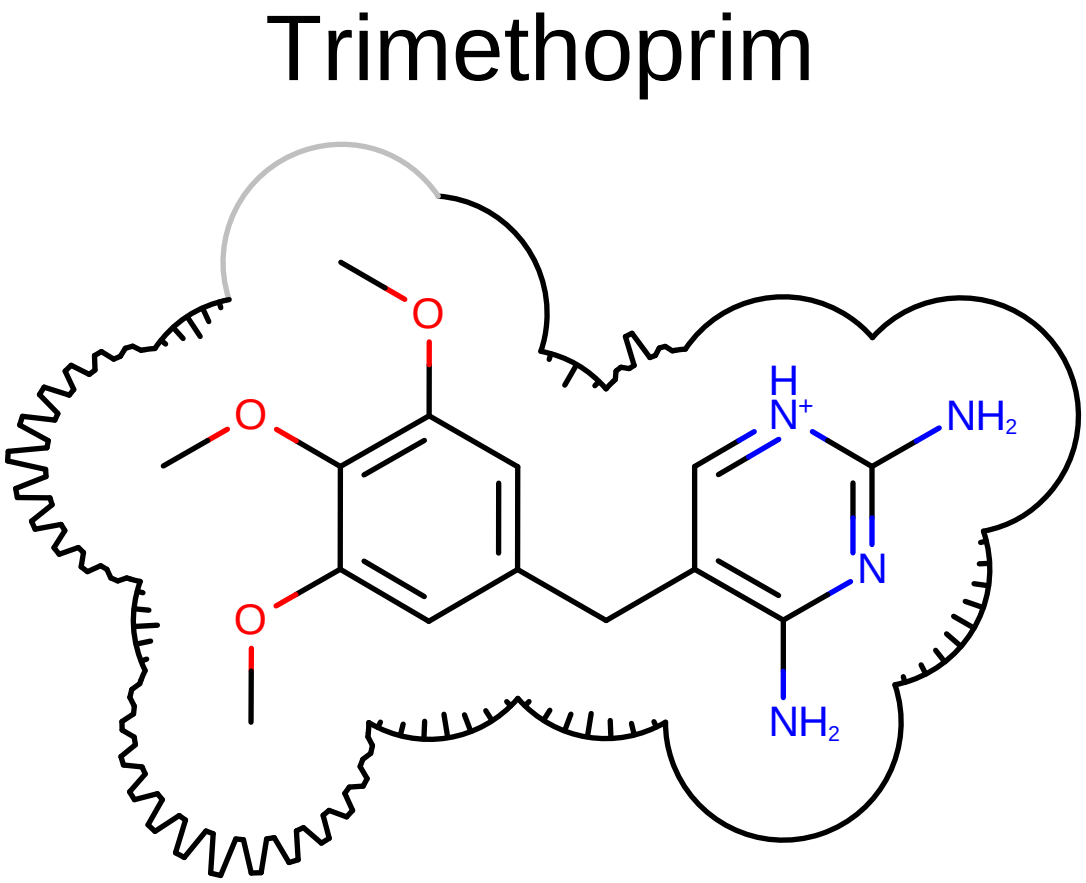


Fig. 14: **The Grapheme representation of the 2w3a complex**

User-defined *solvent* and *buried* surface drawing styles can be implemented by deriving from the *OESurfaceArcFxnBase* base class. Similarly, user-defined *cavity* and *void* surface drawing styles can be implemented by deriving from the *OEComplexSurfaceArcFxnBase* base class. This later abstract class also stores the relative distance calculated between the ligand and the receptor. The following code snippet shows how to use this *depth* to emphasize the volume of the cavity by adjusting the size of the *spikes* when drawing the surface arcs using the *OEDrawSunSurfaceArc* function.

```
    private static void DrawSurfaceArc(OEImageBase image, double depth, OESurfaceArc arc,
↪ OEPen pen)
    {
      double edgeAngle = 5.0;
      double patternAngle = 5.0;
      double minPatternWidthRatio = 0.05;
      double maxPatternWidthRatio = 0.10 * depth;
      uint patternDirection = OEPatternDirection.Outside;
      OEGrapheme.OEDrawSunSurfaceArc(image, arc.GetCenter(), arc.GetBgnAngle(), arc.
↪GetEndAngle(),
                                     arc.GetRadius(), pen, edgeAngle,
                                     patternDirection, patternAngle,
                                     minPatternWidthRatio, maxPatternWidthRatio);
    }
```

(continues on next page)

(continued from previous page)

```
    private class SolventArcFxn : OESurfaceArcFxnBase
    {
        public override bool Call(OEImageBase image, OESurfaceArc arc)
        {
            OEPen pen = new OEPen(OEChem.OELightGrey, OEChem.OELightGrey);
            pen.SetLineWidth(0.5);
            OEGrapheme.OEDrawDefaultSurfaceArc(image, arc.GetCenter(), arc.GetBgnAngle(),
                                               arc.GetEndAngle(), arc.GetRadius(), pen);
            return true;
        }
        public override OESurfaceArcFxnBase CreateCopy()
        {
            SolventArcFxn copy = new SolventArcFxn();
            copy.ReleaseOwnership();
            return copy;
        }
    }

    private class BuriedArcFxn : OESurfaceArcFxnBase
    {
        public override bool Call(OEImageBase image, OESurfaceArc arc)
        {
            OEPen pen = new OEPen(OEChem.OEGrey, OEChem.OEGrey);
            pen.SetLineWidth(2.0);
            OEGrapheme.OEDrawDefaultSurfaceArc(image, arc.GetCenter(), arc.GetBgnAngle(),
                                               arc.GetEndAngle(), arc.GetRadius(), pen);
            return true;
        }
        public override OESurfaceArcFxnBase CreateCopy()
        {
            BuriedArcFxn copy = new BuriedArcFxn();
            copy.ReleaseOwnership();
            return copy;
        }
    }

    private class CavityArcFxn : OEComplexSurfaceArcFxnBase
    {
        public CavityArcFxn() {}
        public CavityArcFxn(CavityArcFxn rhs) : base(rhs) {}
        public override bool Call(OEImageBase image, OESurfaceArc arc)
        {
            OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack);
            pen.SetLineWidth(2.0);
            DrawSurfaceArc(image, GetDepth(), arc, pen);
            return true;
        }
        public override OEComplexSurfaceArcFxnBase CreateComplexCopy()
        {
            CavityArcFxn copy = new CavityArcFxn(this);
            copy.ReleaseOwnership();
            return copy;
```

(continues on next page)

(continued from previous page)

```
        }
        public override OESurfaceArcFxnBase CreateCopy()
        {
            CavityArcFxn copy = new CavityArcFxn(this);
            copy.ReleaseOwnership();
            return copy;
        }
    }

    private class VoidArcFxn : OEComplexSurfaceArcFxnBase
    {
        public VoidArcFxn() {}
        public VoidArcFxn(VoidArcFxn rhs) : base(rhs) {}
        public override bool Call(OEImageBase image, OESurfaceArc arc)
        {
            OEPen pen = new OEPen(OEChem.OEDarkGrey, OEChem.OEDarkGrey);
            pen.SetLineWidth(2.0);
            DrawSurfaceArc(image, GetDepth(), arc, pen);
            return true;
        }
        public override OEComplexSurfaceArcFxnBase CreateComplexCopy()
        {
            VoidArcFxn copy = new VoidArcFxn(this);
            copy.ReleaseOwnership();
            return copy;
        }
        public override OESurfaceArcFxnBase CreateCopy()
        {
            VoidArcFxn copy = new VoidArcFxn(this);
            copy.ReleaseOwnership();
            return copy;
        }
    }
```

These arc drawing predicates then can be used to draw the surface of the ligand-protein complex when calling the *OEAddComplexSurfaceArcs* function. The image created is shown in *Figure: Example of user-defined surface drawing*.

```
SolventArcFxn sfxn = new SolventArcFxn();
BuriedArcFxn  bfxn = new BuriedArcFxn();
CavityArcFxn  cfxn = new CavityArcFxn();
VoidArcFxn    vfxn = new VoidArcFxn();
OEGrapheme.OEAddComplexSurfaceArcsSpecial(ligand, receptor, sfxn, bfxn, cfxn, vfxn);
```

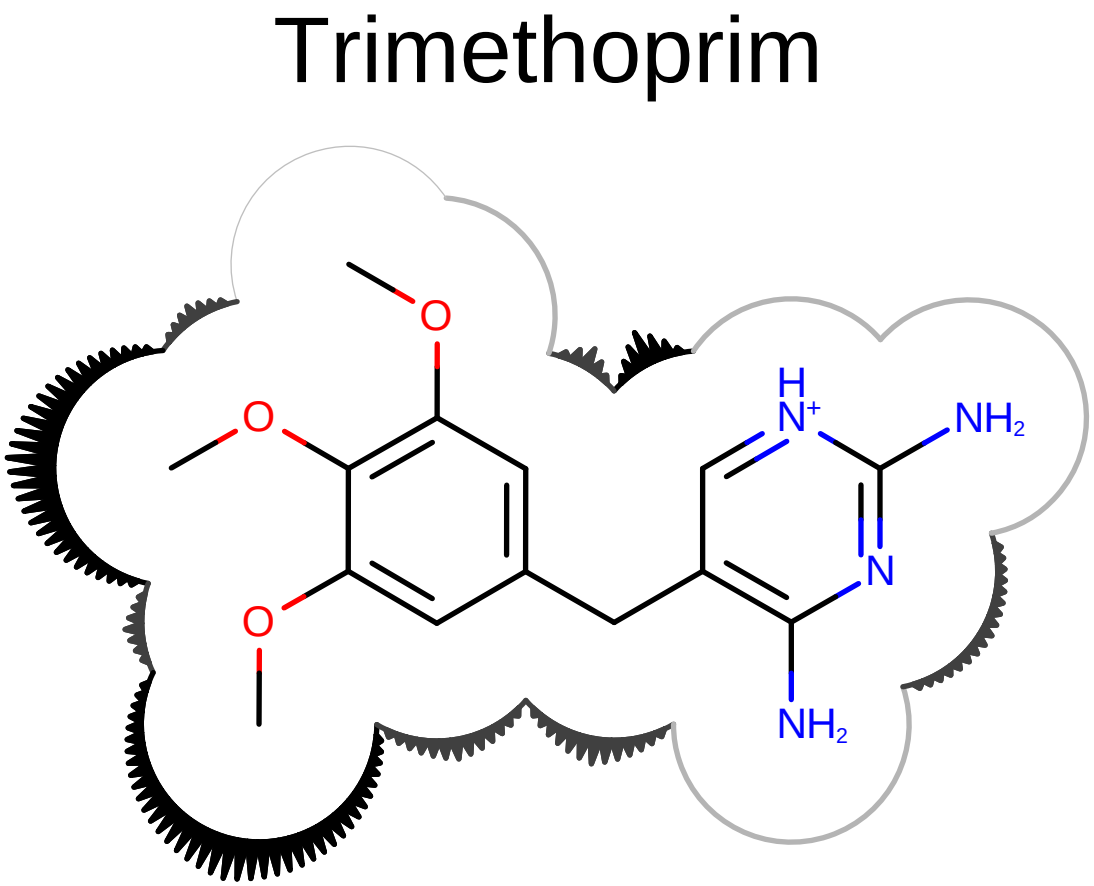


Fig. 15: **Example of user-defined surface drawing**

See also

- Surface Generation chapter in the **Spicoli TK** manual

2.4 Depicting Property Maps

The previous chapters give examples about how to depict atom properties either by using glyphs (see *Annotating Atoms and Bonds* chapter) or projecting them into a 2D molecule surface (see *Drawing a Molecule Surface* chapter). **Grapheme TK** can also map atom properties into a 2D grid, called a property map, using a Gaussian function. After the grid is generated, colors are rendered to each cell using the OELinearColorGradient class that interpolates colors between a specified range.

The following `Listing 1` example demonstrates how to depict partial charges using the property map. After constructing the molecule and preparing it for depiction, the MMFF partial charges are calculated. These partial charges are attached to the corresponding atom as generic data with a specific *tagname*. A *OE2DPropMap* object is then generated by specifying three colors:

- the background color of the property map
- one that represents negative values (in this case charges), and

- the color that represents positive values in the property map

When the *OE2DPropMap.Render* method is called with the *tagname*, the properties that were attached to the atoms as generic data are retrieved and a 2D grid is generated along with an OELinearColorGradient object that is used to assign colors to the *cells* of the grid. The colored grid is then rendered into the *below* layer of the OE2DMolDisplay object, *i.e.*, it appears underneath the molecular diagram. The image created by *Listing 1* is shown in *Figure: Example of depicting a property map*.

Listing 1: Depicting partial charges using property map

```
public class Draw2DPropMapPartialCharge
{
    public static void setPartialCharge(OEMolBase mol, string tagname)
    {
        OEChem.OEMMFFAtomTypes(mol);
        OEChem.OEMMFF94PartialCharges(mol);
        uint tag = OEChem.OEGetTag(tagname);
        foreach (OEAtomBase ai in mol.GetAtoms())
        {
            ai.SetDoubleData(tag, ai.GetPartialCharge());
        }
    }

    public static int Main(string[] args)
    {
        OEGraphMol mol = new OEGraphMol();
        OEChem.OESmilesToMol(mol, "Cc1cc(cc(c1[N+](=O)[O-])F)[N+]#C");
        OEDepict.OEPrepareDepiction(mol);

        string tagname = "PartialCharge";
        setPartialCharge(mol, tagname);

        OEDepict.OEPrepareDepiction(mol);

        uint width  = 450;
        uint height = 350;

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(width, height, OEScale.
↪AutoScale);
        opts.SetAtomColorStyle(OEAtomColorStyle.WhiteMonochrome);
        opts.SetTitleLocation(OETitleLocation.Hidden);
        opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(mol, opts));
        OE2DMolDisplay disp = new OE2DMolDisplay(mol, opts);

        OE2DPropMap propmap = new OE2DPropMap(opts.GetBackgroundColor());
        propmap.SetNegativeColor(OEChem.OEDarkRed);
        propmap.SetPositiveColor(OEChem.OEDarkBlue);
        propmap.Render(disp, tagname);

        OEDepict.OERenderMolecule("Draw2DPropMapPartialCharge.png", disp);
        return 0;
    }
}
```

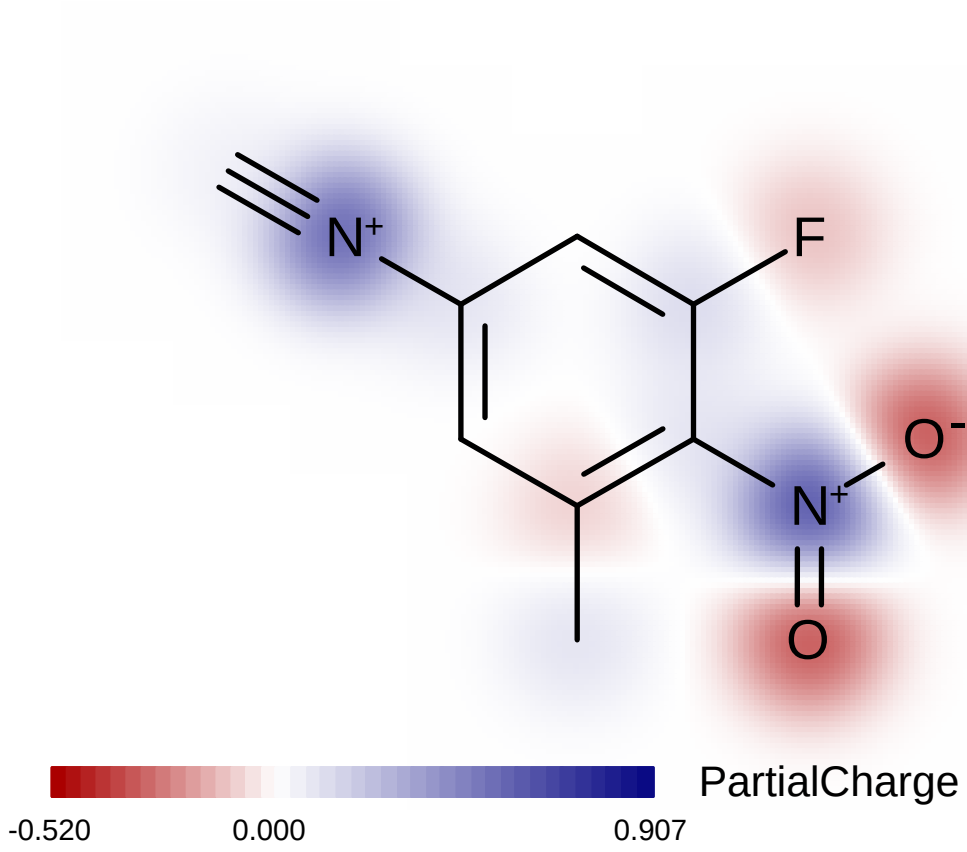


Fig. 16: **Example of depicting a property map**

See also

- *Generic Data* chapter in the **OEChem TK** manual.

When the `OE2DPropMap.Render` method is called, the color gradient is initialized by searching for the minimum and maximum values by the given *tagname*. This means that when more than one molecule is depicted, each depiction will have its own independent value range. (See *Figure: Example of depicting a property map*).

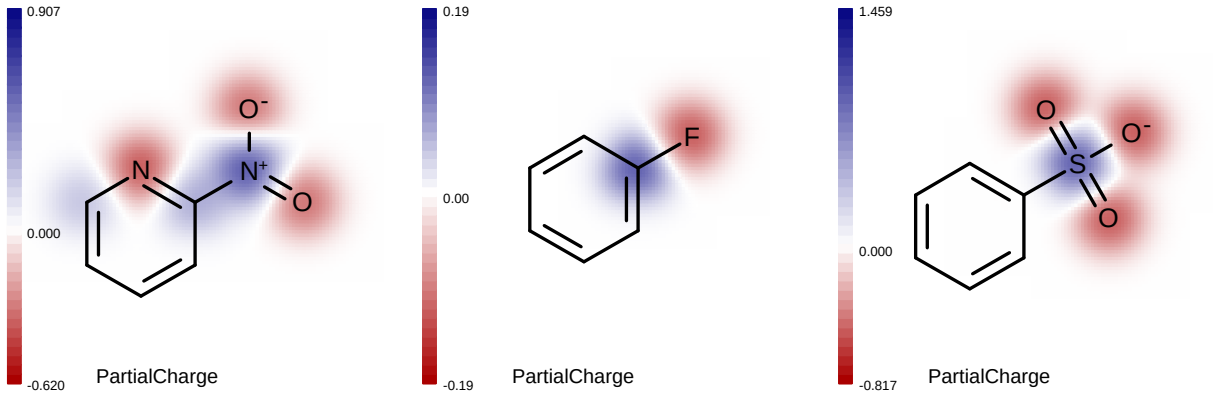


Fig. 17: **Example of depicting multiple property maps**

In the `Listing 2` example, the value range of the property map is pre-set by identifying the minimum and maximum atom partial charges for a whole molecule set. See the result in *Figure: Example of depicting property maps in the same value range*. Each black box around the color gradient legend indicates the range of the atom partial charges for the corresponding molecule.

Listing 2: Depicting multiple molecules using property map

```
public class Draw2DPropMapPChargeMulti
{
    public static void setPartialCharge(OEMolBase mol, string tagname,
                                         ref double minvalue, ref double maxvalue)
    {
        OEChem.OEMMFFAtomTypes(mol);
        OEChem.OEMMFF94PartialCharges(mol);

        uint tag = OEChem.OEGetTag(tagname);
        foreach (OEAtomBase ai in mol.GetAtoms())
        {
            double charge = ai.GetPartialCharge();
            ai.SetDoubleData(tag, charge);
            minvalue = Math.Min(minvalue, charge);
            maxvalue = Math.Max(maxvalue, charge);
        }
    }

    public static int Main(string[] args)
    {
        List<string> smiles = new List<string>();
```

(continues on next page)

(continued from previous page)

```
        smiles.Add("c1ccnc(c1)[N+](=O)[O-]");
        smiles.Add("c1ccccc1F");
        smiles.Add("c1ccc(cc1)S(=O)(=O)[O-]");

        double minvalue = double.MaxValue;
        double maxvalue = double.MinValue;

        string tagname = "PartialCharge";

        List<OEGraphMol> molecules = new List<OEGraphMol>();

        foreach (string s in smiles)
        {
            OEGraphMol mol = new OEGraphMol();
            OEChem.OESmilesToMol(mol, s);
            OEDepict.OEPrepareDepiction(mol);
            setPartialCharge(mol, tagname, ref minvalue, ref maxvalue);
            molecules.Add(mol);
        }

        uint width  = 750;
        uint height = 250;
        OEImage image = new OEImage(width, height);

        uint rows = 1;
        uint cols = 3;
        OEImageGrid grid = new OEImageGrid(image, rows, cols);

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(grid.GetCellWidth(),
                                                               grid.GetCellHeight(),
                                                               OEScale.Default);
        opts.SetAtomColorStyle(OEAtomColorStyle.WhiteMonochrome);
        opts.SetTitleLocation(OETitleLocation.Hidden);

        OE2DPropMap propmap = new OE2DPropMap(opts.GetBackgroundColor());
        propmap.SetNegativeColor(OEChem.OEDarkRed);
        propmap.SetPositiveColor(OEChem.OEDarkBlue);
        propmap.SetLegendLocation(OELegendLocation.Left);

        propmap.SetMinValue(minvalue);
        propmap.SetMaxValue(maxvalue);
        OEImageBaseIter ci = grid.GetCells();
        foreach (OEGraphMol m in molecules)
        {
            OE2DMolDisplay disp = new OE2DMolDisplay(m, opts);
            propmap.Render(disp, tagname);
            OEDepict.OERenderMolecule(ci.Target(), disp);
            ci.Increment();
        }

        OEDepict.OEWriteImage("Draw2DPropMapPartialChargeMulti.png", image);
        return 0;
```

(continues on next page)

(continued from previous page)

```
    }
}
```

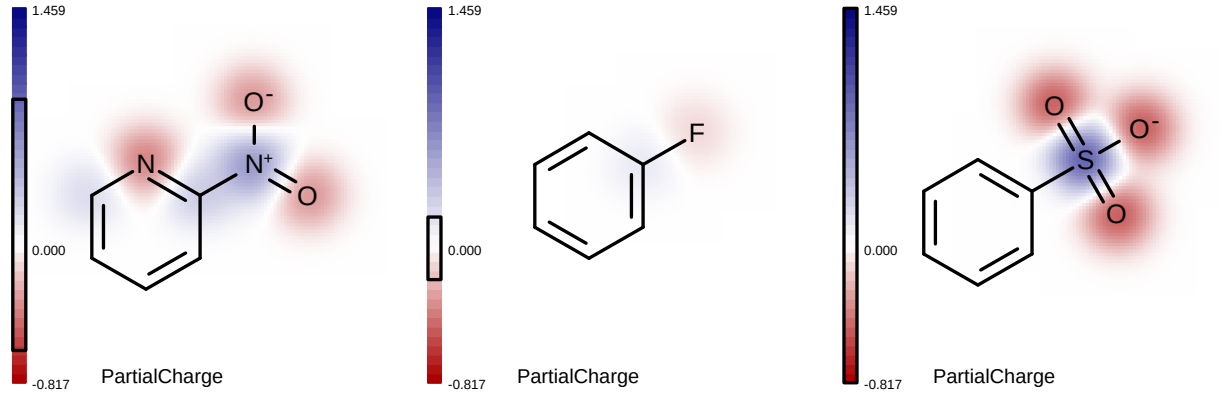


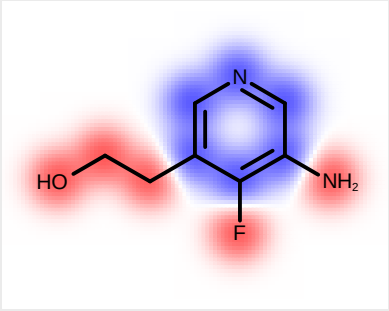
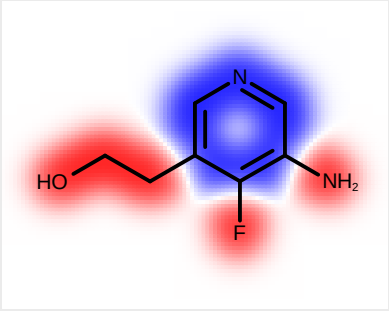
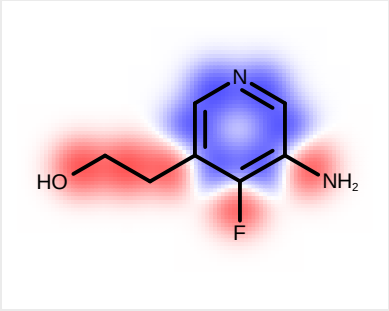
Fig. 18: **Example of depicting a property maps in the same value range**

See also

- OEImageGrid class in the **OEDepict TK** manual

The above examples show how to project atom properties (such as partial charges) into the property map. However, the *OE2DPropMap* class can also be used to visualize bond properties by attaching a value to each bond as a generic data using a *tagname*. When the `OE2DPropMap.Render` method is called with the same *tagname*, the property that was attached to a bond is retrieved and projected at the middle of that bond before applying a Gaussian function to blur out the 2D grid underneath the molecular graph. The *Table: Depicting atom or/and bond properties* table shows the differences between visualizing atom or/and bond properties using the *OE2DPropMap* class.

Table 4: **Depicting atom or/and bond properties**

		
only atom	both atom and bond	only bond

CHAPTER

THREE

EXAMPLES

3.1 Grapheme Examples

The following table lists the currently available **Grapheme TK** examples:

Program	Description
BFactor2Img	depicting BFactor
Complex2Img	depicting active site interactions
ShapeOverlap2PDF	depicting shape and color atom overlaps

Examples:

3.1.1 Depicting BFactor

A program that depicts the B-factor of a ligand and its environment.

See also

- Visualizing Protein-Ligand B-Factor **OpenEye Python Cookbook** example
- Visualizing Protein-Ligand B-Factor Map **OpenEye Python Cookbook** example

Command Line Interface

A description of the command line interface can be obtained by executing the program with the –help argument.

```
prompt> BFactor2Img --help
```

will generate the following output:

```
[-complex] <input> [-out] <output pdf>

Simple parameter list
    -height : Height of output image
    -width : Width of output image

    SplitMolComplex options :
      -ligandname : Ligand name

    input/output options
```

(continues on next page)

(continued from previous page)

```
    -complex : Input filename of the protein complex
    -out : Output filename

  molecule display options
    -aromstyle : Aromatic ring display style
```

Code

```
/*
(C) 2022 Cadence Design Systems, Inc. (Cadence)
All rights reserved.
TERMS FOR USE OF SAMPLE CODE The software below ("Sample Code") is
provided to current licensees or subscribers of Cadence products or
SaaS offerings (each a "Customer").
Customer is hereby permitted to use, copy, and modify the Sample Code,
subject to these terms. Cadence claims no rights to Customer's
modifications. Modification of Sample Code is at Customer's sole and
exclusive risk. Sample Code may require Customer to have a then
current license or subscription to the applicable Cadence offering.
THE SAMPLE CODE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED.  OPENEYE DISCLAIMS ALL WARRANTIES, INCLUDING, BUT
NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE AND NONINFRINGEMENT. In no event shall Cadence be
liable for any damages or liability in connection with the Sample Code
or its use.
*/
/*****************************************************************************
* Depict the BFactor of a ligand and its environment
*****************************************************************************/
using System;
using OpenEye.OEBio;
using OpenEye.OEChem;
using OpenEye.OEDepict;
using OpenEye.OEGrapheme;

public class BFactor2Img
{
    public static int Main(string[] args)
    {
        OEInterface itf = new OEInterface(interfaceData);

        OEDepict.OEConfigureImageWidth (itf, 600.0);
        OEDepict.OEConfigureImageHeight(itf, 600.0);
        OEDepict.OEConfigure2DMolDisplayOptions(itf, OE2DMolDisplaySetup.AromaticStyle);
        OEBio.OEConfigureSplitMolComplexOptions(itf, OESplitMolComplexSetup.LigName);

        if (!OEChem.OEParseCommandLine(itf, args, "BFactor2Img"))
        {
            OEChem.OEThrow.Fatal("Unable to interpret command line");
        }
```

(continues on next page)

(continued from previous page)

```
        string iname = itf.GetString("-complex");
        string oname = itf.GetString("-out");

        oemolistream ifs = new oemolistream();
        if (!ifs.open(iname))
        {
            OEChem.OEThrow.Fatal("Cannot open input file!");
        }

        string ext = OEChem.OEGetFileExtension(oname);
        if (!OEDepict.OEIsRegisteredImageFile(ext))
        {
            OEChem.OEThrow.Fatal("Unknown image type!");
        }

        oeofstream ofs = new oeofstream();
        if (!ofs.open(oname))
        {
            OEChem.OEThrow.Fatal("Cannot open output file!");
        }

        OEGraphMol complexmol = new OEGraphMol();
        if (!OEChem.OEReadMolecule(ifs, complexmol))
        {
            OEChem.OEThrow.Fatal("Unable to read molecule from " + iname);
        }

        if (!OEChem.OEHasResidues(complexmol))
        {
            OEChem.OEPerceiveResidues(complexmol, OEPreserveResInfo.All);
        }

        // Separate ligand and protein

        OESplitMolComplexOptions sopts = new OESplitMolComplexOptions();
        OEBio.OESetupSplitMolComplexOptions(sopts, itf);

        OEGraphMol ligand = new OEGraphMol();
        OEGraphMol protein = new OEGraphMol();
        OEGraphMol water = new OEGraphMol();
        OEGraphMol other = new OEGraphMol();

        OEBio.OESplitMolComplex(ligand, protein, water, other, complexmol, sopts);

        if (ligand.NumAtoms() == 0)
        {
            OEChem.OEThrow.Fatal("Cannot separate complex!");
        }

        double maxdistance = 4.0;

        // Calculate average BFactor of the whole complex
```

(continues on next page)

(continued from previous page)

```
        double avgbfactor = GetAverageBFactor(complexmol);

        //# Calculate minimum and maximum BFactor of the ligand and its environment
        SetMinAndMaxBFactor(ligand, protein, maxdistance);

        OEChem.OEThrow.Info(string.Format("Average BFactor of the complex = {0:0.00}",␣
↪avgbfactor));
        OEChem.OEThrow.Info(string.Format("Minimum BFactor of the ligand and its␣
↪environment = {0:0.00}", minbfactor));
        OEChem.OEThrow.Info(string.Format("Maximum BFactor of the ligand and its␣
↪environment = {0:0.00}", maxbfactor));

        string stag = "avg residue bfactor";
        uint itag = OEChem.OEGetTag(stag);
        SetAverageBFactorOfNearbyProteinAtoms(ligand, protein, itag, maxdistance);

        // Create image
        double imagewidth  = OEDepict.OEGetImageWidth(itf);
        double imageheight = OEDepict.OEGetImageHeight(itf);
        OEImage image = new OEImage(imagewidth, imageheight);

        OEImageFrame mframe = new OEImageFrame(image, imagewidth, imageheight * 0.90,␣
↪new OE2DPoint(0.0, 0.0));
        OEImageFrame lframe = new OEImageFrame(image, imagewidth, imageheight * 0.10,␣
↪new OE2DPoint(0.0, imageheight * 0.90));

        OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(mframe.GetWidth(), mframe.
↪GetHeight(), OEScale.AutoScale);
        OEDepict.OESetup2DMolDisplayOptions(opts, itf);
        opts.SetAtomColorStyle(OEAtomColorStyle.WhiteMonochrome);

        // Create BFactor color gradi    # Create BFactor color gradient
        OELinearColorGradient colorg = new OELinearColorGradient();
        colorg.AddStop(new OEColorStop(  0.0, OEChem.OEDarkBlue));
        colorg.AddStop(new OEColorStop( 10.0, OEChem.OELightBlue));
        colorg.AddStop(new OEColorStop( 25.0, OEChem.OEYellowTint));
        colorg.AddStop(new OEColorStop( 50.0, OEChem.OERed));
        colorg.AddStop(new OEColorStop(100.0, OEChem.OEDarkRose));

        // Prepare ligand for depiction
        OEGrapheme.OEPrepareDepictionFrom3D(ligand);
        BFactorArcFxn arcfxn = new BFactorArcFxn(colorg, itag);
        foreach (OEAtomBase ai in ligand.GetAtoms())
            OEGrapheme.OESetSurfaceArcFxn(ligand, ai, arcfxn);
        opts.SetScale(OEGrapheme.OEGetMoleculeSurfaceScale(ligand, opts));

        // Render ligand and visualize Bfactor
        OE2DMolDisplay disp = new OE2DMolDisplay(ligand, opts);
        ColorLigandAtomByBFactor colorbfactor = new ColorLigandAtomByBFactor(colorg);
        OEGrapheme.OEAddGlyph(disp, colorbfactor, new OEIsTrueAtom());
        OEGrapheme.OEDraw2DSurface(disp);
        OEDepict.OERenderMolecule(mframe, disp);
```

(continues on next page)

(continued from previous page)

```
        // Draw color gradient
        OEColorGradientDisplayOptions copts = new OEColorGradientDisplayOptions();
        copts.SetColorStopPrecision(1);
        copts.AddMarkedValue(avgbfactor);
        copts.SetBoxRange(minbfactor, maxbfactor);

        OEGrapheme.OEDrawColorGradient(lframe, colorg, copts);

        OEDepict.OEWriteImage(oname, image);

        return 0;
    }

    static double GetAverageBFactor(OEMolBase mol)
    {

        double sumbfactor = 0.0;
        foreach (OEAtomBase atom in mol.GetAtoms())
        {
            OEResidue res = OEChem.OEAtomGetResidue(atom);
            sumbfactor += res.GetBFactor();
        }
        double avgbfactor = sumbfactor / mol.NumAtoms();
        return avgbfactor;
    }

    static bool ConsiderResidueAtom(OEAtomBase atom, OEResidue res)
    {
        if (atom.GetAtomicNum() == OEElemNo.H)
        {
            return false;
        }
        if (res.GetName().Equals("HOH"))
        {
            return false;
        }
        return true;
    }

    static double minbfactor = double.MaxValue;
    static double maxbfactor = Double.MinValue;

    static void SetMinAndMaxBFactor(OEMolBase ligand, OEMolBase protein, double␣
↪maxdistance)
    {
        // Ligand atoms
        foreach (OEAtomBase latom in ligand.GetAtoms(new OEIsHeavy()))
        {
            OEResidue res = OEChem.OEAtomGetResidue(latom);
            double bfactor = res.GetBFactor();
            minbfactor = Math.Min(minbfactor, bfactor);
```

(continues on next page)

(continued from previous page)

```
            maxbfactor = Math.Max(maxbfactor, bfactor);
        }

        // Protein atoms close to ligand atoms
        OENearestNbrs nn = new OENearestNbrs(protein, maxdistance);
        foreach (OEAtomBase latom in ligand.GetAtoms(new OEIsHeavy()))
        {
            foreach (OENbrs neigh in nn.GetNbrs(latom))
            {
                OEAtomBase ratom = neigh.GetBgn();
                OEResidue res = OEChem.OEAtomGetResidue(ratom);
                if (ConsiderResidueAtom(ratom, res))

                {
                    double bfactor = res.GetBFactor();
                    minbfactor = Math.Min(minbfactor, bfactor);
                    maxbfactor = Math.Max(maxbfactor, bfactor);
                }
            }
        }
    }

    static void SetAverageBFactorOfNearbyProteinAtoms(OEMolBase ligand, OEMolBase␣
↪protein,
            uint itag, double maxdistance)
    {

        OENearestNbrs nn = new OENearestNbrs(protein, maxdistance);
        foreach (OEAtomBase latom in ligand.GetAtoms(new OEIsHeavy()))
        {
            double sumbfactor = 0.0;
            OEAtomBaseVector neighs = new OEAtomBaseVector();
            foreach (OENbrs neigh in nn.GetNbrs(latom))
            {
                OEAtomBase ratom = neigh.GetBgn();
                OEResidue res = OEChem.OEAtomGetResidue(ratom);
                if (ConsiderResidueAtom(ratom, res))

                {
                    double bfactor = res.GetBFactor();
                    sumbfactor += bfactor;
                    neighs.Add(ratom);
                }
            }
            double avgbfactor = 0.0;
            if (neighs.Count != 0)
            {
                avgbfactor = sumbfactor / neighs.Count;
            }
            latom.SetDoubleData(itag, avgbfactor);
        }
    }
```

(continues on next page)

(continued from previous page)

```
    private class ColorLigandAtomByBFactor : OEAtomGlyphBase
    {
        private OELinearColorGradient colorg;

        private ColorLigandAtomByBFactor()
        {}
        public ColorLigandAtomByBFactor(OELinearColorGradient cg)
        {
            colorg = new OELinearColorGradient(cg);
        }

        public override bool RenderGlyph(OE2DMolDisplay disp, OEAtomBase atom)
        {

            OE2DAtomDisplay adisp = disp.GetAtomDisplay(atom);
            if (adisp == null || !adisp.IsVisible())
            {
                return false;
            }

            OEResidue res = OEChem.OEAtomGetResidue(atom);
            double bfactor = res.GetBFactor();
            OEColor color = new OEColor(colorg.GetColorAt(bfactor));
            OEPen pen = new OEPen(color, color, OEFill.On, 1.0);
            double radius = disp.GetScale()/3.0;

            OEImage layer = disp.GetLayer(OELayerPosition.Below);
            OEGrapheme.OEDrawCircle(layer, OECircleStyle.Default, adisp.GetCoords(),␣
↪radius, pen);

            return true;
        }

        public override OEAtomGlyphBase CreateCopy()
        {
            ColorLigandAtomByBFactor copy = new ColorLigandAtomByBFactor(colorg);
            copy.ReleaseOwnership();
            return copy;
        }
    }

    private class BFactorArcFxn : OESurfaceArcFxnBase
    {
        private OELinearColorGradient colorg;
        private uint itag;

        private BFactorArcFxn()
        {}

        public BFactorArcFxn(OELinearColorGradient cg, uint i)
        {
```

(continues on next page)

(continued from previous page)

```
            colorg = new OELinearColorGradient(cg);
            itag = i;
        }

        public override bool Call(OEImageBase image, OESurfaceArc arc)
        {
            OE2DAtomDisplay adisp = arc.GetAtomDisplay();
            if (adisp == null || !adisp.IsVisible())
            {
                return false;
            }

            OEAtomBase atom = adisp.GetAtom();
            if (atom == null)
            {
                return false;
            }

            double avgresiduebfactor = atom.GetDoubleData(itag);
            if (avgresiduebfactor == 0.0)
            {
                return true;
            }

            OEColor color = new OEColor(colorg.GetColorAt(avgresiduebfactor));
            OEPen pen = new OEPen(color, color, OEFill.Off, 5.0);
            OE2DPoint center = arc.GetCenter();
            double bAngle = arc.GetBgnAngle();
            double eAngle = arc.GetEndAngle();
            double radius = arc.GetRadius();
            OEGrapheme.OEDrawDefaultSurfaceArc(image, center, bAngle, eAngle, radius,␣
↪pen);

            return true;
        }

        public override OESurfaceArcFxnBase CreateCopy()
        {
            BFactorArcFxn copy = new BFactorArcFxn(colorg, itag);
            copy.ReleaseOwnership();
            return copy;
        }
    }

    private static string interfaceData = @"
!BRIEF [-complex] <input> [-out] <output image>

!CATEGORY ""input/output options :""

  !PARAMETER -complex
    !ALIAS -c
    !TYPE string
```

(continues on next page)

(continued from previous page)

```
    !KEYLESS 1
    !REQUIRED true
    !VISIBILITY simple
    !BRIEF Input filename of the protein complex
  !END

  !PARAMETER -out
    !ALIAS -o
    !TYPE string
    !REQUIRED true
    !KEYLESS 2
    !VISIBILITY simple
    !BRIEF Output filename
  !END

!END
";

}
```

See also

- `OESplitMolComplex` function in **OEChem TK**
- OENearestNbrs class in **OEChem TK**
- *`OEPrepareDepictionFrom3D`* function
- *OESurfaceArcFxnBase* class and *`OEDraw2DSurface`* function
- *OEAtomGlyphBase* class and *`OEAddGlyph`* function

Examples

```
prompt> BFactor2Img -complex 1MEH.pdb -ligandname MOA -out MOA.png
```

will generate the following output:

```
Average BFactor of the complex = +32.721
Minimum BFactor of the ligand and its environment = +21.500
Maximum BFactor of the ligand and its environment = +72.060
```

and the image shown in *Figure: Example of depicting the BFactor of MOA and its environment*.

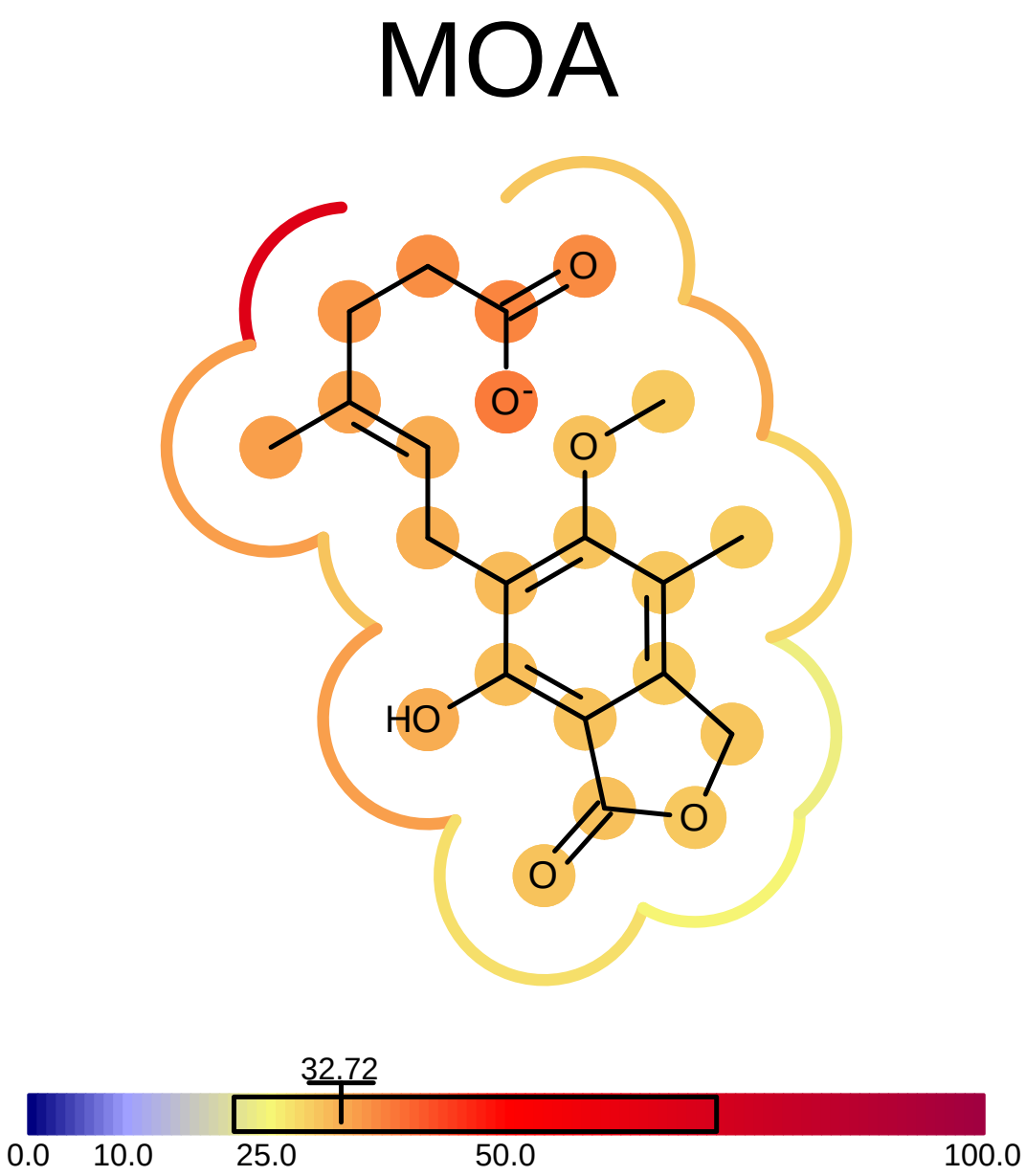


Fig. 1: **Example of depicting the BFactor of MOA and its environment**

```
prompt> BFactor2Img -complex 1LEE.pdb -ligandname R36 -out MOA.R36
```

will generate the following output:

```
Average BFactor of the complex = +23.754
Minimum BFactor of the ligand and its environment = +6.040
Maximum BFactor of the ligand and its environment = +32.050
```

and the image shown in *Figure: Example of depicting the BFactor of R36 and its environment*.

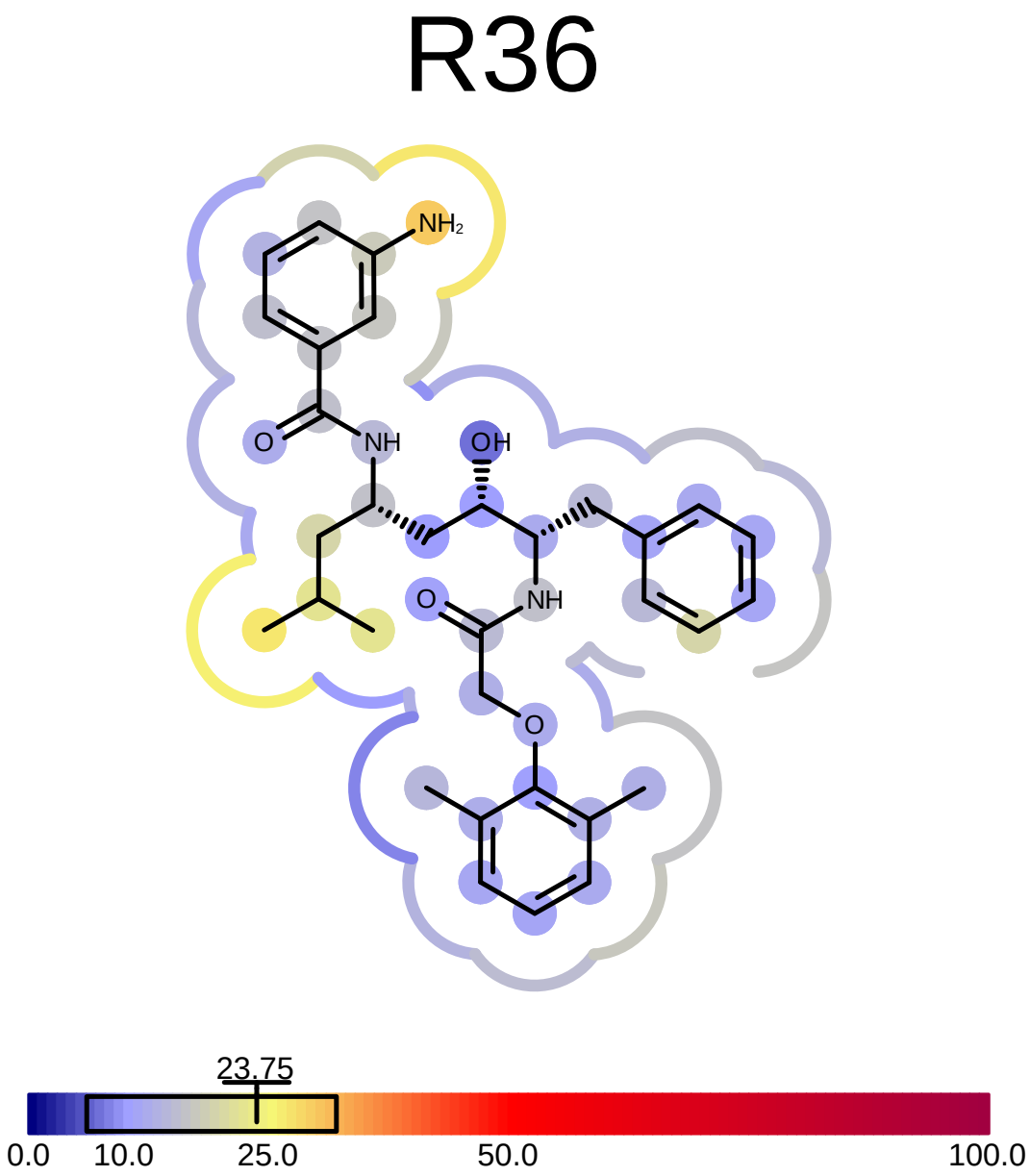


Fig. 2: **Example of depicting the BFactor of R36 and its environment**

3.1.2 Depicting Active Site Interactions

A program that depicts receptor-ligand interactions.

See also

- Visualizing Protein-Ligand Interactions OpenEye Python Cookbook example

Command Line Interface

A description of the command line interface can be obtained by executing the program with the –help argument.

```
prompt> Complex2Img --help
```

will generate the following output:

```
[-complex] <input> [-out] <output image>

Simple parameter list
    -height : Height of output image
    -width : Width of output image

    SplitMolComplex options :
      -ligandname : Ligand name

    input/output options
      -complex : Input filename of the protein complex
      -out : Output filename

    molecule display options
      -aromstyle : Aromatic ring display style
```

Code

```
/*
(C) 2022 Cadence Design Systems, Inc. (Cadence)
All rights reserved.
TERMS FOR USE OF SAMPLE CODE The software below ("Sample Code") is
provided to current licensees or subscribers of Cadence products or
SaaS offerings (each a "Customer").
Customer is hereby permitted to use, copy, and modify the Sample Code,
subject to these terms. Cadence claims no rights to Customer's
modifications. Modification of Sample Code is at Customer's sole and
exclusive risk. Sample Code may require Customer to have a then
current license or subscription to the applicable Cadence offering.
THE SAMPLE CODE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED.  OPENEYE DISCLAIMS ALL WARRANTIES, INCLUDING, BUT
NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE AND NONINFRINGEMENT. In no event shall Cadence be
liable for any damages or liability in connection with the Sample Code
or its use.
*/
/**********************************************************************
 * Depicts the interactions of an active site
 **********************************************************************/
using System;
```

(continues on next page)

(continued from previous page)

```
using OpenEye.OEBio;
using OpenEye.OEChem;
using OpenEye.OEDocking;
using OpenEye.OEDepict;
using OpenEye.OEGrapheme;

public class Complex2Img
{
    public static int Main(string[] args)
    {
        OEInterface itf = new OEInterface(interfaceData);

        if (!OEChem.OEParseCommandLine(itf, args, "Complex2Img"))
        {
            OEChem.OEThrow.Fatal("Unable to interpret command line!");
        }

        OEDepict.OEConfigureImageWidth (itf, 900.0);
        OEDepict.OEConfigureImageHeight(itf, 600.0);
        OEDepict.OEConfigure2DMolDisplayOptions(itf, OE2DMolDisplaySetup.AromaticStyle);
        OEBio.OEConfigureSplitMolComplexOptions(itf, OESplitMolComplexSetup.LigName);

        string iname = itf.GetString("-complex");
        string oname = itf.GetString("-out");

        oemolistream ifs = new oemolistream();
        if (!ifs.open(iname))
        {
            OEChem.OEThrow.Fatal("Cannot open input file!");
        }

        string ext = OEChem.OEGetFileExtension(oname);
        if (!OEDepict.OEIsRegisteredImageFile(ext))
        {
            OEChem.OEThrow.Fatal("Unknown image type!");
        }

        oeofstream ofs = new oeofstream();
        if (!ofs.open(oname))
        {
            OEChem.OEThrow.Fatal("Cannot open output file!");
        }
        ofs.close();

        OEGraphMol complexmol = new OEGraphMol();
        if (!OEChem.OEReadMolecule(ifs, complexmol))
        {
            OEChem.OEThrow.Fatal("Unable to read molecule from " + iname);
        }
        ifs.close();

        if (!OEChem.OEHasResidues(complexmol))
```

(continues on next page)

(continued from previous page)

```
        {
            OEChem.OEPerceiveResidues(complexmol, OEPreserveResInfo.All);
        }

        // Separate ligand and protein

        OESplitMolComplexOptions sopts = new OESplitMolComplexOptions();
        OEBio.OESetupSplitMolComplexOptions(sopts, itf);

        OEGraphMol ligand = new OEGraphMol();
        OEGraphMol protein = new OEGraphMol();
        OEGraphMol water = new OEGraphMol();
        OEGraphMol other = new OEGraphMol();

        OEUnaryRoleSetPred pfilter = sopts.GetProteinFilter();
        OEUnaryRoleSetPred wfilter = sopts.GetWaterFilter();
        sopts.SetProteinFilter(new OEOrRoleSet(pfilter, wfilter));
        sopts.SetWaterFilter(OEBio.OEMolComplexFilterFactory(OEMolComplexFilterCategory.
↪Nothing));

        OEBio.OESplitMolComplex(ligand, protein, water, other, complexmol, sopts);

        if (ligand.NumAtoms() == 0)
        {
            OEChem.OEThrow.Fatal("Cannot separate complex!");
        }

        // Perceive interactions
        OEInteractionHintContainer asite = new OEInteractionHintContainer(protein,␣
↪ligand);
        if (!asite.IsValid())
        {
            OEChem.OEThrow.Fatal("Cannot initialize active site!");
        }
        asite.SetTitle(ligand.GetTitle());

        OEBio.OEPerceiveInteractionHints(asite);
        OEGrapheme.OEPrepareActiveSiteDepiction(asite);

        // Depict active site with interactions
        double imagewidth = OEDepict.OEGetImageWidth(itf);
        double imageheight = OEDepict.OEGetImageHeight(itf);
        OEImage image = new OEImage(imagewidth, imageheight);

        OEImageFrame cframe = new OEImageFrame(image, imagewidth * 0.80, imageheight,
                                               new OE2DPoint(0.0, 0.0));
        OEImageFrame lframe = new OEImageFrame(image, imagewidth * 0.20, imageheight,
                                               new OE2DPoint(imagewidth * 0.80, 0.0));

        OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(
            cframe.GetWidth(), cframe.GetHeight());
        OEDepict.OESetup2DMolDisplayOptions(opts, itf);
```

(continues on next page)

(continued from previous page)

```
        OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);
        OEGrapheme.OERenderActiveSite(cframe, adisp);

        OE2DActiveSiteLegendDisplayOptions lopts = new␣
↪OE2DActiveSiteLegendDisplayOptions(10, 1);
        OEGrapheme.OEDrawActiveSiteLegend(lframe, adisp, lopts);

        OEDepict.OEWriteImage(oname, image);

        return 0;
    }

    static void SeparateLigandAndReceptor(OEMolBase receptor, OEMolBase ligand,
                                          OEMolBase complexmol, String residuename)
    {

        OEChem.OEPerceiveResidues(complexmol, OEPreserveResInfo.All);
        OEHierView hview = new OEHierView(complexmol);

        foreach (OEHierResidue res in hview.GetResidues())
        {
            if (res.GetOEResidue().GetName().Equals(residuename))
            {
                char chainid = res.GetOEResidue().GetChainID();
                int resnum = res.GetOEResidue().GetResidueNumber();
                OEChem.OEThrow.Info("Locating residue " + chainid + residuename
                                    + resnum.ToString());
                OEHierResidue ligandres = hview.GetResidue(chainid,
                                          residuename, resnum);
                if (ligandres.GetOEResidue().GetName().Length == 0)
                {
                    OEChem.OEThrow.Fatal("Failed to find residue!");
                }

                OEIsAtomMember ligandatoms = new OEIsAtomMember(
                    ligandres.GetAtoms());
                OEChem.OESubsetMol(ligand, complexmol, ligandatoms, true);
                OEChem.OESubsetMol(receptor, complexmol, new OENotAtom(
                                   ligandatoms), true);
                break;
            }
        }
    }

    private static string interfaceData = @"
!BRIEF [-complex] <input> [-out] <output image>

!CATEGORY ""input/output options :""

  !PARAMETER -complex
    !ALIAS -c
```

(continues on next page)

(continued from previous page)

```
    !TYPE string
    !KEYLESS 1
    !REQUIRED true
    !VISIBILITY simple
    !BRIEF Input filename of the protein complex
  !END

  !PARAMETER -out
    !ALIAS -o
    !TYPE string
    !REQUIRED true
    !KEYLESS 2
    !VISIBILITY simple
    !BRIEF Output filename
  !END

!END
";
}
```

See also

- `OESplitMolComplex` function in **OEChem TK**
- OEInteractionHintContainer class in **OEChem TK**
- `OEPerceiveInteractionHints` function in **OEChem TK**
- *`OEPrepareActiveSiteDepiction`* function
- *OE2DActiveSiteDisplayOptions* class
- *OE2DActiveSiteDisplay* class and *`OERenderActiveSite`* function
- *`OEDrawActiveSiteLegend`* function

Examples

```
prompt> Complex2Img complex2img.py -complex pdb1br6.ent -out 1br6-PT1.png
```

will generate the image shown in *Figure: Example of depicting the receptor-ligand interaction (PDB: 1br6)*.

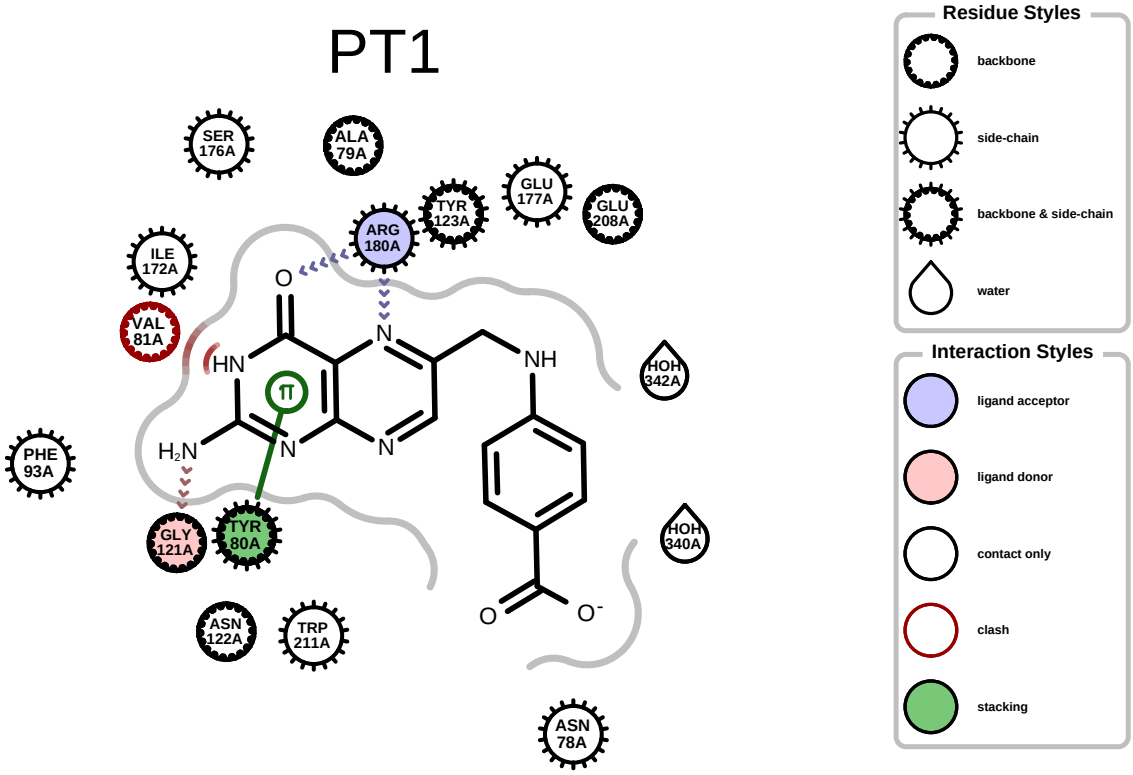


Fig. 3: **Example of depicting the receptor-ligand interactions (PDB: 1br6)**

```
prompt> Complex2Img.py -complex pdb1eoc.ent -out 1eoc-4NC.png
```

will generate the image shown in *Figure: Example of depicting the receptor-ligand interaction (PDB: 1eoc)*.

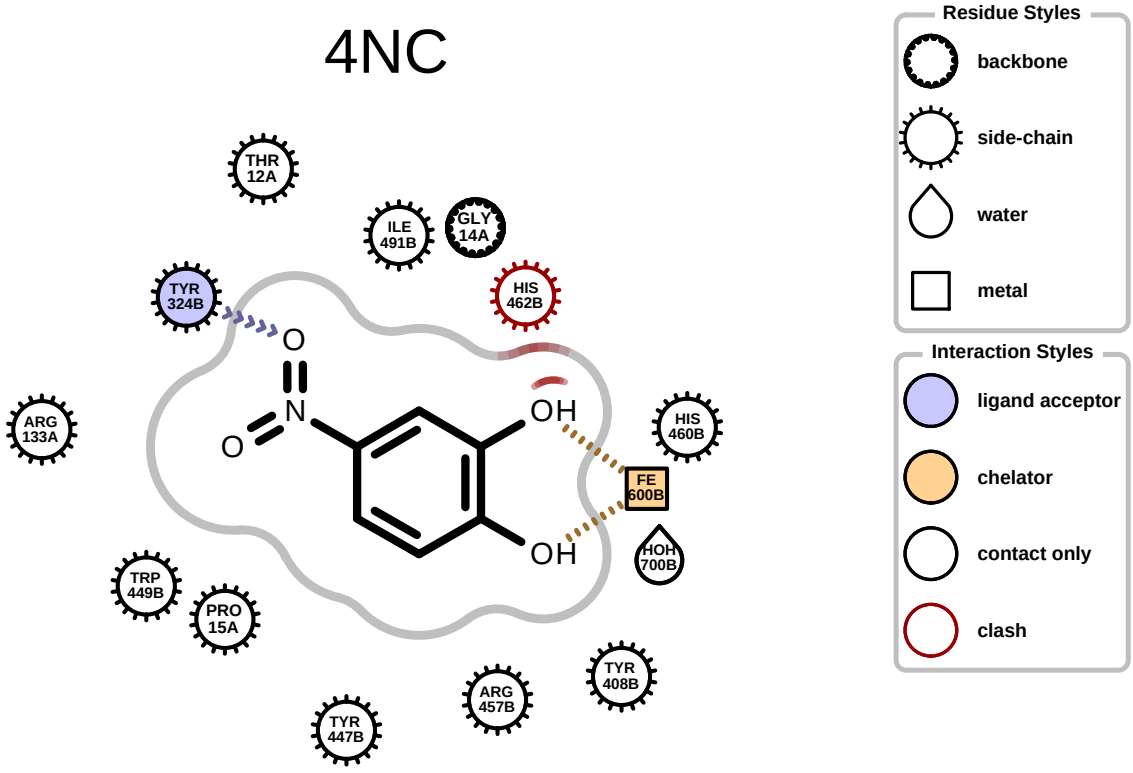


Fig. 4: **Example of depicting the receptor-ligand interactions (PDB: 1eoc)**

3.1.3 Depicting Shape and Color Atom Overlaps

A program that depicts the shape and the color overlap between a 3D reference structure and a sets of pre-aligned 3D fit molecules. It is implemented to visualize the output of ROCS but can be used to visualize the shape and color similarities of any molecules that have been pre-aligned with other applications.

> **Note**
>
> **OEShape TK** license is **not** required to run this example.

Command Line Interface

A description of the command line interface can be obtained by executing the program with the –help argument.

```
prompt> ShapeOverlap2PDF --help
```

will generate the following output:

```
[-in] <input> [-out] <output pdf>

 Simple parameter list
     input/output options:
       -in : Input molecule filename
```

(continues on next page)

(continued from previous page)

```
      -out : Output image filename

    general options:
      -maxhits : Maximum number of hits depicted
```

Code

```
/*
(C) 2022 Cadence Design Systems, Inc. (Cadence)
All rights reserved.
TERMS FOR USE OF SAMPLE CODE The software below ("Sample Code") is
provided to current licensees or subscribers of Cadence products or
SaaS offerings (each a "Customer").
Customer is hereby permitted to use, copy, and modify the Sample Code,
subject to these terms. Cadence claims no rights to Customer's
modifications. Modification of Sample Code is at Customer's sole and
exclusive risk. Sample Code may require Customer to have a then
current license or subscription to the applicable Cadence offering.
THE SAMPLE CODE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED.  OPENEYE DISCLAIMS ALL WARRANTIES, INCLUDING, BUT
NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE AND NONINFRINGEMENT. In no event shall Cadence be
liable for any damages or liability in connection with the Sample Code
or its use.
*/
/*****************************************************************************
 * Depicts shape and color overlap between a 3D reference structure and
 * a sets of 3D fit molecules. The molecules have to be pre-aligned.
 * The first molecule is expected be the reference.
 *****************************************************************************/
using System;
using OpenEye.OEChem;
using OpenEye.OEDepict;
using OpenEye.OEGrapheme;
using OpenEye.OEShape;

public class ShapeOverlap2PDF
{
    public static int Main(string[] args)
    {
        OEInterface itf = new OEInterface(interfaceData);

        if (!OEChem.OEParseCommandLine(itf, args, "ShapeOverlap2PDF"))
        {
            OEChem.OEThrow.Fatal("Unable to interpret command line");
        }

        string iname = itf.GetString("-in");
        string oname = itf.GetString("-out");
        int maxhits  = itf.GetInt("-maxhits");
```

(continues on next page)

(continued from previous page)

```
        string ext = OEChem.OEGetFileExtension(oname);
        if (!OEDepict.OEIsRegisteredMultiPageImageFile(ext))
        {
            OEChem.OEThrow.Fatal("Unknown multipage image type!");
        }

        oemolistream ifs = new  oemolistream();
        if (!ifs.open(iname))
        {
            OEChem.OEThrow.Fatal("Cannot open fit molecule file!");
        }

        OEGraphMol refmol = new OEGraphMol();
        if (!OEChem.OEReadMolecule(ifs, refmol))
        {
            OEChem.OEThrow.Fatal("Cannot read reference molecule!");
        }

        OEReportOptions ropts = new OEReportOptions(3, 1);
        ropts.SetHeaderHeight(40.0);
        ropts.SetFooterHeight(20.0);
        OEReport report = new OEReport(ropts);

        OEColorForceField cff = new OEColorForceField();
        cff.Init(OEColorFFType.ImplicitMillsDean);
        OEColorForceFieldDisplay cffdisplay = new OEColorForceFieldDisplay(cff);


        OEShapeQueryDisplayOptions   qopts = getShapeQueryDisplayOptions();
        OEShapeOverlapDisplayOptions sopts = getShapeOverlapDisplayOptions();
        OEColorOverlapDisplayOptions copts = getColorOverlapDisplayOptions();

        OEShapeQueryDisplay refdisp = new OEShapeQueryDisplay(refmol, cff, qopts);

        OEDots dots = new OEDots(100, 10, "shape overlaps");

        OEGraphMol fitmol = new OEGraphMol();
        while (OEChem.OEReadMolecule(ifs, fitmol))
        {
            dots.Update();
            if (maxhits > 0 && dots.GetCounts() >= maxhits)
            {
                break;
            }
            OEImageBase maincell = report.NewCell();
            OEImageGrid grid = new OEImageGrid(maincell, 1, 3);
            grid.SetMargins(5.0);
            grid.SetCellGap(5.0);

            // SD DATA + SCORE GRAPH + QUERY
            OEImageBase cell = grid.GetCell(1, 1);
            double cellw = cell.GetWidth();
```

(continues on next page)

(continued from previous page)

```
            double cellh = cell.GetHeight();

            OEFont tfont = new OEFont(OEFontFamily.Default, OEFontStyle.Bold, 10,
                                      OEAlignment.Left, OEChem.OEBlack);
            OE2DPoint tpos = new OE2DPoint(10.0, 10.0);
            cell.DrawText(tpos, fitmol.GetTitle(), tfont, cell.GetWidth());

            OEImageFrame rframe = new OEImageFrame(cell, cellw, cellh * 0.35, new␣
↪OE2DPoint(0.0, cellh * 0.10));
            OEImageFrame mframe = new OEImageFrame(cell, cellw, cellh * 0.50, new␣
↪OE2DPoint(0.0, cellh * 0.50));

            renderScoreRadial(rframe, fitmol);
            OEGrapheme.OERenderShapeQuery(mframe, refdisp);

            OEFont font = new OEFont(OEFontFamily.Default, OEFontStyle.Bold, 8,
                                     OEAlignment.Center, OEChem.OEGrey);
            OE2DPoint pos = new OE2DPoint(20.0, 10.0);
            mframe.DrawText(pos, "query", font);
            OEDepict.OEDrawCurvedBorder(mframe, OEDepict.OELightGreyPen, 10.0);

            OEShapeOverlapDisplay odisp = new OEShapeOverlapDisplay(refdisp, fitmol,␣
↪sopts, copts);

            // SHAPE OVERLAP
            cell = grid.GetCell(1, 2);
            OEGrapheme.OERenderShapeOverlap(cell, odisp);
            renderScore(cell, fitmol, "ROCS_ShapeTanimoto", "Shape Tanimoto");

            // COLOR OVERLAP
            cell = grid.GetCell(1, 3);
            OEGrapheme.OERenderColorOverlap(cell, odisp);
            renderScore(cell, fitmol, "ROCS_ColorTanimoto", "Color Tanimoto");

            OEDepict.OEDrawCurvedBorder(maincell, OEDepict.OELightGreyPen, 10.0);
        }
        dots.Total();

        OEColorForceFieldLegendDisplayOptions cffopts = new␣
↪OEColorForceFieldLegendDisplayOptions(1, 6);
        foreach (OEImageBase header in report.GetHeaders())
        {
            OEGrapheme.OEDrawColorForceFieldLegend(header, cffdisplay, cffopts);
            OEDepict.OEDrawCurvedBorder(header, OEDepict.OELightGreyPen, 10.0);
        }

        OEFont pfont = new OEFont(OEFontFamily.Default, OEFontStyle.Default, 12,
                                  OEAlignment.Center, OEChem.OEBlack);
        int idx = 0;
        foreach (OEImageBase footer in report.GetFooters())
        {
            idx = idx + 1;
```

(continues on next page)

(continued from previous page)

```
            string text = "-" + idx.ToString() + "-";
            OEDepict.OEDrawTextToCenter(footer, text, pfont);
        }

        OEDepict.OEWriteReport(oname, report);

        return 0;
    }

    static OEShapeQueryDisplayOptions getShapeQueryDisplayOptions()
    {
        OEShapeQueryDisplayOptions qopts = new OEShapeQueryDisplayOptions();
        qopts.SetTitleLocation(OETitleLocation.Hidden);
        qopts.SetAtomLabelFontScale(1.5);
        OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack, OEFill.Off, 1.5);
        qopts.SetDefaultBondPen(pen);
        qopts.SetSurfaceArcFxn(new OEDefaultArcFxn(OEDepict.OELightGreyPen));
        return qopts;
    }

    static OEShapeOverlapDisplayOptions getShapeOverlapDisplayOptions()
    {
        OEShapeOverlapDisplayOptions sopts = new OEShapeOverlapDisplayOptions();
        sopts.SetTitleLocation(OETitleLocation.Hidden);
        sopts.SetAtomLabelFontScale(1.5);
        OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack, OEFill.Off, 1.5);
        sopts.SetDefaultBondPen(pen);
        OEPen arcpen = new OEPen(OEChem.OEGrey, OEChem.OEGrey, OEFill.Off, 1.0, 0x1111);
        sopts.SetQuerySurfaceArcFxn(new OEDefaultArcFxn(arcpen));
        sopts.SetOverlapColor(new OEColor(110, 110, 190));
        return sopts;
    }

    static OEColorOverlapDisplayOptions getColorOverlapDisplayOptions()
    {
        OEColorOverlapDisplayOptions copts = new OEColorOverlapDisplayOptions();
        copts.SetTitleLocation(OETitleLocation.Hidden);
        copts.SetAtomLabelFontScale(1.5);
        OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack, OEFill.Off, 1.5);
        copts.SetDefaultBondPen(pen);
        OEPen arcpen = new OEPen(OEChem.OEGrey, OEChem.OEGrey, OEFill.Off, 1.0, 0x1111);
        copts.SetQuerySurfaceArcFxn(new OEDefaultArcFxn(arcpen));
        return copts;
    }

    static double getScore(OEMolBase mol, string sdtag)
    {
        if (OEChem.OEHasSDData(mol, sdtag))
        {
            string scorestr = OEChem.OEGetSDData(mol, sdtag);
            double score = System.Convert.ToDouble(scorestr);
            return score;
```

(continues on next page)

(continued from previous page)

```
        }
        return 0.0;
    }

    static void renderScoreRadial(OEImageBase image, OEMolBase mol)
    {
        double sscore = Math.Max(Math.Min(getScore(mol, "ROCS_ShapeTanimoto"), 1.0), 0.
↪00);
        double cscore = Math.Max(Math.Min(getScore(mol, "ROCS_ColorTanimoto"), 1.0), 0.
↪00);

        if (sscore > 0.0 || cscore > 0.0)
        {
            OEDoubleVector scores = new OEDoubleVector();
            scores.Add(sscore);
            scores.Add(cscore);
            OEGrapheme.OEDrawROCSScores(image, scores);
        }
    }
    static void renderScore(OEImageBase image, OEMolBase mol,
                            string sdtag, string label)
    {
        double score = getScore(mol, sdtag);
        if (score == 0.0)
        {
            return;
        }

        double w = image.GetWidth();
        double h = image.GetHeight();
        OEImageFrame frame = new OEImageFrame(image, w, h * 0.10, new OE2DPoint(0.0, h *␣
↪0.90));
        OEFont font = new OEFont(OEFontFamily.Default, OEFontStyle.Default, 9,
                                 OEAlignment.Center, OEChem.OEBlack);

        string text = label +  " = " + System.Convert.ToString(score);
        OEDepict.OEDrawTextToCenter(frame, text, font);
    }


    private static string interfaceData = @"
# ShapeOverlap2PDF interface file
!BRIEF [-in] <input> [-out] <output pdf>

!CATEGORY ""input/output options:"" 0

  !PARAMETER -in
    !ALIAS -i
    !TYPE string
    !REQUIRED true
    !KEYLESS 1
    !VISIBILITY simple
```

(continues on next page)

(continued from previous page)

```
    !BRIEF Input molecule filename
    !DETAIL
         The first molecule in the file is expected to be the reference
         molecule
  !END

  !PARAMETER -out
    !ALIAS -o
    !TYPE string
    !REQUIRED true
    !KEYLESS 2
    !VISIBILITY simple
    !BRIEF Output image filename
  !END

!END

!CATEGORY ""general options:"" 1

  !PARAMETER -maxhits
    !ALIAS -mhits
    !TYPE int
    !REQUIRED false
    !DEFAULT 0
    !LEGAL_RANGE 0 500
    !VISIBILITY simple
    !BRIEF Maximum number of hits depicted
    !DETAIL
            The default of 0 means there is no limit.
  !END

!END
";
}
```

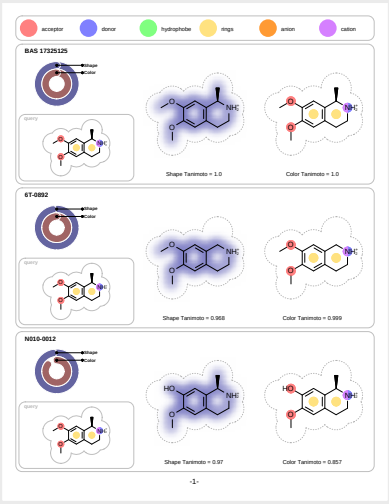
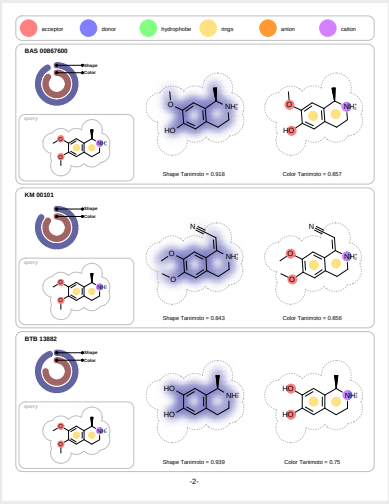
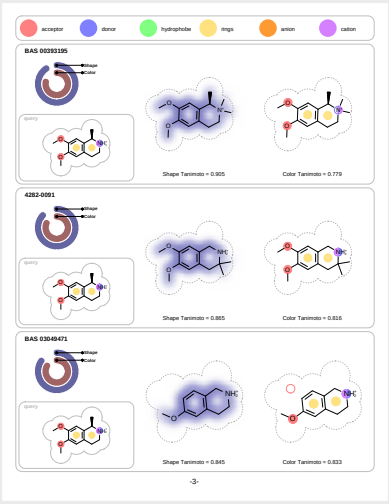
See also

- OEColorForceField class in the **OEShape TK**
- *OEShapeQueryDisplay* and *OEShapeQueryDisplayOptions* classes
- *OERenderShapeQuery* function
- *OEShapeOverlapDisplay*, *OEShapeOverlapDisplayOptions*, and *OEColorOverlapDisplayOptions* classes
- *OERenderShapeOverlap* function
- *OERenderColorOverlap* function
- *OEDrawColorForceFieldLegend* function
- *OEDrawROCSScores* function

Examples

```
prompt> ShapeOverlap2PDF -in rocs.sdf -out overlaps.pdf
```

Table 1: **Example of depicting the shape and color atom overlaps** (The pages are reduced here for visualization convenience)

page 1	page 2	page 3
		

3.2 Python Cookbook Examples

There are large number of code examples available in OpenEye Python Cookbook that use **OpenEye Toolkits** to solve a wide range of cheminformatics and molecular modeling problems.

- 2D Depiction chapter contains examples that illustrate how the **OEDepict TK** and **Grapheme TK** can be utilized to depict molecules and visualize properties calculated with other OpenEye toolkits.
- Visualizing 3D Information chapter contains examples that illustrate how the **Grapheme TK** can be utilized to project complex 3D information into the 2D molecular graph.
- Cheminformatics chapter contains examples that solve various cheminformatics problems such as similarity search, ring perception and manipulating molecular graphs.

- Peptide Informatics chapter contains examples that solve various peptide informatics problems such as generating custom monomer sets, parsing and generating HELM representations, and visualizing peptides.

OpenEye Python Cookbook examples using OEGrapheme TK

- Visualizing Torsional Angle Distribution
- Visualizing Electron Density
- Visualizing Molecular Dipole Moment
- Depicting Molecular Properties
- Visualizing Shape and Color Overlap

Protein-ligand visualization:

- Visualizing Protein-Ligand Interactions
- Visualizing Protein-Ligand Unpaired Interactions
- Visualizing Protein-Ligand B-Factor
- Visualizing Protein-Ligand B-Factor Map
- Visualizing Protein-Ligand Contacts
- Visualizing Protein-Ligand Maps
- Ramachandran Plot

Peptide visualization:

- Depict Peptide
- Depict Monomer

CHAPTER

FOUR

API

4.1 OEGrapheme Classes

4.1.1 OE2DActiveSiteDisplay

```
class OE2DActiveSiteDisplay
```

This class represents *OE2DActiveSiteDisplay* that stores the depiction information (such as 2D coordinates, representation styles etc.) of an active site.

An active site display can be rendered into an image using the *OEPrepareActiveSiteDepiction* and *OERenderActiveSite* functions.

Code Example

- *Depicting Active Site Interactions* example
- Visualizing Protein-Ligand Interactions **OpenEye Python Cookbook** recipe

Constructors

```
OE2DActiveSiteDisplay(const OEBio::OEInteractionHintContainer& asite,
                      const OE2DActiveSiteDisplayOptions& opts)
```

Initializes an *OE2DActiveSiteDisplay* object using the given display options.

asite
The active site for which display information is stored in the *OE2DActiveSiteDisplay* object.

opts
The *OE2DActiveSiteDisplayOptions* object that stores properties that determine the styles of the active site depiction.

Warning

- An *OE2DActiveSiteDisplay* object does not make a copy of the OEInteractionHintContainer object from which it is initialized but only stores its pointer. Therefore, the user responsibility is to not let the interaction container object go out of scope, before the corresponding *OE2DActiveSiteDisplay* object.

See also

- OEIsValidActiveSite function
- *OE2DActiveSiteDisplay.IsValid* method

GetDisplayedLigand

```
const OEChem::OEMolBase* GetDisplayedLigand() const
```

Returns a pointer to the display ligand of the active site.

GetDisplayedResidues

```
OESystem::OEIterBase<OEChem::OEResidue>* GetDisplayedResidues(unsigned int renderstyle =␣
↪OEActiveSiteRenderStyle::Default) const
```

Returns an iterator over the displayed residues of the given style.

renderstyle
This value has to be from the *OEActiveSiteRenderStyle* namespace.

GetHeight

```
double GetHeight() const
```

Returns the height of the displayed active site.

GetOptions

```
const OE2DActiveSiteDisplayOptions& GetOptions() const
```

Returns the options used to initialized the *OE2DActiveSiteDisplay* object.

GetWidth

```
double GetWidth() const
```

IsValid

```
bool IsValid() const
```

Returns whether the *OE2DActiveSiteDisplay* object was initialized successfully.

Hint

It is highly recommend to check whether an active site display is valid after initialization.

4.1.2 OE2DActiveSiteDisplayOptions

```
class OE2DActiveSiteDisplayOptions: public OEDepict::OE2DMolDisplayOptions
```

This class represents the *OE2DActiveSiteDisplayOptions* class that encapsulates properties that determine how an active site is depicted.

The following methods are publicly inherited from OE2DMolDisplayOptions:

operator=	GetDefaultBondPen	SetBondColorStyle
GetAromaticStyle	GetHeight	SetBondLineAtomLabelGapScale
GetAtomColor	GetHydrogenStyle	SetBondLineGapScale
GetAtomColorStyle	GetScale	SetBondPropLabelFont
GetAtomLabelFont	GetSuperAtomLabelFont	SetBondPropLabelFontScale
GetAtomLabelFontScale	GetSuperAtomStyle	SetBondPropertyFunctor
GetAtomPropLabelFont	GetTitleFont	SetBondStereoStyle
GetAtomPropLabelFontScale	GetTitleLocation	SetBondWidthScaling
GetAtomPropertyFunctor	GetWidth	SetDefaultBondPen
GetAtomStereoStyle	SetAromaticStyle	SetDimensions
GetBackgroundColor	SetAtomColor	SetHeight
GetBondColorStyle	SetAtomColorStyle	SetHydrogenStyle
GetBondLineAtomLabelGapScale	SetAtomLabelFont	SetScale
GetBondLineGapScale	SetAtomLabelFontScale	SetSuperAtomLabelFont
GetBondPropLabelFont	SetAtomPropLabelFont	SetSuperAtomStyle
GetBondPropLabelFontScale	SetAtomPropLabelFontScale	SetTitleFont
GetBondPropertyFunctor	SetAtomPropertyFunctor	SetTitleLocation
GetBondStereoStyle	SetAtomStereoStyle	SetWidth
GetBondWidthScaling	SetBackgroundColor	

Code Example

- *Depicting Active Site Interactions* example
- Visualizing Protein-Ligand Interactions **OpenEye Python Cookbook** recipe

Constructors

```
OE2DActiveSiteDisplayOptions(double width, double height)
```

Creates an *OE2DActiveSiteDisplayOptions* object with the specified dimension.

width*, *height
The width and height of the active site display. Both numbers have to be at least 200.0.

```
OE2DActiveSiteDisplayOptions(const OE2DActiveSiteDisplayOptions& rhs)
```

Copy constructor.

GetLigandAlignerFunctor

```
const OELigandAlignerBase& GetLigandAlignerFunctor() const
```

Returns the functor that is used to align ligands in the active site display. By default, the functor is set to *OENoOpLigandAligner*.

See also

- *OE2DActiveSiteDisplayOptions.SetLigandAlignerFunctor* method

GetOptimizeLigandOrientation

```
bool GetOptimizeLigandOrientation() const
```

Returns whether the ligand orientation is optimized after generated the 2D layout. The default value is *true*.

See also

- *OE2DActiveSiteDisplayOptions.SetOptimizeLigandOrientation* method

GetRenderInteractiveLegend

```
bool GetRenderInteractiveLegend() const
```

Returns whether and interactive legend is rendered only when mouse is hovered over the "Legend" button depicted on the right top corner on the active site depiction.

See also

- *OE2DActiveSiteDisplayOptions.SetRenderInteractiveLegend* method

GetResidueSVGMarkupFunctor

```
const OEResidueSVGMarkupBase& GetResidueSVGMarkupFunctor() const
```

Returns the functor that defines how the residues are marked in `svg` image. By default, residues are not marked (*OEResidueSVGNoMarkup*).

See also

- *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method

GetSVGMagnifyResidueInHover

```
double GetSVGMagnifyResidueInHover() const
```

Returns the scaling factor that is used to magnify residue glyphs when mouse is hover over them.

See also

- *OE2DActiveSiteDisplayOptions.SetSVGMagnifyResidueInHover* method

SetBackgroundColor

```
void SetBackgroundColor(const OESystem::OEColor &)
```

This is a no-op method. The background of the active site display is always `OEWhite`

SetDimensions

```
void SetDimensions(double width, double height, double scale)
```

This is a no-op method. The dimension of the active site display only be specified with the *OE2DActiveSiteDisplayOptions* constructor.

SetHeight

```
void SetHeight(double)
```

This is a no-op method. The height of the active site display can only be specified with the *OE2DActiveSiteDisplayOptions* constructor.

SetLigandAlignerFunctor

```
void SetLigandAlignerFunctor(const OELigandAlignerBase& func)
```

Sets the functor that can be used to custom align ligands in the active site display.

Hint

Turning off optimization of ligand orientation is also recommended for custom alignments with *OE2DActiveSiteDisplayOptions.SetOptimizeLigandOrientation* method.

See also

- *OELigandAlignerBase* abstract base class

SetOptimizeLigandOrientation

```
void SetOptimizeLigandOrientation(bool optimize)
```

Sets whether the ligand orientation is optimized after generated the 2D layout to maximize scaling for the depiction.

Hint

Its is recommended to disable this feature when using custom ligand alignment.

SetRenderInteractiveLegend

```
void SetRenderInteractiveLegend(bool render)
```

Sets whether and interactive legend is rendered only when mouse is hovered over the "Legend" button depicted on the right top corner on the active site depiction.

> **Note**
>
> This functionality is **only available** for `.svg` image format. In other image formats it has no effect.

SetResidueSVGMarkupFunctor

```
void SetResidueSVGMarkupFunctor(const OEResidueSVGMarkupBase& func)
```

Sets the functor that defines how the residues are marked in `svg` image. Drawing elements representing residues in `svg` image are grouped together in the following format in which the `<group id>` and `<class name>` strings are defined by the given functor:

```
<g id='<group id>' class='<class name>'>
..
list of drawing elements
..
</g>
```

> **Note**
>
> This setting has only effect when generating `.svg` images.

Example:

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(receptor, ligand);
OEBio.OEPerceiveInteractionHints(asite);
OEGrapheme.OEPrepareActiveSiteDepiction(asite);

OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(600.0, 400.0);
opts.SetResidueSVGMarkupFunctor(new OEResidueSVGStandardMarkup());

OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

OEGrapheme.OERenderActiveSite("SVGMarkResidues.svg", adisp);
```

> **See also**
>
> - *OEResidueSVGMarkupBase* abstract base class
> - *OEResidueSVGNoMarkup* class
> - *OEResidueSVGStandardMarkup* class
> - *OE2DActiveSiteDisplayOptions.GetResidueSVGMarkupFunctor* method

SetSVGMagnifyResidueInHover

```
void SetSVGMagnifyResidueInHover(double scale)
```

Sets the scaling factor that is used to magnify residue glyphs when mouse is hover over them.

scale
: This value has to be in a range of `[1.00, 3.00]`. The default value is 1.00 which has has no effect.

> **Hint**
>
> This is a very useful feature when generating an active site image with residue labels that are too small to read.

> **Note**
>
> This functionality is **only available** for `.svg` image format. In other image formats it has no effect.

SetScale

```
void SetScale(double)
```

This is a no-op method. The active site display can not be scaled.

SetSuperAtomStyle

```
void SetSuperAtomStyle(unsigned int)
```

This is a no-op method. The super atom style can not be used when depicting an active site.

SetWidth

```
void SetWidth(double)
```

This is a no-op method. The width of the active site display can only be specified with the *OE2DActiveSiteDisplayOptions* constructor.

4.1.3 OE2DActiveSiteLegendDisplayOptions

```
class OE2DActiveSiteLegendDisplayOptions
```

This class represents *OE2DActiveSiteLegendDisplayOptions* that encapsulates properties that determine how the legend of a color force field is depicted by the *OEDrawActiveSiteLegend* function.

> **See also**
>
> - *OE2DActiveSiteDisplay* class
> - *OEDrawActiveSiteLegend* function

Code Example

- *Depicting Active Site Interactions* example
- Visualizing Protein-Ligand Interactions **OpenEye Python Cookbook** recipe

Constructors

```
OE2DActiveSiteLegendDisplayOptions(unsigned int rows,
                                   unsigned int cols)
```

Creates an *OE2DActiveSiteLegendDisplayOptions* object with the specified number of rows and columns..

rows
The number of rows in the legend.

cols
The number of columns in the legend.

```
OE2DActiveSiteLegendDisplayOptions(const OE2DActiveSiteLegendDisplayOptions& rhs)
```

Copy constructor.

NumCols

```
unsigned int NumCols() const
```

Returns the number of columns of the legend.

NumRows

```
unsigned int NumRows() const
```

4.1.4 OE2DPropMap

```
class OE2DPropMap
```

The *OE2DPropMap* class maps atom properties into a 2D grid and then *blurs* the image by using a Gaussian function. The points of the 2D grid are then colored using the OELinearColorGradient class that interpolates colors between a specified range.

See also

- *Depicting Property Maps* chapter

Code Example

- Visualizing Molecular Dipole Moment **OpenEye Python Cookbook** recipe

The *OE2DPropMap* class can be customized with the following properties:

Property	Get method	Set method	Corresponding namespace/class
legend font	*GetLegendFont*	*SetLegendFont*	OEFont
legend font scale	*GetLegendFontScal*	*SetLegendFontScal*	
legend location	*GetLegendLocatior*	*SetLegendLocatior*	*OELegendLocation*
maximum value	*GetMaxValue*	*SetMaxValue*	
minimum value	*GetMinValue*	*SetMinValue*	
negative color	*GetNegativeColor*	*SetNegativeColor*	OEColor
positive color	*GetPositiveColor*	*SetPositiveColor*	OEColor
radius ratio	*GetRadiusRatio*	*SetRadiusRatio*	
grid resolution	*GetResolution*	*SetResolution*	

Constructors

```
OE2DPropMap(const OESystem::OEColor& backgroundColor = OESystem::OEColor(255, 255, 255))
```

Default constructor.

```
OE2DPropMap(const OE2DPropMap& rhs)
```

Copy constructor.

Render

```
bool Render(OEDepict::OE2DMolDisplay& disp,
            const std::string& tagname) const
```

Renders the property map into a molecule display.

disp
: The molecule display that holds the molecule for which the property map is calculated and on which the property map is drawn. The property map is drawn onto the *below* layer of the molecule display *i.e.*, it appears underneath the molecular diagram.

tagname
: The generic data identifier that is used to retrieve the properties being depicted on the property map.

```
bool Render(OEDepict::OEImageBase& image,
            const OEDepict::OE2DMolDisplay& disp,
            const std::string& tagname) const
```

Renders the property map into an image.

image
: The image on which the property map is drawn.

disp
: The molecule display that holds the molecule for which the property map is calculated.

tagname
: The generic data identifier that is used to retrieve the properties being depicted on the property map.

```
bool Render(OEDepict::OE2DMolDisplay& disp,
            const OEDepict::OE2DMolDisplay& refdisp,
            const std::string& tagname) const
```

Renders the property map calculated for one display, onto another display.

disp
The molecule display on which the property map is drawn. The property map is drawn onto the *below* layer of the molecule display *i.e.*, it appears underneath the molecular diagram.

refdisp
The molecule display that holds the molecule for which the property map is calculated.

tagname
The generic data identifier that used to retrieve the properties being depicted on the property map.

GetLegendFont

```
const OEDepict::OEFont& GetLegendFont() const
```

Returns the font that is used to render the title of the legend and the color stops of the linear color gradient.

See also

- *OE2DPropMap.SetLegendFont* method
- OEFont class

GetLegendFontScale

```
double GetLegendFontScale() const
```

Returns the multiplier that can be used to increase or decrease the size of the legend fonts.

See also

- *OE2DPropMap.SetLegendFontScale* method

GetLegendLocation

```
unsigned int GetLegendLocation() const
```

Returns the position of the legend. The return value is taken from the *OELegendLocation* namespace.

See also

- *OE2DPropMap.SetLegendLocation* method
- *OELegendLocation* namespace

GetMaxValue

```
double GetMaxValue() const
```

Returns the maximum value that is used to construct the color gradient of the property map.

See also

- *OE2DPropMap.SetMaxValue* method

GetMinValue

```
double GetMinValue() const
```

Returns the minimum value that is used to construct the color gradient of the property map.

See also

- *OE2DPropMap.SetMinValue* method

GetNegativeColor

```
const OESystem::OEColor& GetNegativeColor() const
```

Returns the color that is used to represent negative atom property values.

See also

- *OE2DPropMap.SetNegativeColor* method
- OEColor class

GetPositiveColor

```
const OESystem::OEColor& GetPositiveColor() const
```

Returns the color that is used to represent positive atom property values.

See also

- *OE2DPropMap.SetPositiveColor* method
- OEColor class

GetRadiusRatio

```
double GetRadiusRatio() const
```

Returns he ratio that is used to scale the Gaussian radius.

See also

- *OE2DPropMap.SetRadiusRatio* method

GetResolution

```
unsigned int GetResolution() const
```

Returns the resolution of the property grid.

See also

- *OE2DPropMap.SetResolution* method

SetLegendFont

```
void SetLegendFont(const OEDepict::OEFont& font)
```

Sets the font that is used to render the title of the legend and the color stops of the linear color gradient.

Note

The size of fonts of the legend also depends on the scaling factor used to fit the linear color gradient into the given dimensions and the multiplier set by the *OE2DPropMap.SetLegendFontScale* method.

Example (see *Figure: Example of using the SetLegendFont method*)

```
OE2DPropMap propmap = new OE2DPropMap();
OEFont f = new OEFont();
f.SetStyle(OEFontStyle.Bold);
f.SetColor(OEChem.OEDarkBlue);
propmap.SetLegendFont(f);
```

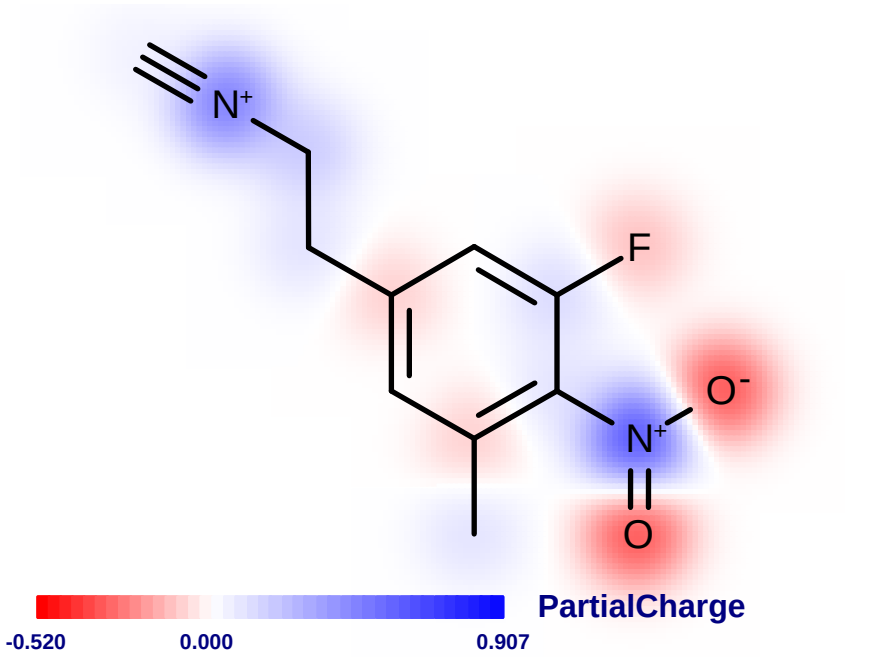


Fig. 1: **Example of using the SetLegendFont method**

See also

- *OE2DPropMap.GetLegendFont* method
- OEFont class

SetLegendFontScale

```
void SetLegendFontScale(double scale)
```

Sets the multiplier that can be used to increase or decrease the size of the legend fonts.

scale
This value has to be in the range of `[0.5, 2.0]`.

Example (see *Figure: Example of using the SetLegendFontScale method*)

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetLegendFontScale(1.5);
```

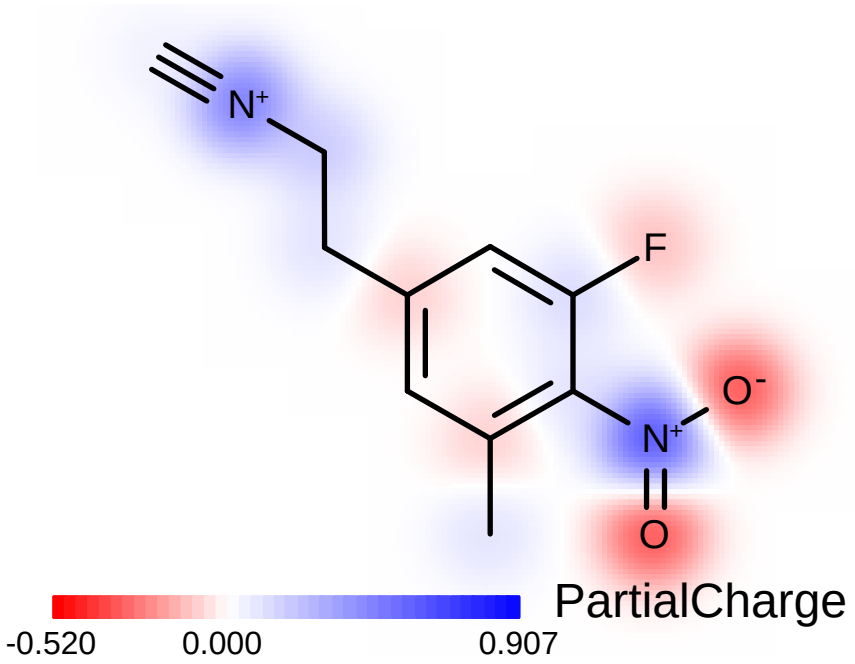


Fig. 2: **Example of using the SetLegendFontScale method**

See also

- *OE2DPropMap.GetLegendFontScale* method

SetLegendLocation

```
void SetLegendLocation(unsigned int loc)
```

Sets the position of the legend.

loc
: This value has to be from the *OELegendLocation* namespace.

Example (see *Figure: Example of using the SetLegendLocation method*)

Full listing.

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetLegendLocation(OELegendLocation.Left);
```

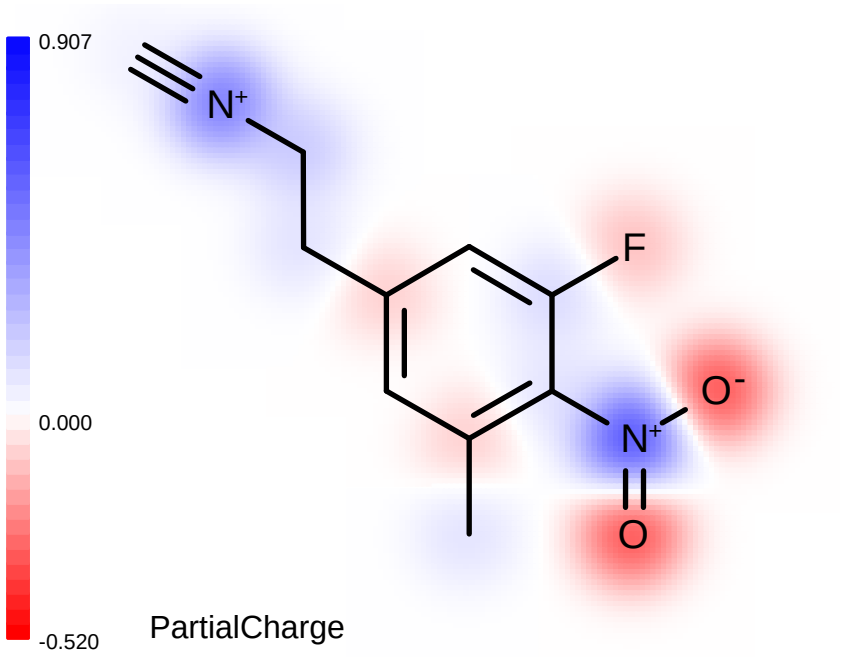


Fig. 3: **Example of using the SetLegendLocation method**

See also

- *OE2DPropMap.GetLegendLocation* method
- *OELegendLocation* namespace

SetMaxValue

```
void SetMaxValue(double maxv)
```

Sets the maximum value that is used to construct the color gradient of the property map. If no maximum value is specified prior to calling the *OE2DPropMap.Render* method, then the maximum value will be the largest atom property value.

Example (see *Figure: Example of using the SetMaxValue method*)

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetMaxValue(0.0);
```

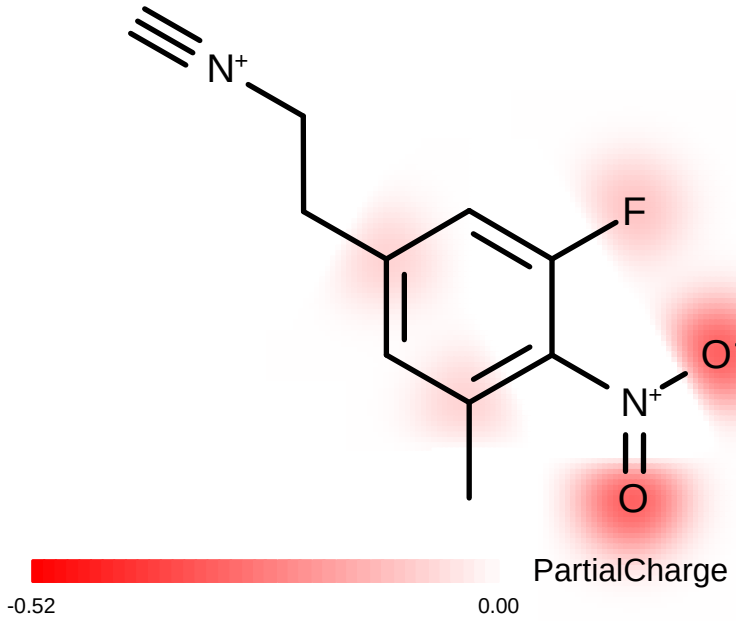


Fig. 4: **Example of using the SetMaxValue method**

See also

- *OE2DPropMap.GetMaxValue* method

SetMinValue

```
void SetMinValue(double minv)
```

Sets the minimum value that is used to construct the color gradient of the property map. If no minimum value is specified prior to calling the *OE2DPropMap.Render* method, then the minimum value will be the smallest atom property value.

Example (see *Figure: Example of using the SetMinValue method*)

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetMinValue(0.0);
```

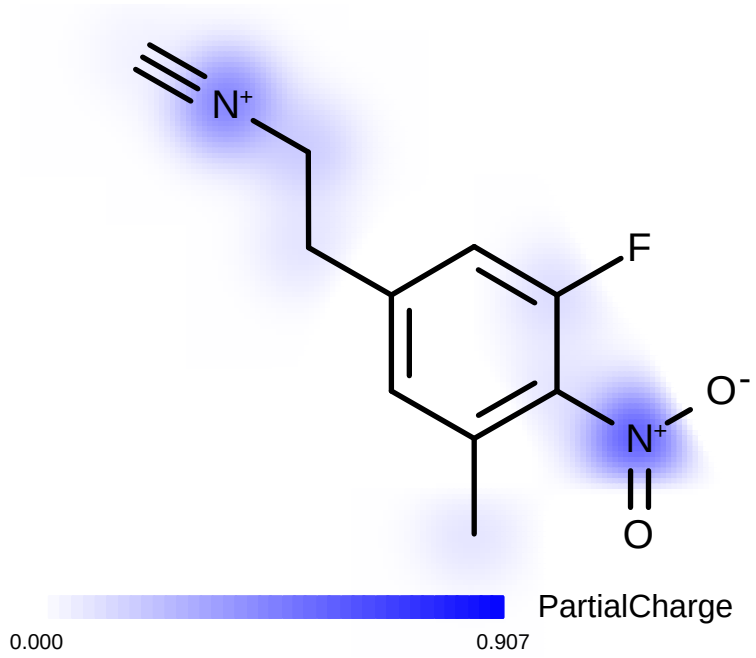


Fig. 5: **Example of using the SetMinValue method**

See also

- *OE2DPropMap.GetMinValue* method

SetNegativeColor

```
void SetNegativeColor(const OESystem::OEColor& color)
```

Sets the color that is used to represent negative atom property values.

Example (see *Figure: Example of using the SetNegativeColor method*)

Full listing.

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetNegativeColor(OEChem.OEDarkGreen);
```

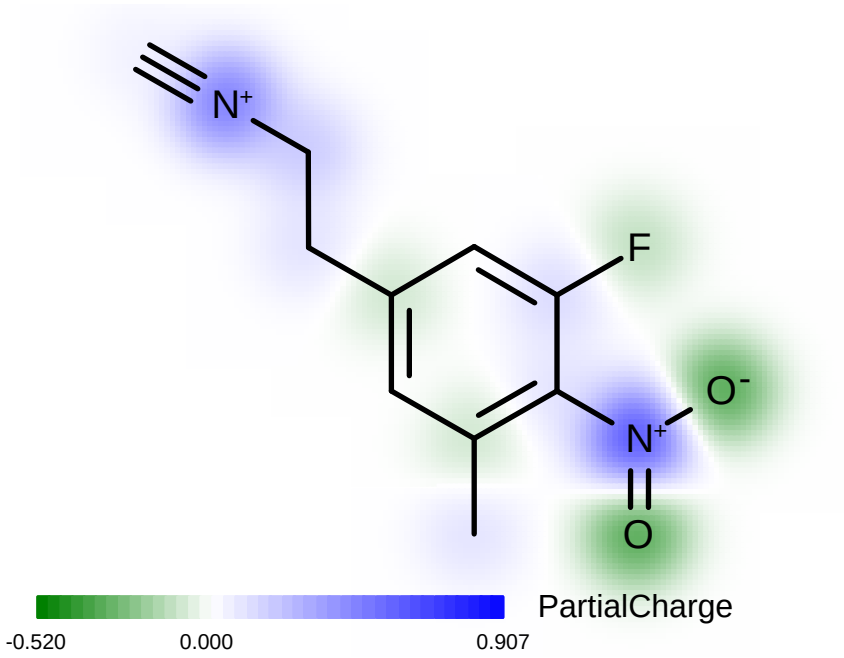


Fig. 6: **Example of using the SetNegativeColor method**

See also

- *OE2DPropMap.GetNegativeColor* method
- OEColor class

SetPositiveColor

```
void SetPositiveColor(const OESystem::OEColor& color)
```

Sets the color that is used to represent positive atom property values.

Example (see *Figure: Example of using the SetPositiveColor method*)

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetPositiveColor(OEChem.OEDarkGreen);
```

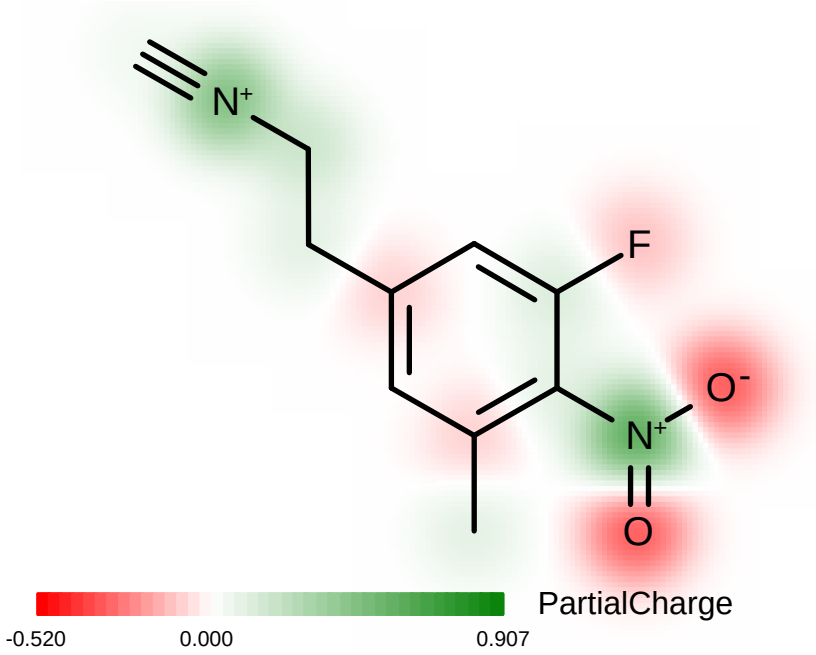


Fig. 7: **Example of using the SetPositiveColor method**

See also

- *OE2DPropMap.GetPositiveColor* method
- OEColor class

SetRadiusRatio

```
void SetRadiusRatio(double ratio)
```

Sets the ratio that is used to scale the Gaussian radius.

ratio
This value has to be in the range of [0.5, 2.0].

Example (see *Figure: Example of using the SetRadiusRatio method*)

Full listing.

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetRadiusRatio(0.75);
```

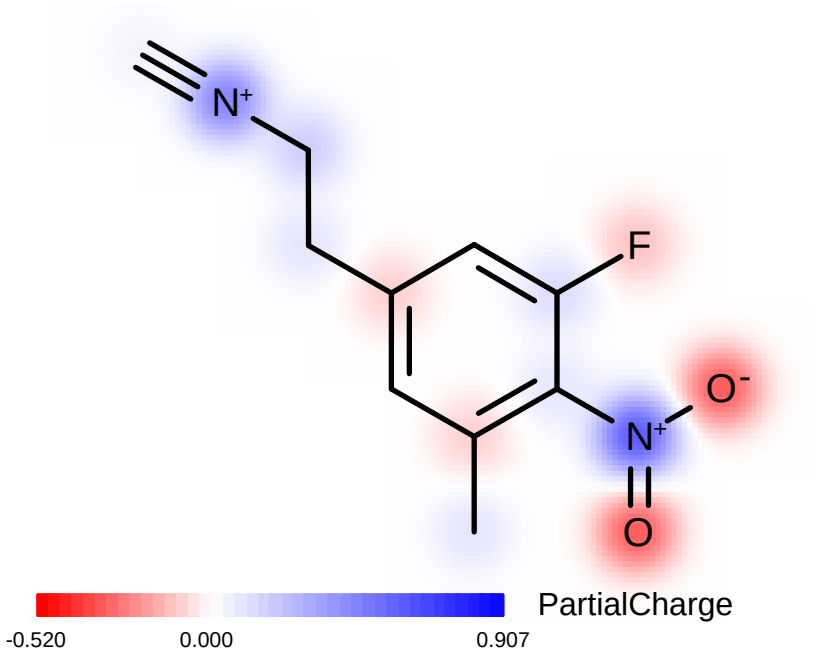


Fig. 8: **Example of using the SetRadiusRatio method**

See also

- *OE2DPropMap.GetRadiusRatio* method

SetResolution

```
void SetResolution(unsigned int resolution)
```

Sets the resolution of the property grid.

resolution
This value has to be in the range of [2, 20].

Example (see *Figure: Example of using the SetResolution method*)

```
OE2DPropMap propmap = new OE2DPropMap();
propmap.SetResolution(5);
```

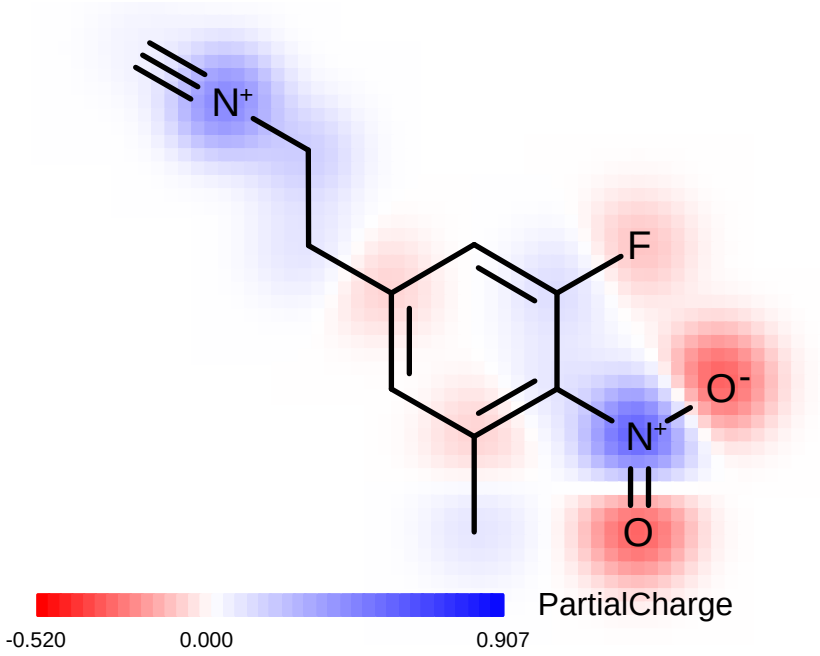


Fig. 9: **Example of using the SetResolution method**

See also

- *OE2DPropMap.GetResolution* method.

4.1.5 OEAlignedDepictionFrom3DOptions

```
class OEAlignedDepictionFrom3DOptions
```

This class represents *OEAlignedDepictionFrom3DOptions* that stores parameters that are used when generating 2D coordinates by calling the *OEPrepareAlignedDepictionFrom3D* function.

Property	Get method	Set method
clear coords	*GetClearCoords*	*SetClearCoords*
max number of bond rotations	*GetMaxBondRotations*	*SetMaxBondRotations*
suppressing hydrogens	*GetSuppressHydrogens*	*SetSuppressHydrogens*

Constructors

```
OEAlignedDepictionFrom3DOptions(bool clearCoords = true,
                                bool suppressH = true)
```

Default constructor that initializes an *OEAlignedDepictionFrom3DOptions* object with the following properties:

Table 1: **Default parameters of the OEAlignedDepiction-From3DOptions class**

Property	Default value
clear coordinates	true
max number of bond rotations	$2^{16} = 65536$
suppressing hydrogens	true

```
OEAlignedDepictionFrom3DOptions(const OEAlignedDepictionFrom3DOptions& rhs)
```

Copy constructor.

GetClearCoords

```
bool GetClearCoords() const
```

See also

- *OEAlignedDepictionFrom3DOptions.SetClearCoords* method

GetMaxBondRotations

```
unsigned int GetMaxBondRotations() const
```

See also

- *OEAlignedDepictionFrom3DOptions.SetSuppressHydrogens* method

GetSuppressHydrogens

```
bool GetSuppressHydrogens() const
```

Returns whether the explicit hydrogens are suppressed in the molecule prior to generating the 2D coordinates.

See also

- *OEAlignedDepictionFrom3DOptions.SetSuppressHydrogens* method

SetClearCoords

```
void SetClearCoords(bool clearcoords)
```

If false and the fit molecule has 2D coordinates, then these coordinates are used for the alignment. Otherwise the 2D coordinates of the fitted molecule are generated by calling the *OEPrepareDepictionFrom3D* function.

See also

- *OEAlignedDepictionFrom3DOptions.GetClearCoords* method

SetMaxBondRotations

```
void SetMaxBondRotations(unsigned int maxrotations)
```

Sets the maximum number of bond rotations performed during the 2D coordinate generation process.

maxrotations

The zero value means that there is no limit. Upon reaching this limit, the process terminates and returns the "best" aligned 2D coordinates.

Hint

The default value for maximum number of bond rotations is $2^{16} = 65536$. This means that by default only the first 16 single bonds of the molecule will be rotated to find the "best" aligned 2D layout.

It is recommended to use this parameter if the input 3D molecule has a large number of rotatable bonds.

See also

- *OEAlignedDepictionFrom3DOptions.GetMaxBondRotations* method

SetSuppressHydrogens

```
void SetSuppressHydrogens(bool)
```

Sets whether the explicit hydrogens are suppressed in the molecule prior to generating the 2D coordinates. Only hydrogens that are necessary to faithfully represent tetrahedral stereochemistry will be kept.

See also

- *OEAlignedDepictionFrom3DOptions.GetSuppressHydrogens* method

4.1.6 OEAtomGlyphBase

```
class OEAtomGlyphBase
```

OEAtomGlyphBase is an abstract class that defines the interface used for atom annotation.

See also

- *Annotating Atoms and Bonds* chapter
- *OEAddGlyph* function.

The following classes derive from this class:

- *OEAtomGlyphCircle*

Constructors

```
OEAtomGlyphBase()
```

Default constructor.

CreateCopy

```
virtual OEAtomGlyphBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEAtomBase* atom) const = 0
```

It is a virtual const method that is implemented in the concrete derived classes to annotate the given atom (*i.e.* to draw a *glyph* to mark the atom).

disp
: The OE2DMolDisplay object that is modified to annotate the specified atom.

atom
: The OEAtomBase object being annotated.

4.1.7 OEAtomGlyphCircle

```
class OEAtomGlyphCircle : public OEAtomGlyphBase
```

By using the *OEAtomGlyphCircle* class, atoms can be annotated by drawing a circle around them.

See also

- *Annotating Atoms and Bonds* chapter
- *OECircleStyle* namespace

The following methods are publicly inherited from *OEAtomGlyphBase*:

CreateCopy *RenderGlyph*

Constructors

```
OEAtomGlyphCircle(const OEDepict::OEPen& pen,
                  unsigned int circlestyle = OECircleStyle::Default,
                  double circleRadiusScale = 1.0,
                  unsigned int layer = OEDepict::OELayerPosition::Below)
```

Creates an *OEAtomGlyphCircle* object with the specified parameters

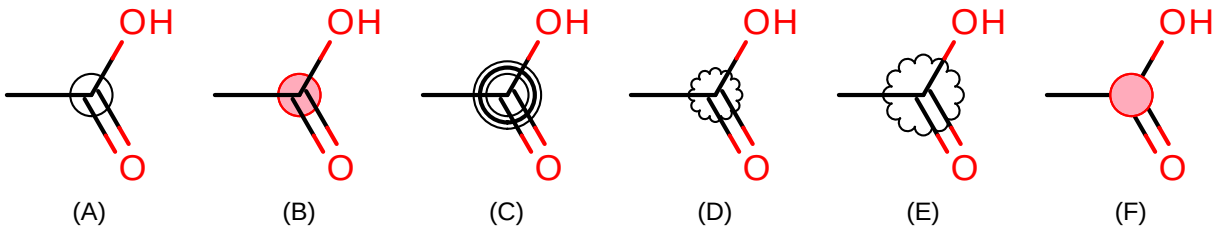


Fig. 10: **Example of atom annotations**

pen
The pen used when drawing a circle around an atom.

See example (A) and (B) in *Figure: Example of atom annotations*.

circlestyle
The style of the drawn circle from the `OECircleStyle` namespace.

See examples (C) and (D) in *Figure: Example of atom annotations*.

circleRadiusScale
The multiplier used to modify the radius of the drawn circles

See examples (D) and (E) in *Figure: Example of atom annotations*.

layer
Specifies whether the circle is drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

See examples (B) and (F) in *Figure: Example of atom annotations*.

CreateCopy

```
OEAtomGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEAtomGlyphCircle* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEAtomBase* atom) const
```

Draws a glyph (in this case a circle) around the given atom.

disp
The OE2DMolDisplay object that is modified to annotate the specified atom.

atom
The OEAtomBase object being annotated.

4.1.8 OEBondGlyphArrow

```
class OEBondGlyphArrow : public OEBondGlyphBase
```

By using the *OEBondGlyphArrow* class, bonds can be annotated by drawing an arrow across them.

See also

- *Annotating Atoms and Bonds* chapter
- *OEBondGlyphCircle* class
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphArrow(const OEDepict::OEPen &,
                 double arrowLengthScale = 1.0,
                 unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphArrow* object with the specified parameters.

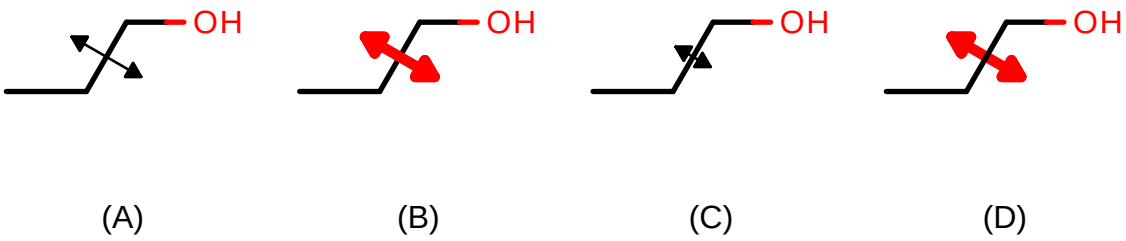


Fig. 11: **Example of bond annotations**

pen
: The pen used when drawing an arrow across the middle of the bond.

  See examples (A) and (B) in *Figure: Example of bond annotations*.

arrowLengthScale
: The multiplier used to modify the length of the drawn arrows.

See examples (A) and (C) in *Figure: Example of bond annotations*.

layer
Specifies whether the arrow is drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

See examples (B) and (D) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphArrow* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a straight arrow) in the middle of the given bond.

disp
The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
The OEBondBase object being annotated.

4.1.9 OEBondGlyphBase

```
class OEBondGlyphBase
```

OEBondGlyphBase is an abstract class that defines the interface used for bond annotation.

See also

- *Annotating Atoms and Bonds* chapter
- *OEAddGlyph* function.

The following classes derive from this class:

- *OEBondGlyphArrow*
- *OEBondGlyphCircle*
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

Constructors

```
OEBondGlyphBase()
```

Default constructor.

CreateCopy

```
virtual OEBondGlyphBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay &,
                 const OEChem::OEBondBase *) const = 0
```

It is a virtual const method that is implemented in the concrete derived classes to annotate the given bond (*i.e.* to draw a *glyph* to mark the bond).

disp
: The OE2DMolDisplay object that is modified to annotate the specified bond.

atom
: The OEBondBase object being annotated.

4.1.10 OEBondGlyphCircle

```
class OEBondGlyphCircle : public OEBondGlyphBase
```

By using the *OEBondGlyphCircle* class, bonds can be annotated by drawing a circle in the middle of them.

See also

- *Annotating Atoms and Bonds* chapter
- *OECircleStyle* namespace
- *OEBondGlyphArrow* class
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphCircle(const OEDepict::OEPen &,
                  unsigned int circlestyle = OECircleStyle::Default,
                  double circleRadiusScale = 1.0,
                  unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphCircle* object with the specified parameters.

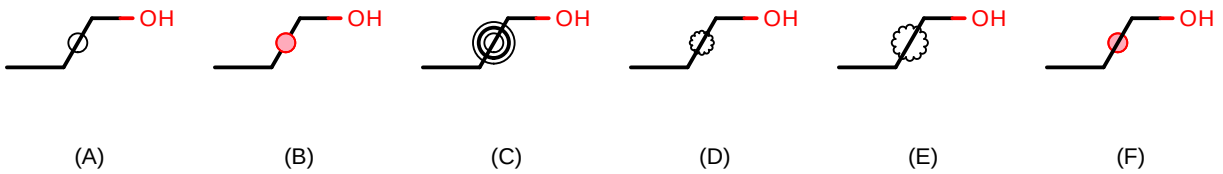


Fig. 12: **Example of bond annotations**

pen
: The pen used when drawing a circle in the middle of a bond.

  See examples (A) and (B) in *Figure: Example of bond annotations*.

circlestyle
: The style of the drawn circle from the `OECircleStyle` namespace.

  See examples (C) and (D) in *Figure: Example of bond annotations*.

circleRadiusScale
: The multiplier used to modify the radius of the drawn circles.

  See examples (D) and (E) in *Figure: Example of bond annotations*.

layer
: Specifies whether the circle is drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

  See examples (B) and (F) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphCircle* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a circle) in the middle of the given bond.

disp
: The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
The OEBondBase object being annotated.

4.1.11 OEBondGlyphCross

```
class OEBondGlyphCross : public OEBondGlyphBase
```

By using the *OEBondGlyphCross* class, bonds can be annotated by drawing a cross in the middle of them.

See also

- *Annotating Atoms and Bonds* chapter
- `OECircleStyle` namespace
- *OEBondGlyphArrow* class
- *OEBondGlyphCircle* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphCross(const OEDepict::OEPen& pen,
                 double crossLengthScale = 1.0,
                 unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphCross* object with the specified parameters.

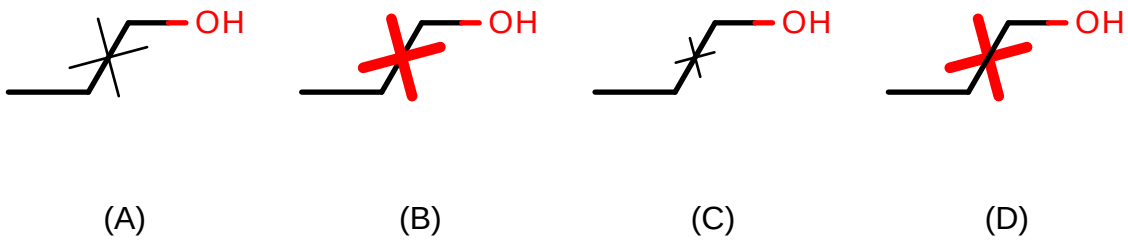


Fig. 13: **Example of bond annotations**

pen
The pen used when drawing a cross across the middle of the bond.

See examples (A) and (B) in *Figure: Example of bond annotations*.

crossLengthScale
The multiplier used to modify the length of the drawn cross.

See examples (A) and (C) in *Figure: Example of bond annotations*.

layer
Specifies whether the cross drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

See examples (B) and (D) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphCross* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a cross) in the middle of the given bond.

disp
The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
The OEBondBase object being annotated.

4.1.12 OEBondGlyphCurvedArrow

```
class OEBondGlyphCurvedArrow : public OEBondGlyphBase
```

By using the *OEBondGlyphCurvedArrow* class, bonds can be annotated by drawing a curved arrow across them.

See also

- *Annotating Atoms and Bonds* chapter
- *OEBondGlyphArrow* class
- *OEBondGlyphCircle* class
- *OEBondGlyphCross* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphCurvedArrow(const OEDepict::OEPen& pen,
                       double arrowRadiusScale = 1.0,
                       bool clockwise = true,
                       double arcAngle = 270.0,
                       unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphCurvedArrow* object with the specified parameters.

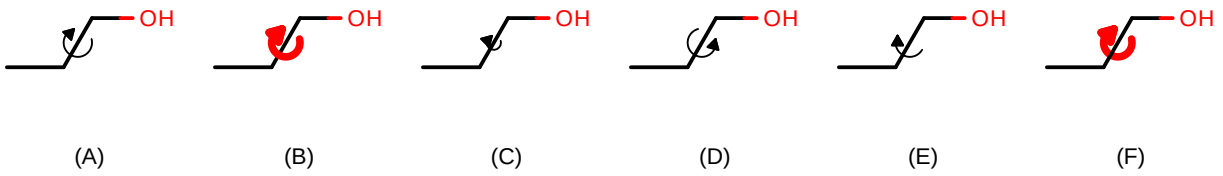


Fig. 14: **Example of bond annotations**

pen
: The pen used when drawing a curved arrow at the middle of the bond.

  See examples (A) and (B) in *Figure: Example of bond annotations*.

arrowRadiusScale
: The multiplier used to modify the radius of the drawn arc.

  See examples (A) and (C) in *Figure: Example of bond annotations*.

clockwise
: The direction of the arrow.

  See examples (A) and (D) in *Figure: Example of bond annotations*.

arcAngle
: The angel of the arc in degrees. The value has to be in a range of $[0.0°, 360.0°]$.

  See examples (A) and (E) in *Figure: Example of bond annotations*.

layer
: Specifies whether the curved arrow is drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

  See examples (B) and (F) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphCurvedArrow* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a curved arrow) in the middle of the given bond.

disp
: The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
: The OEBondBase object being annotated.

4.1.13 OEBondGlyphScissors

```
class OEBondGlyphScissors : public OEBondGlyphBase
```

By using the *OEBondGlyphCross* class, bonds can be annotated by drawing a cross in the middle of them.

See also

- *Annotating Atoms and Bonds* chapter
- *OECircleStyle* namespace
- *OEBondGlyphArrow* class
- *OEBondGlyphCircle* class
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphStitch* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphScissors(const OEDepict::OEPen& pen,
                    double scissorsLengthScale = 1.0,
                    unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphCross* object with the specified parameters.

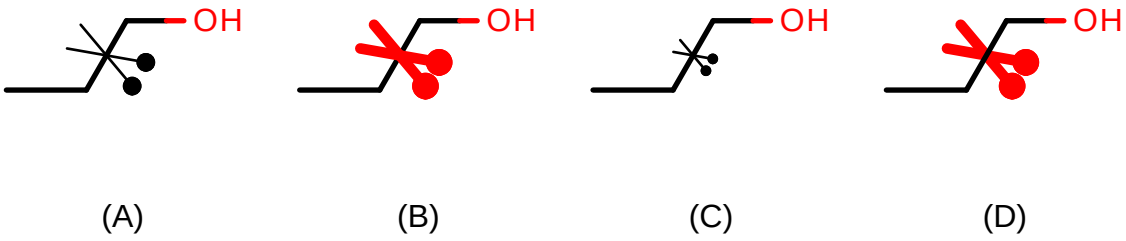


Fig. 15: **Example of bond annotations**

pen
: The pen used when drawing scissors across the middle of the bond.

  See examples (A) and (B) in *Figure: Example of bond annotations*.

scissorsLengthScale
: The multiplier used to modify the length of the drawn scissors.

  See examples (A) and (C) in *Figure: Example of bond annotations*.

layer
: Specifies whether the scissors are drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

  See examples (B) and (D) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphScissors* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a scissors) in the middle of the given bond.

disp
: The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
: The OEBondBase object being annotated.

4.1.14 OEBondGlyphStitch

```
class OEBondGlyphStitch : public OEBondGlyphBase
```

By using the *OEBondGlyphStitch* class, bonds can be annotated by drawing perpendicular lines across them.

See also

- *Annotating Atoms and Bonds* chapter
- *OEBondGlyphArrow* class
- *OEBondGlyphCircle* class
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphZigZag* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphStitch(const OEDepict::OEPen& pen,
                  unsigned int nrstitches = 2u,
                  double stitchLengthScale = 1.0,
                  unsigned int layer = OEDepict::OELayerPosition::Above)
```

Creates an *OEBondGlyphZigZag* object with the specified parameters.

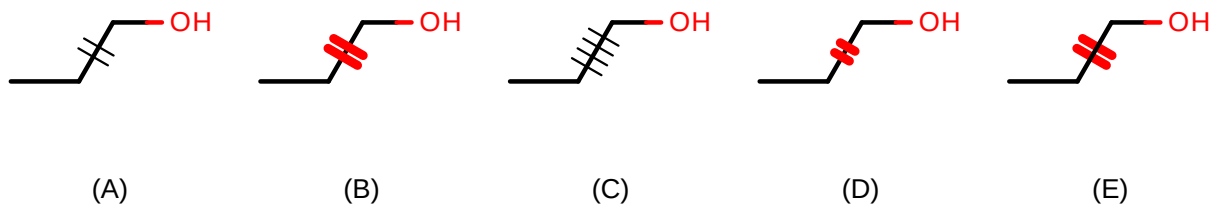


Fig. 16: **Example of bond annotations**

pen
The pen used when drawing the perpendicular lines across the bond.

See examples (A) and (B) in *Figure: Example of bond annotations*.

nrstitches
The number of perpendicular lines drawn across the bond. This value has to be in the range of [1, 5].

See examples (A) and (C) in *Figure: Example of bond annotations*.

stitchLengthScale
The multiplier used to modify the length of the drawn perpendicular lines.

See examples (A) and (D) in *Figure: Example of bond annotations*.

layer

Specifies whether the perpendicular lines are drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

See examples (B) and (E) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphStitch* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

4.1.15 OEBondGlyphZigZag

```
class OEBondGlyphZigZag : public OEBondGlyphBase
```

By using the *OEBondGlyphZigZag* class, bonds can be annotated by drawing a zig-zag line across them.

See also

- *Annotating Atoms and Bonds* chapter
- *OEBondGlyphArrow* class
- *OEBondGlyphCircle* class
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class
- *OEBondGlyphStitch* class

The following methods are publicly inherited from *OEBondGlyphBase*:

CreateCopy	*RenderGlyph*

Constructors

```
OEBondGlyphZigZag(const OEDepict::OEPen& pen,
                  double zigzagLengthScale = 1.0,
                  unsigned int layer = OEDepict::OELayerPosition::Above,
                  bool diagonal = false)
```

Creates an *OEBondGlyphZigZag* object with the specified parameters.

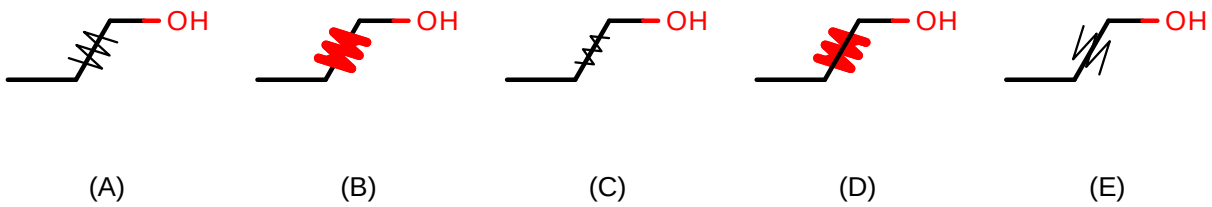


Fig. 17: **Example of bond annotations**

pen
The pen used when drawing a zig-zag line across bond.

See examples (A) and (B) in *Figure: Example of bond annotations*.

zigzagLengthScale
The multiplier used to modify the length of the drawn zig-zag line.

See examples (A) and (C) in *Figure: Example of bond annotations*.

layer
Specifies whether the zig-zag line is drawn above (`OELayerPosition.Above`) or below (`OELayerPosition.Below`) the molecule.

See examples (B) and (D) in *Figure: Example of bond annotations*.

diagonal
Specifies whether the zig-zag line is drawn across the bond or in parallel with it.

See examples (A) and (E) in *Figure: Example of bond annotations*.

CreateCopy

```
OEBondGlyphBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBondGlyphZigZag* object is dynamically allocated and owned by the caller.

RenderGlyph

```
bool RenderGlyph(OEDepict::OE2DMolDisplay& disp,
                 const OEChem::OEBondBase* bond) const
```

Draws a glyph (in this case a zig-zag line) across the given bond.

disp
The OE2DMolDisplay object that is modified to annotate the specified bond.

bond
The OEBondBase object being annotated.

4.1.16 OEColorGradientDisplayOptions

```
class OEColorGradientDisplayOptions
```

This class represents the *OEColorGradientDisplayOptions* class that holds properties that determine how a color gradient is rendered.

See also

- *OEDrawColorGradient* function

Constructors

```
OEColorGradientDisplayOptions()
```

Default constructor.

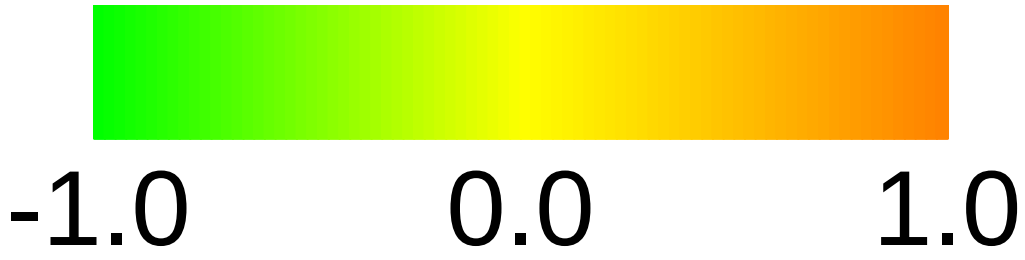


Fig. 18: **Example of drawing a color gradient with default options**

```
OEColorGradientDisplayOptions(const OEColorGradientDisplayOptions& rhs)
```

Copy constructor.

AddLabel

```
void AddLabel(const OEColorGradientLabel& label)
```

Adds an additional label that will be rendered on the strip of the color gradient.

label
The *OEColorGradientLabel* object that defines the label and the position where the label is going to be rendered to the color gradient.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddLabel(new OEColorGradientLabel(-1.0, "dissimilar"));
opts.AddLabel(new OEColorGradientLabel(+1.0, "similar"));
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

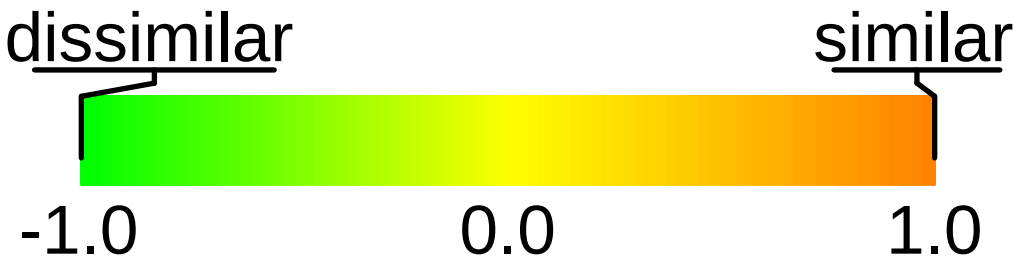


Fig. 19: **Example of using the AddLabel method**

See also

- *OEColorGradientLabel* class
- *OEColorGradientDisplayOptions.GetLabels* method
- *OEColorGradientDisplayOptions.ClearLabels* method

AddMarkedValue

```
void AddMarkedValue(double value)
```

Adds an additional value that will be marked on the strip of the color gradient along with the label of the value above the strip.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddMarkedValue(0.20);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

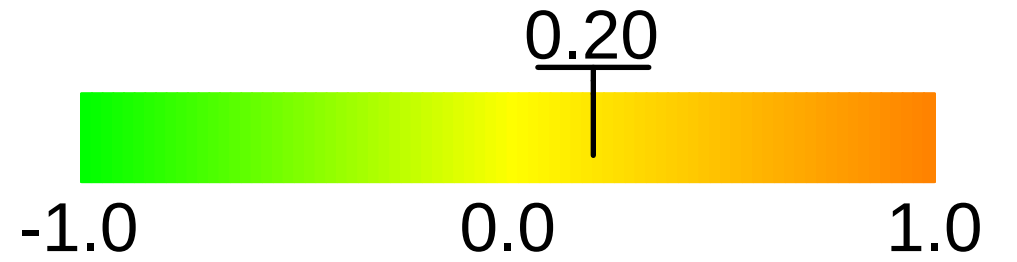


Fig. 20: **Example of using the AddMarkedValue method**

See also

- *OEColorGradientDisplayOptions.AddMarkedValues* method
- *OEColorGradientDisplayOptions.GetMarkedValues* method
- *OEColorGradientDisplayOptions.SetMarkedValues* method
- *OEColorGradientDisplayOptions.ClearMarkedValues* method

AddMarkedValues

```
void AddMarkedValues(const std::vector<double>& values)
```

Adds additional values that will be marked on the strip of the color gradient along with their labels above the strip.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.AddMarkedValues(markedvalues);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

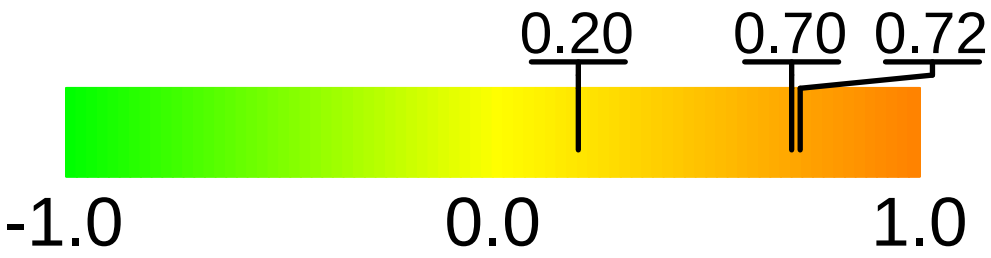


Fig. 21: **Example of using the AddMarkedValues method**

See also

- *OEColorGradientDisplayOptions.AddMarkedValue* method
- *OEColorGradientDisplayOptions.ClearMarkedValues* method
- *OEColorGradientDisplayOptions.GetMarkedValues* method
- *OEColorGradientDisplayOptions.SetMarkedValues* method

ClearBoxRange

```
void ClearBoxRange()
```

Removes the range previously set to be marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.SetBoxRange* method
- *OEColorGradientDisplayOptions.GetBoxRangeMax* method
- *OEColorGradientDisplayOptions.GetBoxRangeMin* method

ClearLabels

```
void ClearLabels()
```

Removes labels previously set by the *OEColorGradientDisplayOptions.AddLabel* method to be marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.AddLabel* method
- *OEColorGradientDisplayOptions.GetLabels* method

ClearMarkedValues

```
void ClearMarkedValues()
```

Removes any values previously set to be marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.AddMarkedValue* method
- *OEColorGradientDisplayOptions.AddMarkedValues* method
- *OEColorGradientDisplayOptions.GetMarkedValues* method
- *OEColorGradientDisplayOptions.SetMarkedValues* method

GetBoxRangeMax

```
double GetBoxRangeMax() const
```

Returns the minimum value of the range being marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.ClearBoxRange* method
- *OEColorGradientDisplayOptions.SetBoxRange* method

GetBoxRangeMin

```
double GetBoxRangeMin() const
```

Returns the maximum value of the range being marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.ClearBoxRange* method
- *OEColorGradientDisplayOptions.SetBoxRange* method

GetBoxRangePen

Returns the pen used to draw the box to mark the range set by the *OEColorGradientDisplayOptions.SetBoxRange* method.

```
const OEDepict::OEPen& GetBoxRangePen() const
```

See also

- OEPen class
- *OEColorGradientDisplayOptions.SetBoxRange*
- *OEColorGradientDisplayOptions.SetBoxRangePen* method

GetColorStopLabelFont

```
const OEDepict::OEFont& GetColorStopLabelFont() const
```

Returns the font that is used to render the color stops of the color gradient.

See also

- OEFont class
- *OEColorGradientDisplayOptions.SetColorStopLabelFont* method

GetColorStopLabelFontScale

```
double GetColorStopLabelFontScale() const
```

Returns the multiplier that can be used to increase or decrease the size of the color stop label fonts.

See also

- *OEColorGradientDisplayOptions.SetColorStopLabelFontScale* method

GetColorStopPrecision

```
unsigned int GetColorStopPrecision() const
```

Returns the precision of the color stop labels *i.e.* the number of digits that is written to express the floating-point values.

See also

- *OEColorGradientDisplayOptions.GetColorStopPrecision* method

GetColorStopVisibility

```
bool GetColorStopVisibility() const
```

Returns whether the color stop labels are rendered.

See also

- *OEColorGradientDisplayOptions.SetColorStopVisibility* method

GetLabels

```
OESystem::OEIterBase<OEColorGradientLabel>* GetLabels() const
```

Returns an iterator over the labels that are being rendered onto the color gradient.

See also

- *OEColorGradientLabel* class
- *OEColorGradientDisplayOptions.AddLabel* method
- *OEColorGradientDisplayOptions.ClearLabels* method

GetMarkedValuePen

```
const OEDepict::OEPen& GetMarkedValuePen() const
```

Returns the pen that is used to draw the lines for the marked values.

See also

- OEPen class
- *OEColorGradientDisplayOptions.SetMarkedValuePen* method

GetMarkedValuePrecision

```
unsigned int GetMarkedValuePrecision() const
```

Returns the precision of the marked value labels *i.e.* the number of digits that is written to express the floating-point values.

See also

- *OEColorGradientDisplayOptions.SetMarkedValuePrecision* method

GetMarkedValues

```
OESystem::OEIterBase<double>* GetMarkedValues() const
```

Returns an iterator over the values that are being marked on the color gradient.

See also

- *OEColorGradientDisplayOptions.AddMarkedValue* method
- *OEColorGradientDisplayOptions.AddMarkedValues* method
- *OEColorGradientDisplayOptions.ClearMarkedValues* method
- *OEColorGradientDisplayOptions.SetMarkedValues* method

SetBoxRange

```
void SetBoxRange(double minvalue, double maxvalue)
```

Sets the range being marked on the color gradient.

minvalue, maxvalue
The minimum and maximum value of the range.

Example:

Full listing.

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetBoxRange(-0.5, 0.5);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```



Fig. 22: **Example of using the SetBoxRange method**

See also

- *OEColorGradientDisplayOptions.ClearBoxRange* method

- *OEColorGradientDisplayOptions.GetBoxRangeMax* method
- *OEColorGradientDisplayOptions.GetBoxRangeMin* method
- *OEColorGradientDisplayOptions.SetBoxRange* method

SetBoxRangePen

```
void SetBoxRangePen(const OEDepict::OEPen& pen)
```

Sets the pen used to mark a specified range on the color gradient.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetBoxRange(-0.5, 0.5);
OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack, OEFill.On, 2.0, OEStipple.
↪ShortDash);
opts.SetBoxRangePen(pen);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

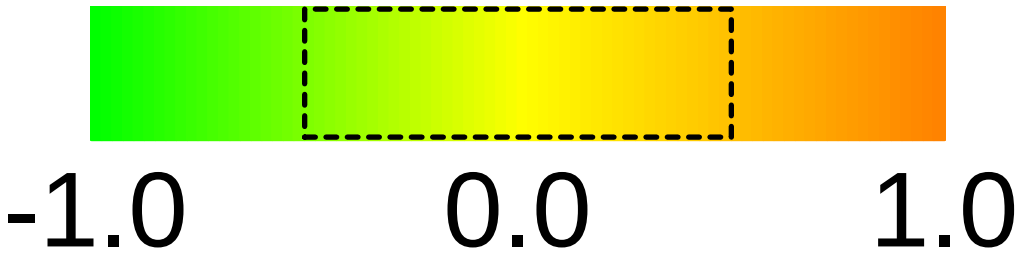


Fig. 23: **Example of using the SetBoxRangePen method**

See also

- OEPen class
- *OEColorGradientDisplayOptions.GetBoxRangePen* method
- *OEColorGradientDisplayOptions.SetBoxRange* method

SetColorStopLabelFont

```
void SetColorStopLabelFont(const OEDepict::OEFont& font)
```

Sets the font that is used to render color stops of the color gradient.

Example:

Full listing.

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEFont font = new OEFont();
```

(continues on next page)

(continued from previous page)

```
font.SetColor(OEChem.OEDarkGreen);
font.SetStyle(OEFontStyle.Bold);
opts.SetColorStopLabelFont(font);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

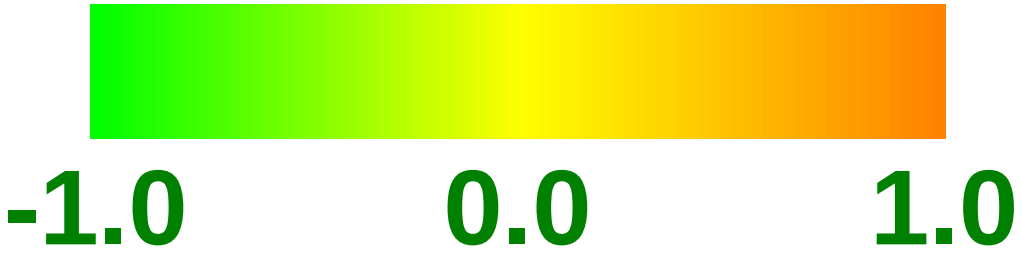


Fig. 24: **Example of using the SetColorStopLabelFont method**

Note

The size of font also depends on dimensions of the image into which the color gradient is rendered. See also *OEDrawColorGradient* function.

See also

- OEFont class
- *OEColorGradientDisplayOptions.GetColorStopLabelFont* method

SetColorStopLabelFontScale

```
void SetColorStopLabelFontScale(double scale)
```

Sets the multiplier that can be used increase or decrease the size of the fonts of the color stop labels.

scale
: This value has to be in the range of `[0.5, 3.0]`.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetColorStopLabelFontScale(0.5);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

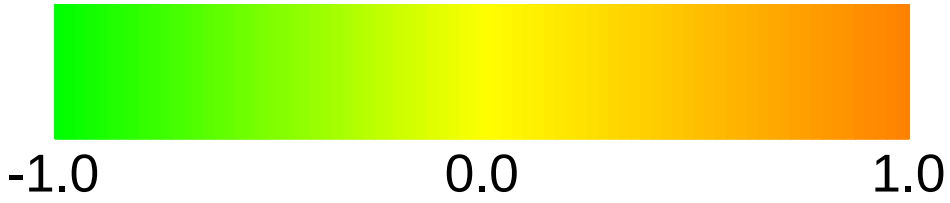


Fig. 25: **Example of using the SetColorStopLabelFontScale method**

See also

- *OEColorGradientDisplayOptions.GetColorStopLabelFontScale* method
- *OEColorGradientDisplayOptions.GetColorStopLabelFont* method
- *OEColorGradientDisplayOptions.SetColorStopLabelFont* method

SetColorStopPrecision

```
void SetColorStopPrecision(unsigned int precision)
```

Sets the precision of the color stop labels *i.e.* the number of digits that is written to express the floating-point values.

precision
: The zero value means that the precision is determined by the lowest significant digits of the color stop values that is in the range of [1,4].

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetColorStopPrecision(4);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

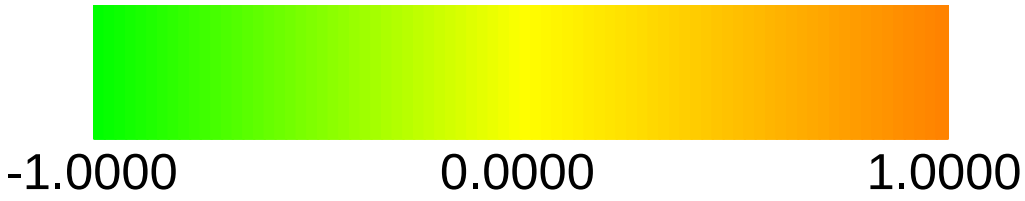


Fig. 26: **Example of using the SetColorStopPrecision method**

See also

- *OEColorGradientDisplayOptions.GetColorStopPrecision* method

SetColorStopVisibility

```
void SetColorStopVisibility(bool visible)
```

Sets whether the color stop labels are rendered.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetColorStopVisibility(false);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetColorStopVisibility
↪");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetBoxRange(-0.5, 0.5);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetBoxRange");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.SetBoxRange(-0.5, 0.5);
OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEBlack, OEFill.On, 2.0, OEStipple.
↪ShortDash);
opts.SetBoxRangePen(pen);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetBoxRangePen");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddMarkedValue(0.20);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_AddMarkedValue");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.AddMarkedValues(markedvalues);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_AddMarkedValues");
}
```

(continues on next page)

(continued from previous page)

```
{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddMarkedValue(0.0);
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.75);
markedvalues.Add(0.50);
opts.SetMarkedValues(markedvalues);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetMarkedValues");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.SetMarkedValues(markedvalues);
opts.SetMarkedValuePrecision(1);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetMarkedValuePrecision
↪");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.SetMarkedValues(markedvalues);
OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEDarkGreen, OEFill.On, 2.0, OEStipple.
↪ShortDash);
opts.SetMarkedValuePen(pen);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_SetMarkedValuePen");
}

{
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddLabel(new OEColorGradientLabel(-1.0, "dissimilar"));
opts.AddLabel(new OEColorGradientLabel(+1.0, "similar"));
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
DepictColorGradient(colorg, opts, "OEColorGradientDisplayOptions_AddLabel");
}

return 0;
```

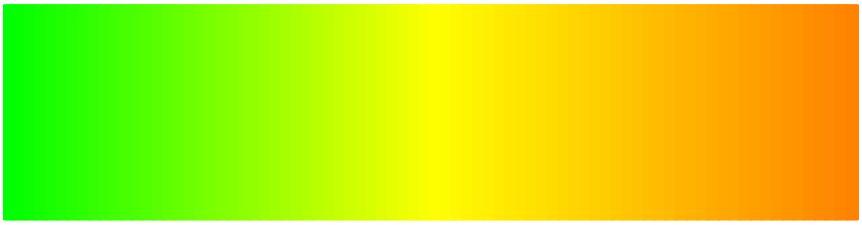


Fig. 27: **Example of using the SetColorStopVisibility method**

See also

- *OEColorGradientDisplayOptions.GetColorStopVisibility* method

SetMarkedValuePen

```
void SetMarkedValuePen(const OEDepict::OEPen& pen)
```

Sets the pen that is used to draw the lines for the marked values.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.SetMarkedValues(markedvalues);
OEPen pen = new OEPen(OEChem.OEBlack, OEChem.OEDarkGreen, OEFill.On, 2.0, OEStipple.
↪ShortDash);
opts.SetMarkedValuePen(pen);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

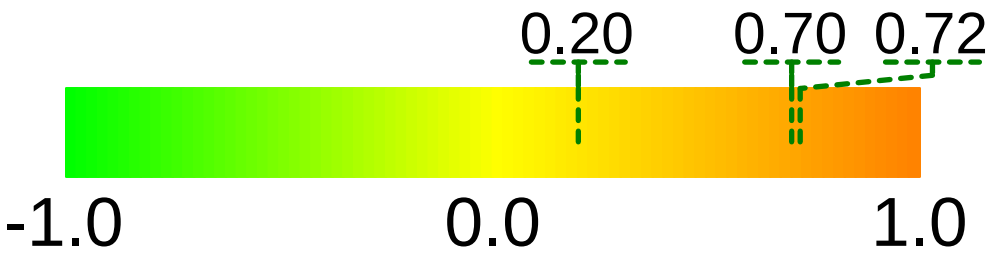


Fig. 28: **Example of using the SetMarkedValuePen method**

See also

- OEPen class
- *OEColorGradientDisplayOptions.GetMarkedValuePen* method

SetMarkedValuePrecision

```
void SetMarkedValuePrecision(unsigned int precision)
```

Sets the precision of the marked value labels *i.e.* the number of digits that is written to express the floating-point values. Decreasing the precision can result in collapsing that labels of two or more distinct values into one label.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.20);
markedvalues.Add(0.70);
markedvalues.Add(0.72);
opts.SetMarkedValues(markedvalues);
opts.SetMarkedValuePrecision(1);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

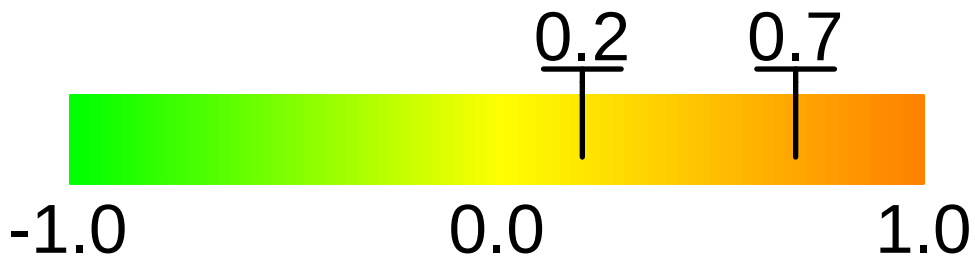


Fig. 29: **Example of using the SetMarkedValuePrecision method**

See also

- *OEColorGradientDisplayOptions.GetMarkedValuePrecision* method

SetMarkedValues

```
void SetMarkedValues(const std::vector<double>& values)
```

Initializes values that will be marked on the strip of the color gradient along with the value labels above the strip. *OEColorGradientDisplayOptions.SetMarkedValues* method discards the list of values set in prior calls.

Example:

```
OEColorGradientDisplayOptions opts = new OEColorGradientDisplayOptions();
opts.AddMarkedValue(0.0);
```

(continues on next page)

(continued from previous page)

```
OEDoubleVector markedvalues = new OEDoubleVector();
markedvalues.Add(0.75);
markedvalues.Add(0.50);
opts.SetMarkedValues(markedvalues);
OEGrapheme.OEDrawColorGradient(image, colorg, opts);
```

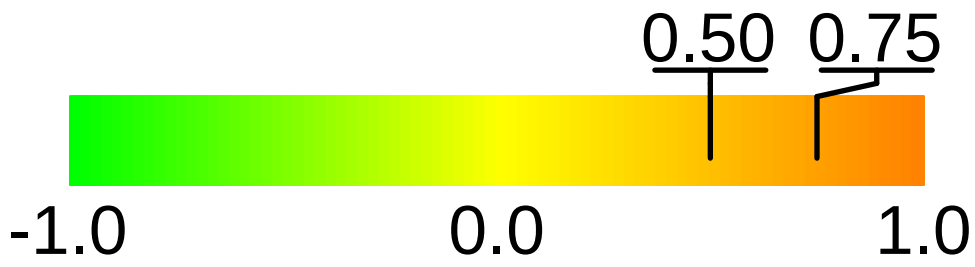


Fig. 30: **Example of using the SetMarkedValues method**

See also

- *OEColorGradientDisplayOptions.AddMarkedValue* method
- *OEColorGradientDisplayOptions.AddMarkedValues* method
- *OEColorGradientDisplayOptions.ClearMarkedValues* method

4.1.17 OEColorGradientLabel

```
class OEColorGradientLabel
```

The *OEColorGradientLabel* class stores a value (double) and label (string) pair that is used by the *OEColorGradientDisplayOptions* class to render user defined labels into the color gradients.

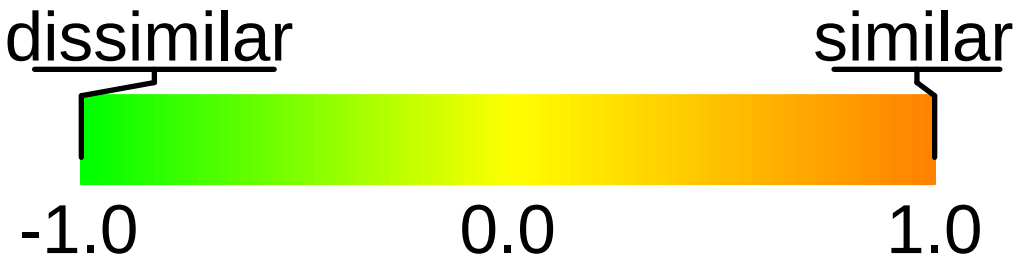


Fig. 31: **Example of depicting labels above the color gradient**

See also

- *OEColorGradientDisplayOptions.AddLabel* method

Constructors

```
OEColorGradientLabel(double value, const std::string& label)
```

value
Defines the position of the string being rendered.

label
Defines the string being rendered.

Constructor.

```
OEColorGradientLabel(const OEColorGradientLabel& rhs)
```

Copy constructor.

GetLabel

```
std::string GetLabel() const
```

Returns the string stored in the *OEColorGradientLabel* object that will be rendered into the color gradient.

GetValue

```
double GetValue() const
```

Returns the value stored in the *OEColorGradientLabel* object that defined the position of the associated label on the color gradient.

4.1.18 OEColorForceFieldDisplay

```
class OEColorForceFieldDisplay
```

This class represents *OEColorForceFieldDisplay* that stores the depiction information of a OEColorForceField object. See example in *Figure: Example of depicting a color force field in a legend*.

See also

- The Theory chapter of the **Shape TK** manual.
- *OEColorForceFieldLegendDisplayOptions* class
- *OEDrawColorForceFieldLegend* function

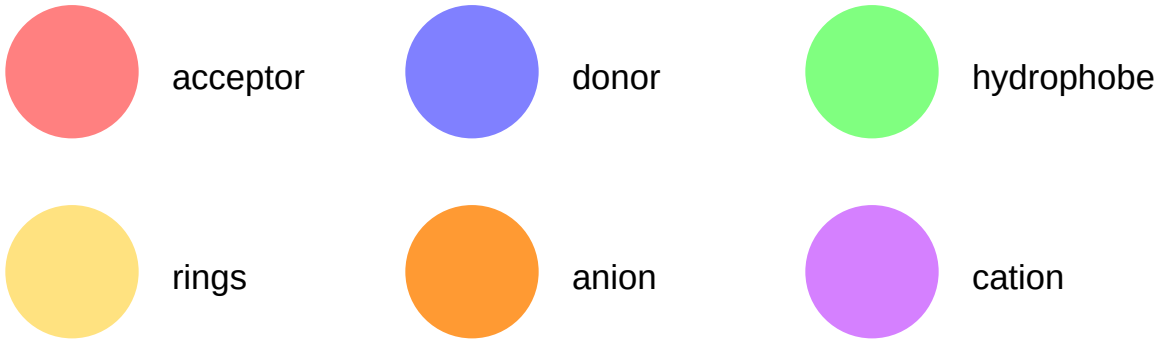


Fig. 32: **Example of depicting a color force field in a legend**

Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

Constructors

```
OEColorForceFieldDisplay(const OEShape::OEColorForceField& cff,
                         bool preserveQueryColors = false)
```

Creates an *OEColorForceFieldDisplay* object with a given color force field and an option to preserve colors on input query molecules.

```
OEColorForceFieldDisplay(const OEColorForceFieldDisplay& rhs)
```

Copy constructor.

GetColor

```
const OESystem::OEColor& GetColor(unsigned int type) const
```

Returns the color that is associated with the given color atom type.

GetColorTypes

```
OESystem::OEIterBase<unsigned int>* GetColorTypes() const
```

Returns the iterator over the color atom types.

GetName

```
std::string GetName(unsigned int type) const
```

Returns the name that is associated with the given color atom type.

NumColorTypes

```
unsigned int NumColorTypes() const
```

Returns the number of color atom types.

SetColor

```
void SetColor(unsigned int type, const OEDepict::OEColor& color)
```

Sets the color that is associated with the given color atom type.

SetName

```
void SetName(unsigned int type, const std::string& name)
```

Sets the name that is associated with the given color atom type.

4.1.19 OEColorForceFieldLegendDisplayOptions

```
class OEColorForceFieldLegendDisplayOptions
```

This class represents *OEColorForceFieldLegendDisplayOptions* that encapsulates properties that determine how the legend of a color force field is depicted by the *OEDrawColorForceFieldLegend* function.

See also

- *OEColorForceFieldDisplay* class
- *OEDrawColorForceFieldLegend* function

Constructors

```
OEColorForceFieldLegendDisplayOptions(unsigned int rows,
                                      unsigned int cols)
```

Creates an *OEColorForceFieldLegendDisplayOptions* object with the specified number of rows and columns..

rows
: The number of rows in the legend.

cols
: The number of columns in the legend.

```
OEColorForceFieldLegendDisplayOptions(const OEColorForceFieldLegendDisplayOptions& rhs)
```

Copy constructor.

NumCols

```
unsigned int NumCols() const
```

Returns the number of columns of the legend.

NumRows

```
unsigned int NumRows() const
```

Returns the number of rows of the legend.

4.1.20 OEColorOverlapDisplayOptions

```
class OEColorOverlapDisplayOptions : public OEDepict::OE2DMolDisplayOptions
```

This class represents *OEColorOverlapDisplayOptions* that encapsulates properties that determine how the color atom overlap between the reference and the fit molecule is depicted by the *OERenderColorOverlap* function.

See also

- *OEShapeOverlapDisplay* class
- *OERenderColorOverlap* function

Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

The following methods are publicly inherited from OE2DMolDisplayOptions:

operator=	GetHydrogenStyle	SetBondLineAtomLabelGapScale
GetAromaticStyle	GetMargin	SetBondLineGapScale
GetAtomColor	GetScale	SetBondPropLabelFont
GetAtomColorStyle	GetSuperAtomLabelFont	SetBondPropLabelFontScale
GetAtomLabelFont	GetSuperAtomStyle	SetBondPropertyFunctor
GetAtomLabelFontScale	GetTitleFont	SetBondStereoStyle
GetAtomPropLabelFont	GetTitleFontScale	SetBondWidthScaling
GetAtomPropLabelFontScale	GetTitleHeight	SetDefaultBondPen
GetAtomPropertyFunctor	GetTitleLocation	SetDimensions
GetAtomStereoStyle	GetWidth	SetHeight
GetBackgroundColor	SetAromaticStyle	SetHydrogenStyle
GetBondColorStyle	SetAtomColor	SetMargin
GetBondLineAtomLabelGapScale	SetAtomColorStyle	SetMargins
GetBondLineGapScale	SetAtomLabelFont	SetScale
GetBondPropLabelFont	SetAtomLabelFontScale	SetSuperAtomLabelFont
GetBondPropLabelFontScale	SetAtomPropLabelFont	SetSuperAtomStyle
GetBondPropertyFunctor	SetAtomPropLabelFontScale	SetTitleFont
GetBondStereoStyle	SetAtomPropertyFunctor	SetTitleFontScale
GetBondWidthScaling	SetAtomStereoStyle	SetTitleHeight
GetDefaultBondPen	SetBackgroundColor	SetTitleLocation
GetHeight	SetBondColorStyle	SetWidth

Constructors

```
OEColorOverlapDisplayOptions()
```

Default constructor.

```
OEColorOverlapDisplayOptions(const OEColorOverlapDisplayOptions& rhs)
```

Copy constructor.

ClearQuerySurfaceArcFxn

```
void ClearQuerySurfaceArcFxn()
```

Removes the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule.

See also

- *OEColorOverlapDisplayOptions.GetQuerySurfaceArcFxn*
- *OEColorOverlapDisplayOptions.SetQuerySurfaceArcFxn*

GetClearBackground

```
bool GetClearBackground() const
```

Returns whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the color overlap.

See also

- *OEColorOverlapDisplayOptions.SetClearBackground*

GetQuerySurfaceArcFxn

```
const OESurfaceArcFxnBase* GetQuerySurfaceArcFxn() const
```

Returns the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule.

See also

- *OEColorOverlapDisplayOptions.ClearQuerySurfaceArcFxn*
- *OEColorOverlapDisplayOptions.SetQuerySurfaceArcFxn*

SetClearBackground

```
void SetClearBackground(bool clear)
```

Returns whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the color overlap.

See also

- *OEColorOverlapDisplayOptions.GetClearBackground*

SetQuerySurfaceArcFxn

```
void SetQuerySurfaceArcFxn(const OESurfaceArcFxnBase& arcfxn)
```

Sets the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule when calling the *OERenderColorOverlap* function to depict the atom color overlap.

See also

- *OEColorOverlapDisplayOptions.ClearQuerySurfaceArcFxn*
- *OEColorOverlapDisplayOptions.GetQuerySurfaceArcFxn*

4.1.21 OEDepictionFrom3DOptions

```
class OEDepictionFrom3DOptions
```

This class represents *OEDepictionFrom3DOptions* that stores parameters that are used when generating 2D coordinates by calling the *OEPrepareDepictionFrom3D* function.

Property	Get method	Set method
inertial frame	*GetCoordsToInertialFrame*	*SetCoordsToInertialFrame*
max number of bond rotations	*GetMaxBondRotations*	*SetMaxBondRotations*
rotate to 2D reference	*GetRotateTo2DReference*	*SetRotateTo2DReference*
suppressing hydrogens	*GetSuppressHydrogens*	*SetSuppressHydrogens*

Constructors

```
OEDepictionFrom3DOptions(bool suppressH = true)
```

Default constructor that initializes an *OEDepictionFrom3DOptions* object with the following properties:

Table 2: **Default parameters of the OEDepictionFrom3DOptions class**

Property	Default value
inertial frame	false
max number of bond rotations	$2^{16} = 65536$
rotate to 2D reference	false
suppressing hydrogens	true

```
OEDepictionFrom3DOptions(const OEDepictionFrom3DOptions& rhs)
```

Copy constructor.

GetCoordsToInertialFrame

```
bool GetCoordsToInertialFrame() const
```

Returns whether the 3D molecule is moved to its inertial frame before generating the 2D coordinates.

See also

- *OEDepictionFrom3DOptions.SetCoordsToInertialFrame* method

GetMaxBondRotations

```
unsigned int GetMaxBondRotations() const
```

Returns the maximum number of bond rotations performed during the 2D coordinate generation process.

The default value for maximum number of bond rotations is $2^{16} = 65536$. This means that by default only the first 16 single bonds of the molecule will be rotated to find the 'closest' 2D layout to the original 3D conformation.

See also

- *OEDepictionFrom3DOptions.SetMaxBondRotations* method

GetRotateTo2DReference

```
bool GetRotateTo2DReference() const
```

Returns whether after the 2D coordinate generation a final rotation is performed to orient the 2D coordinates according to the 3D conformation.

See also

- *OEDepictionFrom3DOptions.SetRotateTo2DReference* method

GetSuppressHydrogens

```
bool GetSuppressHydrogens() const
```

Returns whether the explicit hydrogens are suppressed in the molecule prior to generating the 2D coordinates.

See also

- *OEDepictionFrom3DOptions.SetSuppressHydrogens* method

SetCoordsToInertialFrame

```
void SetCoordsToInertialFrame(bool)
```

Sets whether the 3D molecule is moved to its inertial frame before generating the 2D coordinates.

See also

- *OEDepictionFrom3DOptions.SetCoordsToInertialFrame* method

SetMaxBondRotations

```
void SetMaxBondRotations(unsigned int maxrotations)
```

Sets the maximum number of bond rotations performed during the 2D coordinate generation process.

maxrotations

The zero value means that there is no limit. Upon reaching this limit, the process terminates and returns the 2D coordinates that is the "closest" to the original 3D coordinates up to that point.

Hint

The default value for maximum number of bond rotations is $2^{16} = 65536$. This means that by default only the first 16 single bonds of the molecule will be rotated to find the 2D layout that is the 'closest' to the original 3D conformation.

It is recommended to use this parameter if the input 3D molecule has a large number of rotatable bonds.

See also

- *OEDepictionFrom3DOptions.GetMaxBondRotations* method

SetRotateTo2DReference

```
void SetRotateTo2DReference(bool)
```

Sets whether after the 2D coordinate generation a final rotation is performed to orient the 2D coordinates according to the 3D conformation. This can be used to ensure that 2D molecules are oriented similarly when they 3D conformations were similar.

See also

- *OEDepictionFrom3DOptions.GetRotateTo2DReference* method

SetSuppressHydrogens

```
void SetSuppressHydrogens(bool suppressH)
```

Sets whether the explicit hydrogens are suppressed in the molecule prior to generating the 2D coordinates. Only hydrogens that are necessary to faithfully represent tetrahedral stereochemistry will be kept.

See also

- *OEDepictionFrom3DOptions.GetSuppressHydrogens* method

4.1.22 OENearestResidue

```
class OENearestResidue
```

The *OENearestResidue* class that is used by *OEGetNearestResidue* and *OEGetNearbyResidue* functions. It stores an atom pointer, an associated distance and the display position of the atom.

Constructors

```
OENearestResidue(const OEChem::OEResidue& residue, double dist,
                 const OEDepict::OE2DPoint& pos)
```

Constructor.

GetDisplayPosition

```
const OEDepict::OE2DPoint& GetDisplayPosition() const
```

Returns the display position of the nearest residue.

GetDist

```
double GetDist() const
```

Returns the distance associated with the residue.

GetResidue

```
const OEChem::OEResidue& GetResidue() const
```

Returns the residue stored in the *OENearestResidue* object.

IsValid

```
bool IsValid() const
```

Returns whether the *OENearestResidue* object is initialized correctly.

4.1.23 OEPeptideDisplayOptions

```
class OEPeptideDisplayOptions
```

This class represents the *OEPeptideDisplayOptions* class that encapsulates properties that determine how a peptide is depicted.

See also

- *OEDrawPeptide* function

Code Example

- Depict Peptide **OpenEye Python Cookbook** recipe

Constructors

```
OEPeptideDisplayOptions()
```

Default constructor.

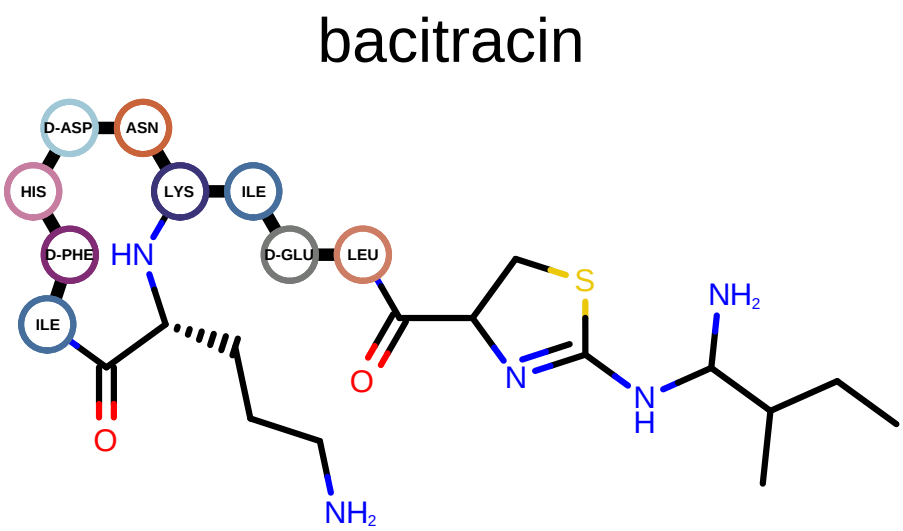


Fig. 33: **Peptide depiction with default options**

```
OEPeptideDisplayOptions(const OEPeptideDisplayOptions& rhs)
```

Copy constructor.

GetAminoAcidScale

```
double GetAminoAcidScale() const
```

Returns the scaling factor of amino-acid components in interactive mode.

See also

- *OEPeptideDisplayOptions.SetAminoAcidScale* method

GetInteractive

```
bool GetInteractive() const
```

Returns whether the generated `svg` image will be interactive.

See also

- *OEPeptideDisplayOptions.SetInteractive* method

GetLabelStyle

```
unsigned int GetLabelStyle() const
```

Warning

This is a deprecated API.

SetAminoAcidScale

```
void SetAminoAcidScale(double scale)
```

Sets the scaling factor of amino-acid components in interactive mode.

scale
: This value has to be in a range of [0.25, 0.75]. The default value is 0.50.

Note

This option has to be used along with the *OEPeptideDisplayOptions.SetInteractive* method

Example:

```
OEPeptideDisplayOptions opts = new OEPeptideDisplayOptions();
opts.SetInteractive(true);
opts.SetAminoAcidScale(0.75);
```

There is no image available for the PDF version of this book.

> **Note**
>
> This functionality is **only available** for `.svg` image format. In other image formats it has no effect.

SetInteractive

```
void SetInteractive(bool interactive)
```

Sets whether the generated `svg` image will be interactive. In interactive mode the amino acid components of the compact peptide representation are depicted on mouse over.

Example:

```
OEPeptideDisplayOptions opts = new OEPeptideDisplayOptions();
opts.SetInteractive(true);
```

There is no image available for the PDF version of this book.

> **Note**
>
> This functionality is **only available** for `.svg` image format. In other image formats it has no effect.

SetLabelStyle

```
void SetLabelStyle(unsigned int style)
```

> **Warning**
>
> This is a deprecated API.

style
: This value has to be from the *OEPeptideLabelStyle* namespace.

Example:

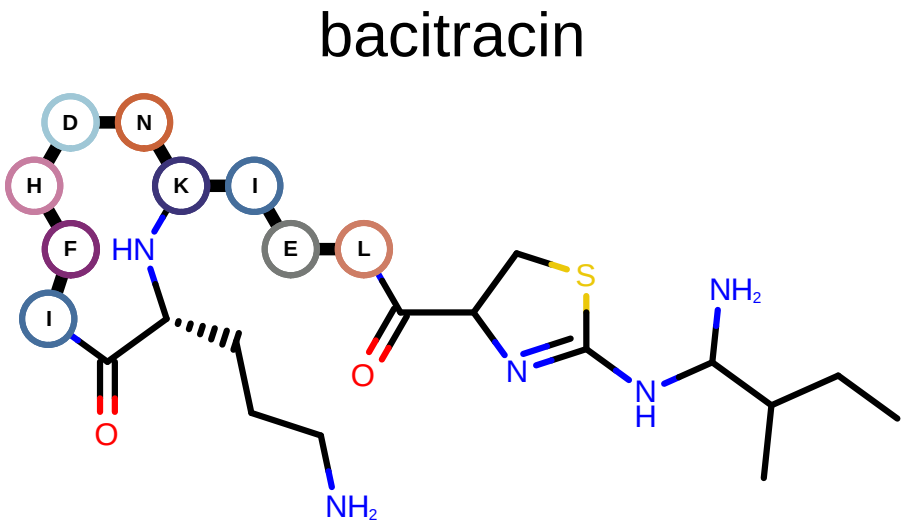


Fig. 34: **Peptide depiction with single letter style**

See also

- *OEPeptideLabelStyle* namespace

4.1.24 OEPlotMarker

```
class OEPlotMarker
```

This class represents the *OEPlotMarker* class that encapsulates properties that determine how a plot is depicted.

See also

- *OERamachandranPlot* class

Constructors

```
OEPlotMarker(const OEDepict::OEPen& pen,
             unsigned int style = OEPlotMarkerStyle::Circle,
             double scale = 1.0)
```

Creates and initializes an *OEPlotMarker* object with the given parameters.

pen
The OEPen object that is used to draw the plot marker.

style
The property that determine the style of the plot marker. This value has to be from the *OEPlotMarkerStyle* namespace.

scale
The multiplier that can be used to increase or to decrease the size of the plot market. This value has to be in the range of `[0.25, 4.0]`.

```
OEPlotMarker(const OEPlotMarker& rhs)
```

Copy constructor.

GetPen

```
const OEDepict::OEPen& GetPen() const
```

Returns the pen of the plot marker.

See also

- OEPen class

GetScale

```
double GetScale() const
```

Returns the multiplier that is used to increase or to decrease size of the plot marker.

GetStyle

```
unsigned int GetStyle() const
```

Returns the style of the plot marker. The return value is taken from the *OEPlotMarkerStyle* namespace.

See also

- *OEPlotMarkerStyle* namespace

4.1.25 OERamachandranPlot

```
class OERamachandranPlot
```

This class represents *OERamachandranPlot* that stores data of a Ramachandran plot.

There is no image available for the PDF version of this book.

See also

- *OERenderRamachandranPlot* function
- OERamachandranAnalysis class in **OEChem TK** manual

Code Example

- Ramachandran Plot **OpenEye Python Cookbook** recipe

Constructors

```
OERamachandranPlot()
```

Default constructor that creates an empty *OERamachandranPlot* object.

```
OERamachandranPlot(const OERamachandranPlot& rhs)
```

Copy constructor.

AddMolecule

```
bool AddMolecule(const OEChem::OEMolBase& mol)
```

Adds the molecule to the plot with default style.

mol
The macro-peptide of which backbone dihedral phi (ϕ) and psi (ψ) angle pairs are added to the plot. The residues of the molecule have to be perceived beforehand.

```
bool AddMolecule(const OEChem::OEMolBase& mol,
                 const OEPlotMarker& outmarker,
                 const OEPlotMarker& inmarker)
```

Adds the mol to the plot with the given style.

mol
The macro-peptide of which backbone dihedral phi (ϕ) and psi (ψ) angle pairs are added to the plot. The residues of the molecule have to be perceived beforehand.

outmarker
The style that is used to render data points with `OERamaCategory.Outlier` category.

inmarker
The style that is used to render data points with `OERamaCategory.Favored` or `OERamaCategory.Allowed` category.

AddRamachandran

```
bool AddRamachandran(const OEBio::OERamachandranAnalysis& rama,
                     const std::string& label,
                     const OEPlotMarker& marker)
```

Adds the data of the given OERamachandranAnalysis object to the plot with the given style and interactive label.

rama
The OERamachandranAnalysis object

label
The label that will be displayed in hover mode if the (ϕ, ψ) value pair stored in the OERamachandranAnalysis object is an outlier.

marker
The style that is used to render the data point on the plot.

See also

- OERamachandranAnalysis class

NumDataPoints

```
unsigned int NumDataPoints(unsigned int ramatype,
                           unsigned int category) const
```

Returns the number of data points stored in the *OERamachandranPlot* object with the given type and category.

ramatype
This value has to be from the `OERamaType` namespace.

category
This value has to be from the `OERamaCategory` namespace.

4.1.26 OEResidueSVGMarkupBase

```
class OEResidueSVGMarkupBase
```

OEResidueSVGMarkupBase is an abstract base class. Concrete class derived from this base class defines the markup of the residues in `.svg` images generated when calling the *OERenderActiveSite* function. Drawing elements representing residues in `.svg` image are grouped together in the following format in which the `<group id>` and `<class name>` strings can be defined in the derived concrete classes.

```
<g id='<group id>' class='<class name>'>
..
list of drawing elements
..
</g>
```

The following classes derive from this class:

- *OEResidueSVGNoMarkup*
- *OEResidueSVGStandardMarkup*

See also

- *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method
- Generating Interactive SVG Images chapter in **OEDepict TK** manual

CreateCopy

```
OEResidueSVGMarkupBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

GetClassName

```
std::string GetClassName(const OEChem::OEResidue &) const = 0
```

Returns the <class name> associated with a specific residue group in the .svg image. This is a virtual method that has to be implemented in the concrete derived classes.

GetGroupId

```
std::string GetGroupId(const OEChem::OEResidue &) const = 0
```

Returns the <group id> associated with a specific residue group in the .svg image. This is a virtual method that has to be implemented in the concrete derived classes.

4.1.27 OEResidueSVGNoMarkup

```
class OEResidueSVGNoMarkup : public OEResidueSVGMarkupBase
```

This class represents *OEResidueSVGNoMarkup* residue SVG markup functor. If this functor is passed to the *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method, the residues will be not marked in the svg image when calling the *OERenderActiveSite* function.

See also

- Generating Interactive SVG Images chapter in **OEDepict TK** manual
- *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method
- *OEResidueSVGStandardMarkup* functor

The following methods are publicly inherited from *OEResidueSVGMarkupBase*:

CreateCopy *GetClassName* *GetGroupId*

Constructors

```
OEResidueSVGNoMarkup()
```

Default constructor.

CreateCopy

```
OEResidueSVGMarkupBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEResidueSVGNoMarkup* object is dynamically allocated and owned by the caller.

GetClassName

```
std::string GetClassName(const OEChem::OEResidue& residue) const
```

Returns an empty string.

GetGroupId

```
std::string GetGroupId(const OEChem::OEResidue& residue) const
```

Returns an empty string.

4.1.28 OEResidueSVGStandardMarkup

```
class OEResidueSVGStandardMarkup : public OEResidueSVGMarkupBase
```

This class represents *OEResidueSVGStandardMarkup* residue SVG markup functor. If this functor is passed to the *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method, the residues will be in `svg` image when calling the *OERenderActiveSite* function.

See also

- Generating Interactive SVG Images chapter in **OEDepict TK** manual
- *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* method
- *OEResidueSVGNoMarkup* functor

The following methods are publicly inherited from *OEResidueSVGMarkupBase*:

CreateCopy	*GetClassName*	*GetGroupId*

Constructors

```
OEResidueSVGStandardMarkup(const std::string prefix = "",
                           const std::string classname = "residue",
                           const char separator = '-')
```

Creates an *OEResidueSVGStandardMarkup* object.

prefix
: The string that will be added at the beginning of each residue `group id`. For information about valid ids see `OEIsValidSVGGroupId` function. By default, no prefix will be added.

classname
: The `class name` that is going to be used to mark residues. For information about valid names see `OEIsValidSVGClassName` function.

separator
: The character used to separate the components (residue name, chain id, residue number) of the `group id`

By default, the following will be showed in `svg` image for marking a Leucine residue with chain id 'A' and residue number `715`:

```
<g id='LEU-A-715' class='residue'>
..
list of drawing elements
..
</g>
```

CreateCopy

```
OEResidueSVGMarkupBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEResidueSVGNoMarkup* object is dynamically allocated and owned by the caller.

GetClassName

```
std::string GetClassName(const OEChem::OEResidue& residue) const
```

Returns the string that specifies the `group id` of a residue in an `svg` image.

GetGroupId

```
std::string GetGroupId(const OEChem::OEResidue& residue) const
```

Returns the string that specifies the `class name` of a residue in an `svg` image.

4.1.29 OEShapeOverlapDisplay

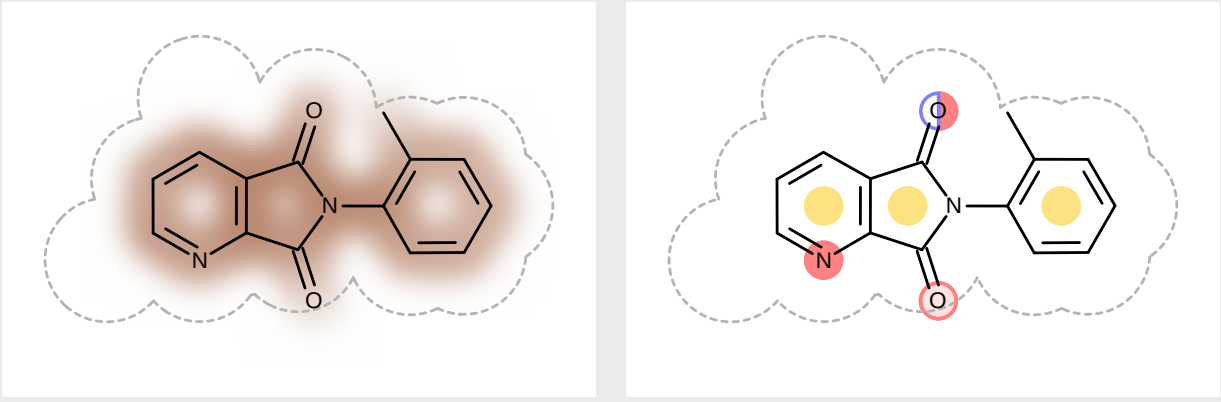
```
class OEShapeOverlapDisplay
```

This class represents *OEShapeOverlapDisplay* that stores the depiction information of depicting the for molecule of shape overlap. See example in *Table: Example of depicting the shape and the color overlap*.

See also

- The Theory chapter of the **Shape TK** manual.
- *OERenderColorOverlap* function
- *OERenderShapeOverlap* function

Table 3: **Example of depicting the shape and the color overlap**



Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

Constructors

```
OEShapeOverlapDisplay(const OEShapeQueryDisplay& refdisp,
                      const OEChem::OEMolBase& fitmol)
```

Constructs an *OEShapeOverlapDisplay* object with default depiction options.

refdisp
The *OEShapeQueryDisplay* object that stores the reference molecule of the shape overlap.

fitmol
The **3D** molecule that is overlaid to the reference molecule.

Warning

No overlay is performed to maximize the shape and color overlap between the reference and the fit molecule *i.e.* the *fitmol* has to be pre-aligned to the reference molecule stored in the *OEShapeQueryDisplay* object.

```
OEShapeOverlapDisplay(const OEShapeQueryDisplay& refdisp,
                      const OEChem::OEMolBase& fitmol,
                      const OEColorOverlapDisplayOptions& copts)
OEShapeOverlapDisplay(const OEShapeQueryDisplay& refdisp,
                      const OEChem::OEMolBase& fitmol,
                      const OEShapeOverlapDisplayOptions& sopts)
OEShapeOverlapDisplay(const OEShapeQueryDisplay& refdisp,
                      const OEChem::OEMolBase& fitmol,
                      const OEShapeOverlapDisplayOptions& sopts,
                      const OEColorOverlapDisplayOptions& copts)
```

Constructs an *OEShapeOverlapDisplay* object with given depiction options.

refdisp
The *OEShapeQueryDisplay* object that stores the reference molecule of the shape overlap.

fitmol
The **3D** molecule that is overlaid to the reference molecule.

copts
The *OEColorOverlapDisplayOptions* object that defines how the color atom overlap between the reference and the fit molecule is depicted.

sopts
The *OEShapeOverlapDisplayOptions* object that defines how the shape overlap between the reference and the fit molecule is depicted.

Warning

No overlay is performed to maximize the shape and color overlap between the reference and the fit molecule *i.e.* the *fitmol* has to be pre-aligned to the reference molecule stored in the *OEShapeQueryDisplay* object.

```
OEShapeOverlapDisplay(const OEShapeOverlapDisplay& rhs)
```

Copy constructor.

GetTitle

```
std::string GetTitle() const
```

Returns the title of the molecule the *OEShapeOverlapDisplay* object was initialized.

IsValid

```
bool IsValid() const
```

Returns whether the *OEShapeOverlapDisplay* object was initialized successfully. If initialization was attempted either with an invalid *OEShapeOverlapDisplay* object, an empty fit molecule or a fit molecule without 3D coordinates, then this method returns false.

4.1.30 OEShapeOverlapDisplayOptions

```
class OEShapeOverlapDisplayOptions : public OEDepict::OE2DMolDisplayOptions
```

This class represents *OEShapeOverlapDisplayOptions* that encapsulates properties that determine how the color atom overlap between the reference and the fit molecule is depicted by the *OERenderShapeOverlap* function.

See also

- *OEShapeOverlapDisplay* class
- *OERenderShapeOverlap* function

Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

The following methods are publicly inherited from OE2DMolDisplayOptions:

operator=	GetHydrogenStyle	SetBondLineAtomLabelGapScale
GetAromaticStyle	GetMargin	SetBondLineGapScale
GetAtomColor	GetScale	SetBondPropLabelFont
GetAtomColorStyle	GetSuperAtomLabelFont	SetBondPropLabelFontScale
GetAtomLabelFont	GetSuperAtomStyle	SetBondPropertyFunctor
GetAtomLabelFontScale	GetTitleFont	SetBondStereoStyle
GetAtomPropLabelFont	GetTitleFontScale	SetBondWidthScaling
GetAtomPropLabelFontScale	GetTitleHeight	SetDefaultBondPen
GetAtomPropertyFunctor	GetTitleLocation	SetDimensions
GetAtomStereoStyle	GetWidth	SetHeight
GetBackgroundColor	SetAromaticStyle	SetHydrogenStyle
GetBondColorStyle	SetAtomColor	SetMargin
GetBondLineAtomLabelGapScale	SetAtomColorStyle	SetMargins
GetBondLineGapScale	SetAtomLabelFont	SetScale
GetBondPropLabelFont	SetAtomLabelFontScale	SetSuperAtomLabelFont
GetBondPropLabelFontScale	SetAtomPropLabelFont	SetSuperAtomStyle
GetBondPropertyFunctor	SetAtomPropLabelFontScale	SetTitleFont
GetBondStereoStyle	SetAtomPropertyFunctor	SetTitleFontScale
GetBondWidthScaling	SetAtomStereoStyle	SetTitleHeight
GetDefaultBondPen	SetBackgroundColor	SetTitleLocation
GetHeight	SetBondColorStyle	SetWidth

Constructors

```
OEShapeOverlapDisplayOptions()
```

Default constructor.

```
OEShapeOverlapDisplayOptions(const OEShapeOverlapDisplayOptions& rhs)
```

Copy constructors.

ClearQuerySurfaceArcFxn

```
void ClearQuerySurfaceArcFxn()
```

Removes the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule.

See also

- *OEShapeOverlapDisplayOptions.GetQuerySurfaceArcFxn* method
- *OEShapeOverlapDisplayOptions.SetQuerySurfaceArcFxn* method

GetClearBackground

```
bool GetClearBackground() const
```

Returns whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the shape overlap.

See also

- *OEShapeOverlapDisplayOptions.SetClearBackground* method

GetOverlapColor

```
const OESystem::OEColor& GetOverlapColor() const
```

Returns the color of the property map that is used to visualize shape overlap between the reference and the fit molecule.

See also

- *OEShapeOverlapDisplayOptions.SetOverlapColor* method

GetOverlapDisplayStyle

```
unsigned int GetOverlapDisplayStyle() const
```

Returns the style that defines how shape overlay is depicted by the *OERenderShapeOverlap* function. The return value is taken from the *OEShapeOverlapDisplayStyle* namespace.

See also

- *OEShapeOverlapDisplayOptions.SetOverlapDisplayStyle* method
- *OEShapeOverlapDisplayStyle* namespace

GetQuerySurfaceArcFxn

```
const OESurfaceArcFxnBase* GetQuerySurfaceArcFxn() const
```

Returns the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule.

See also

- *OEShapeOverlapDisplayOptions.ClearQuerySurfaceArcFxn* method
- *OEShapeOverlapDisplayOptions.SetQuerySurfaceArcFxn* method

SetClearBackground

```
void SetClearBackground(bool clear)
```

Sets whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the shape overlap.

See also

- *OEShapeOverlapDisplayOptions.GetClearBackground* method

SetOverlapColor

```
void SetOverlapColor(const OESystem::OEColor& color)
```

Sets the color of the property map that is used to visualize shape overlap between the reference and the fit molecule.

See also

- *OEShapeOverlapDisplayOptions.GetOverlapColor* method

SetOverlapDisplayStyle

```
void SetOverlapDisplayStyle(unsigned int style)
```

Sets the style that defines how shape overlay is depicted by the *OERenderShapeOverlap* function.

style
This value has to be from the *OEShapeOverlapDisplayStyle* namespace.

See also

- *OEShapeOverlapDisplayOptions.GetOverlapDisplayStyle* method
- *OEShapeOverlapDisplayStyle* namespace

SetQuerySurfaceArcFxn

```
void SetQuerySurfaceArcFxn(const OESurfaceArcFxnBase& arcfxn)
```

Sets the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule around the fit molecule when calling the *OERenderShapeOverlap* function to depict the shape overlap.

See also

- *OEShapeOverlapDisplayOptions.ClearQuerySurfaceArcFxn* method
- *OEShapeOverlapDisplayOptions.GetQuerySurfaceArcFxn* method

4.1.31 OEShapeQueryDisplay

```
class OEShapeQueryDisplay
```

This class represents *OEShapeQueryDisplay* that stores the depiction information of a shape query *i.e.* shape reference molecule. See example in *Figure: Example of depicting a shape query*.

See also

- The Theory chapter of the **Shape TK** manual.
- *OERenderShapeQuery* function

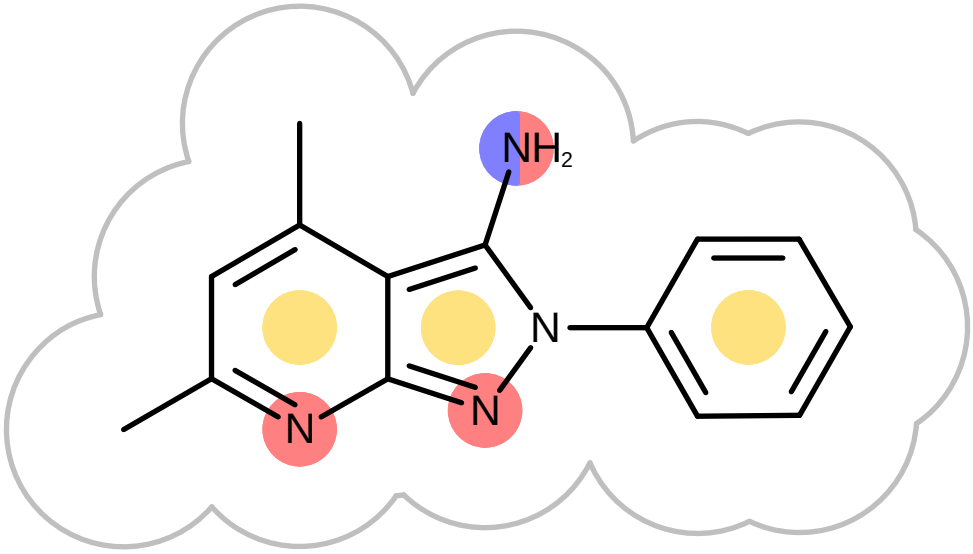


Fig. 35: **Example of depicting a shape query**

Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

Constructors

```
OEShapeQueryDisplay(const OEChem::OEMolBase& refmol,
                    const OEShape::OEColorForceField& cff)
```

Constructs an *OEShapeQueryDisplay* object with default depiction options.

refmol
The **3D** reference molecule of the shape overlap.

cff
The OEColorForceField object that defines various chemical centers and their interactions.

```
OEShapeQueryDisplay(const OEChem::OEMolBase& refmol,
                    const OEShape::OEColorForceField& cff,
                    const OEShapeQueryDisplayOptions& opts)
```

Constructs an *OEShapeQueryDisplay* object with given depiction options.

refmol
The **3D** reference molecule of the shape overlap.

cff
The OEColorForceField object that defines various chemical centers and their interactions.

opts
The *OEShapeQueryDisplayOptions* object that defines how the shape query *i.e.* reference molecule is depicted.

```
OEShapeQueryDisplay(const OEChem::OEMolBase& refmol,
                    const OEColorForceFieldDisplay& cffdisp)
```

Constructs an *OEShapeQueryDisplay* object with default depiction options.

refmol
The **3D** reference molecule of the shape overlap.

cffdisp
The *OEColorForceFieldDisplay* object that stores the depiction information of an OEColorForceField object.

```
OEShapeQueryDisplay(const OEChem::OEMolBase& refmol,
                    const OEColorForceFieldDisplay& cffdisp,
                    const OEShapeQueryDisplayOptions& opts)
```

Constructs an *OEShapeQueryDisplay* object with given depiction options.

refmol
The **3D** reference molecule of the shape overlap.

cffdisp
The *OEColorForceFieldDisplay* object that stores the depiction information of an OEColorForceField object.

opts
The *OEShapeQueryDisplayOptions* object that defines how the shape query *i.e.* reference molecule is depicted.

```
OEShapeQueryDisplay(const OEShapeQueryDisplay& rhs)
```

Copy constructor.

GetTitle

```
std::string GetTitle() const
```

Returns the title of the molecule the *OEShapeQueryDisplay* object was initialized.

IsValid

```
bool IsValid() const
```

Returns whether the *OEShapeQueryDisplay* object was initialized successfully. If initialization was attempted with either an empty molecule or a molecule without 3D coordinates, then this method returns false.

4.1.32 OEShapeQueryDisplayOptions

```
class OEShapeQueryDisplayOptions : public OEDepict::OE2DMolDisplayOptions
```

This class represents *OEShapeQueryDisplayOptions* that encapsulates properties that determine how the reference molecule of a shape overlap is depicted by the *OERenderShapeQuery* function.

See also

- *OEShapeQueryDisplay* class
- *OERenderShapeQuery* function

Code Example

- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

The following methods are publicly inherited from OE2DMolDisplayOptions:

operator=	GetHydrogenStyle	SetBondLineAtomLabelGapScale
GetAromaticStyle	GetMargin	SetBondLineGapScale
GetAtomColor	GetScale	SetBondPropLabelFont
GetAtomColorStyle	GetSuperAtomLabelFont	SetBondPropLabelFontScale
GetAtomLabelFont	GetSuperAtomStyle	SetBondPropertyFunctor
GetAtomLabelFontScale	GetTitleFont	SetBondStereoStyle
GetAtomPropLabelFont	GetTitleFontScale	SetBondWidthScaling
GetAtomPropLabelFontScale	GetTitleHeight	SetDefaultBondPen
GetAtomPropertyFunctor	GetTitleLocation	SetDimensions
GetAtomStereoStyle	GetWidth	SetHeight
GetBackgroundColor	SetAromaticStyle	SetHydrogenStyle
GetBondColorStyle	SetAtomColor	SetMargin
GetBondLineAtomLabelGapScale	SetAtomColorStyle	SetMargins
GetBondLineGapScale	SetAtomLabelFont	SetScale
GetBondPropLabelFont	SetAtomLabelFontScale	SetSuperAtomLabelFont
GetBondPropLabelFontScale	SetAtomPropLabelFont	SetSuperAtomStyle
GetBondPropertyFunctor	SetAtomPropLabelFontScale	SetTitleFont
GetBondStereoStyle	SetAtomPropertyFunctor	SetTitleFontScale
GetBondWidthScaling	SetAtomStereoStyle	SetTitleHeight
GetDefaultBondPen	SetBackgroundColor	SetTitleLocation
GetHeight	SetBondColorStyle	SetWidth

Constructors

```
OEShapeQueryDisplayOptions()
```

Default constructor.

```
OEShapeQueryDisplayOptions(const OEShapeQueryDisplayOptions& rhs)
```

Copy constructors.

ClearSurfaceArcFxn

```
void ClearSurfaceArcFxn()
```

Removes the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule.

See also

- *OEShapeQueryDisplayOptions.GetSurfaceArcFxn*
- *OEShapeQueryDisplayOptions.SetSurfaceArcFxn*

GetClearBackground

```
bool GetClearBackground() const
```

Returns whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the shape query.

See also

- *OEShapeQueryDisplayOptions.SetClearBackground*

GetColorAtomStyle

```
unsigned int GetColorAtomStyle() const
```

Returns the style that is used to define how color atoms are depicted by the *OERenderShapeQuery* function. The return value is taken from the *OEColorAtomStyle* namespace.

See also

- *OEShapeQueryDisplayOptions.SetColorAtomStyle*
- *OEColorAtomStyle* namespace

GetDepictOrientation

```
unsigned int GetDepictOrientation() const
```

Returns the preferred orientation of 2D layout if the shape query. The return value is taken from the `OEDepictOrientation` namespace.

See also

- *OEShapeQueryDisplayOptions.SetDepictOrientation* method

GetSurfaceArcFxn

```
const OESurfaceArcFxnBase* GetSurfaceArcFxn() const
```

Returns the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule.

See also

- *OEShapeQueryDisplayOptions.ClearSurfaceArcFxn*
- *OEShapeQueryDisplayOptions.SetSurfaceArcFxn*

SetClearBackground

```
void SetClearBackground(bool clear)
```

Sets whether or not the image is cleared (by calling `OEImageBase.Clear` method) prior to the rendering of the shape query.

See also

- *OEShapeQueryDisplayOptions.GetClearBackground*

SetColorAtomStyle

```
void SetColorAtomStyle(unsigned int style)
```

Sets the style that defines how color atoms are depicted by the *OERenderShapeQuery* function.

style
This value has to be from the *OEColorAtomStyle* namespace.

See also

- *OEShapeQueryDisplayOptions.GetColorAtomStyle*
- *OEColorAtomStyle* namespace

SetDepictOrientation

```
void SetDepictOrientation(unsigned int orientation)
```

Sets the preferred orientation of 2D layout of the shape query.

orientation
: This value has to be from the `OEDepictOrientation` namespace.

Note

Changing the orientation of the layout of the shape query consequently effects the orientation of the shape and color overlaps mapped to the given shape query.

See also

- *`OEShapeQueryDisplayOptions.GetDepictOrientation`* method
- `OEDepictOrientation` namespace

SetSurfaceArcFxn

```
void SetSurfaceArcFxn(const OESurfaceArcFxnBase& arcfxn)
```

Sets the surface drawing functor that is drawn to represent the 2D molecule surface of the reference molecule when calling the *`OERenderShapeQuery`* function.

See also

- *OEShapeQueryDisplayOptions.ClearSurfaceArcFxn*
- *OEShapeQueryDisplayOptions.GetSurfaceArcFxn*

4.1.33 OESurfaceArc

```
class OESurfaceArc
```

This class represents *OESurfaceArc* that stores data for drawing the arcs of molecule surface. See *Figure: Example of the data stored in the OESurfaceArc class*.

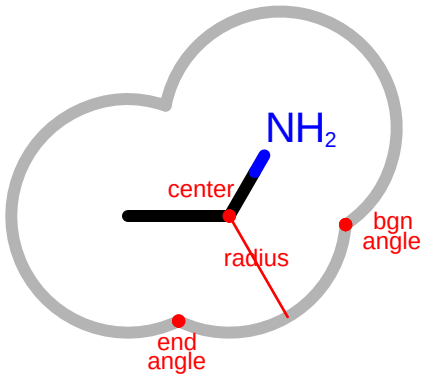


Fig. 36: **Example of the data stored in the OESurfaceArc class**

Note

Angles are interpreted such that 0.0° and 360.0° degrees are at the *12 o'clock* position, 90.0° degree corresponds to *3 o'clock*, etc.

See also

- *OESurfaceArcFxnBase* class
- *OEGet2DSurfaceArcs* function

Constructors

```
OESurfaceArc()
```

Default constructor.

```
OESurfaceArc(const OEDepict::OE2DAtomDisplay* adisp,
             const OEDepict::OE2DPoint& center, double bgnAngle,
             double endAngle, double radius)
```

Creates an *OESurfaceArc* object with the given parameters.

```
OESurfaceArc(const OESurfaceArc& rhs)
```

Copy constructor.

GetAtomDisplay

```
const OEDepict::OE2DAtomDisplay* GetAtomDisplay() const
```

Returns the pointer of the OE2DAtomDisplay object the given arc belongs to.

See also

- OE2DAtomDisplay class in the **OEDepict TK** manual

GetBgnAngle

```
double GetBgnAngle() const
```

Returns the angle (in degrees) where the arc of the surface starts.

GetCenter

```
const OEDepict::OE2DPoint& GetCenter() const
```

Returns the (x,y) coordinates of the center of the surface arc.

See also

- OE2DPoint class in the **OEDepict TK** manual

GetEndAngle

```
double GetEndAngle() const
```

Returns the angle (in degrees) where the arc of the surface ends.

GetRadius

```
double GetRadius() const
```

Returns the radius of the arc.

SetAtomDisplay

```
bool SetAtomDisplay(const OEDepict::OE2DAtomDisplay* adisp)
```

Sets the atom display to which the arc belongs.

See also

- OE2DAtomDisplay class in the OEDepict manual

SetBgnAngle

```
bool SetBgnAngle(double angle)
```

Sets the angle where the arc of the surface begins.

angle
The angle has to in the range of $[0.0°, 360.0°]$

SetCenter

```
bool SetCenter(const OEDepict::OE2DPoint& center)
```

Sets the (x,y) coordinates of the center of the surface arc.

SetEndAngle

```
bool SetEndAngle(double angle)
```

Sets the angle where the arc of the surface ends.

angle
The angle has to in the range of $[0.0°, 360.0°]$

SetRadius

```
bool SetRadius(double radius)
```

Sets the radius of the arc.

4.2 OEGrapheme Basic Arc Drawing Classes

4.2.1 OEAlphaRainbowArcFxn

```
class OEAlphaRainbowArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEAlphaRainbowArcFxn*. See example in *Figure: Example of drawing molecule surface using OEAlphaRainbowArcFxn*.

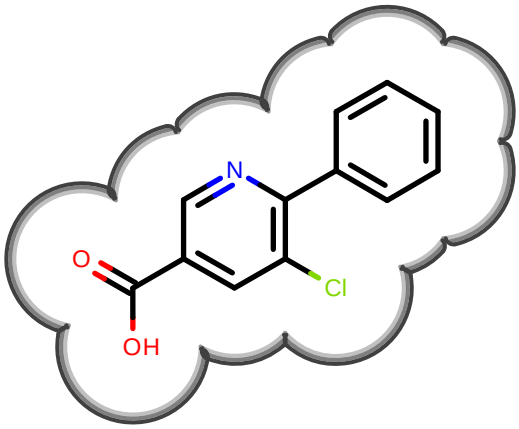


Fig. 37: **Example of drawing molecule surface using OEAlphaRainbowArcFxn**

See also

- *OEDrawAlphaRainbowSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEAlphaRainbowArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.AlphaRainbow* style.

See example in *Figure: Example of drawing molecule surface using OEAlphaRainbowArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEAlphaRainbowArcFxn* object is dynamically allocated and owned by the caller.

4.2.2 OEBrickRoadArcFxn

```
class OEBrickRoadArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEBrickRoadArcFxn*. See example in *Figure: Example of drawing molecule surface using OEBrickRoadArcFxn*.

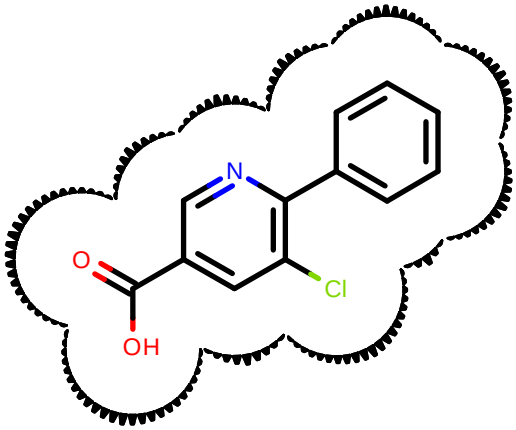


Fig. 38: **Example of drawing molecule surface using OEBrickRoadArcFxn**

See also

- *OEDrawBrickRoadSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEBrickRoadArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.BrickRoad* style.

See example in *Figure: Example of drawing molecule surface using OEBrickRoadArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEBrickRoadArcFxn* object is dynamically allocated and owned by the caller.

4.2.3 OECastleArcFxn

```
class OECastleArcFxn : public OESurfaceArcFxnBase
```

This class represents *OECastleArcFxn*. See example in *Figure: Example of drawing molecule surface using OECastleArcFxn*.

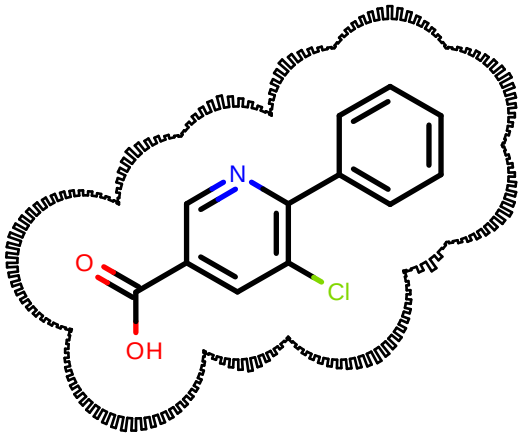


Fig. 39: **Example of drawing molecule surface using OECastleArcFxn**

See also

- *OEDrawCastleSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OECastleArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Castle* style.

See example in *Figure: Example of drawing molecule surface using OECastleArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OECastleArcFxn* object is dynamically allocated and owned by the caller.

4.2.4 OECogArcFxn

```
class OECogArcFxn : public OESurfaceArcFxnBase
```

This class represents *OECogArcFxn*. See example in *Figure: Example of drawing molecule surface using OECogArcFxn*.

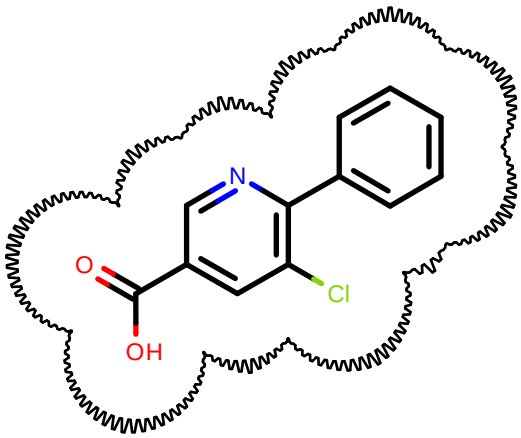


Fig. 40: **Example of drawing molecule surface using OECogArcFxn**

See also

- *OEDrawCogSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OECogArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Cog* style.

See example in *Figure: Example of drawing molecule surface using OECogArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OECogArcFxn* object is dynamically allocated and owned by the caller.

4.2.5 OEComplexSurfaceArcFxnBase

```
class OEComplexSurfaceArcFxnBase : public OESurfaceArcFxnBase
```

This class represents *OEComplexSurfaceArcFxnBase* which is an abstract class that defines the interface used for drawing molecule surfaces. The *OEComplexSurfaceArcFxnBase* class also stores a *depth* parameter that reflects the relative distance between the ligand and the receptor when drawing the complex surface.

See also

Drawing a Ligand-Protein Complex Surface chapter

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

The following classes derive from this class:

- *OEDefaultCavityArcFxn*

- *OEDefaultVoidArcFxn*

Constructors

```
OEComplexSurfaceArcFxnBase()
OEComplexSurfaceArcFxnBase(double depth)
OEComplexSurfaceArcFxnBase(const OEComplexSurfaceArcFxnBase& rhs)
```

Default and copy constructors.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const = 0
```

It is a virtual const method that is implemented in the concrete derived classes to draw surface arcs.

image
: The image on which the surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

CreateComplexCopy

```
OEComplexSurfaceArcFxnBase* CreateComplexCopy() const
```

Virtual const constructor which allows copying of concrete derived objects using a reference to the *OEComplexSurfaceArcFxnBase* base class.

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to the *OESurfaceArcFxnBase* base class.

GetDepth

```
double GetDepth() const
```

Returns the *depth* parameter of the surface arc.

SetDepth

```
void SetDepth(double depth)
```

Sets the *depth* parameter of the surface arc.

4.2.6 OEDefaultArcFxn

```
class OEDefaultArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEDefaultArcFxn*. See example in *Figure: Example of drawing molecule surface using OEDefaultArcFxn*.

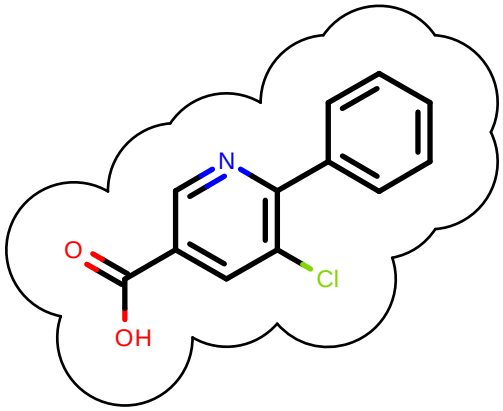


Fig. 41: **Example of drawing molecule surface using OEDefaultArcFxn**

See also

- *OEDrawDefaultSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEDefaultArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Default* style.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEDefaultArcFxn* object is dynamically allocated and owned by the caller.

4.2.7 OEDefaultBuriedArcFxn

```
class OEDefaultBuriedArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEDefaultBuriedArcFxn* functor that is used to draw the arc segment of the 2D molecule surface, where the ligand and the protein tightly fit in the 3D complex. See example in *Figure: Example of drawing an arc using OEDefaultBuriedArcFxn*.

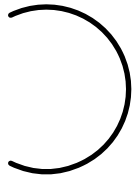


Fig. 42: **Example of drawing an arc using OEDefaultBuriedArcFxn**

See also

- *Drawing a Ligand-Protein Complex Surface* chapter
- *OEAddComplexSurfaceArcs* function
- *OEDefaultBuriedArcFxn* class
- *OEDefaultCavityArcFxn* class
- *OEDefaultSolventArcFxn* class

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEDefaultBuriedArcFxn()
```

Default constructor.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& a) const
```

Draws a surface arc with the *OESurfaceArcStyle.Default* style using a black pen.

image
The image on which the *buried* surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OEAddComplexSurfaceArcs* function
- *OEDrawEyelashSurfaceArc* function

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEDefaultBuriedArcFxn* object is dynamically allocated and owned by the caller.

4.2.8 OEDefaultCavityArcFxn

```
class OEDefaultCavityArcFxn : public OEComplexSurfaceArcFxnBase
```

This class represents *OEDefaultCavityArcFxn* functor that is used to draw the arc segment of the 2D molecule surface, where there is a cavity detected between the ligand and the protein in the 3D complex. See example in *Figure: Example of drawing an arc using OEDefaultCavityArcFxn*.

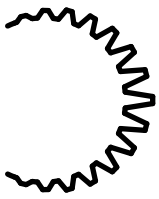


Fig. 43: **Example of drawing an arc using OEDefaultCavityArcFxn**

See also

- *Drawing a Ligand-Protein Complex Surface* chapter
- *OEAddComplexSurfaceArcs* function
- *OEDefaultBuriedArcFxn* class
- *OEDefaultSolventArcFxn* class
- *OEDefaultVoidArcFxn* class

The following methods are publicly inherited from *OEComplexSurfaceArcFxnBase*:

operator=	*CreateComplexCopy*	*GetDepth*
operator()	*CreateCopy*	*SetDepth*

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEDefaultCavityArcFxn()
```

Default constructors.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& a) const
```

Draws a surface arc with the *OESurfaceArcStyle.Cog* style using a black pen.

image
: The image on which the *cavity* surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OEAddComplexSurfaceArcs* function
- *OEDrawCogSurfaceArc* function

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEDefaultCavityArcFxn* object is dynamically allocated and owned by the caller.

4.2.9 OEDefaultSolventArcFxn

```
class OEDefaultSolventArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEDefaultSolventArcFxn* functor that is used to draw the arc segment of the 2D molecule surface, where the ligand is exposed to the solvent in the 3D complex. See example in *Figure: Example of drawing an arc using OEDefaultSolventArcFxn*.



Fig. 44: **Example of drawing an arc using OEDefaultSolventArcFxn**

See also

- *Drawing a Ligand-Protein Complex Surface* chapter
- *OEAddComplexSurfaceArcs* function
- *OEDefaultBuriedArcFxn* class
- *OEDefaultCavityArcFxn* class
- *OEDefaultVoidArcFxn* class

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEDefaultSolventArcFxn()
```

Default constructor.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& a) const
```

Draws a surface arc with the *OESurfaceArcStyle.Default* style using a grey pen.

image
: The image on which the *solvent* surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OEAddComplexSurfaceArcs* function
- *OEDrawDefaultSurfaceArc* function

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEDefaultSolventArcFxn* object is dynamically allocated and owned by the caller.

4.2.10 OEDefaultVoidArcFxn

```
class OEDefaultVoidArcFxn : public OEComplexSurfaceArcFxnBase
```

This class represents *OEDefaultVoidArcFxn* functor that is used to draw the arc segment of the 2D molecule surface, where there is a small gap detected between the ligand and the protein in the 3D complex. See example in *Figure: Example of drawing an arc using OEDefaultVoidArcFxn*.

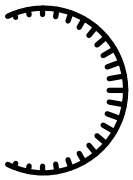


Fig. 45: **Example of drawing an arc using OEDefaultVoidArcFxn**

See also

- *Drawing a Ligand-Protein Complex Surface* chapter
- *OEAddComplexSurfaceArcs* function
- *OEDefaultBuriedArcFxn* class
- *OEDefaultCavityArcFxn* class
- *OEDefaultSolventArcFxn* class

The following methods are publicly inherited from *OEComplexSurfaceArcFxnBase*:

operator=	*CreateComplexCopy*	*GetDepth*
operator()	*CreateCopy*	*SetDepth*

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEDefaultVoidArcFxn()
```

Default constructor.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Eyelash* style using a black pen.

image
: The image on which the *void* surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OEAddComplexSurfaceArcs* function
- *OEDrawEyelashSurfaceArc* function

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEDefaultVoidArcFxn* object is dynamically allocated and owned by the caller.

4.2.11 OEEyelashArcFxn

```
class OEEyelashArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEEyelashArcFxn*. See example in *Example of drawing molecule surface using OEEyelashArcFxn*.

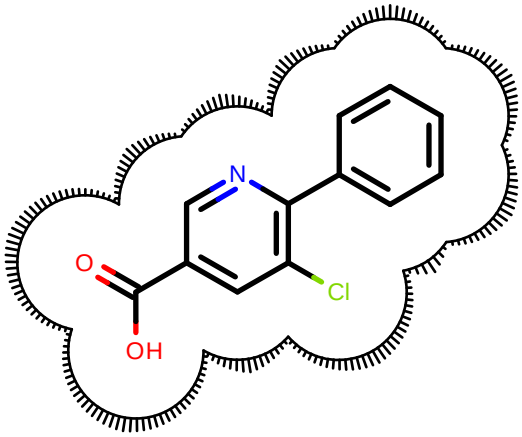


Fig. 46: **Example of drawing molecule surface using OEEyelashArcFxn**

See also

- *OEDrawEyelashSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEEyelashArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Eyelash* style.

See example in *Example of drawing molecule surface using OEEyelashArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEEyelashArcFxn* object is dynamically allocated and owned by the caller.

4.2.12 OEFlowerArcFxn

```
class OEFlowerArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEFlowerArcFxn*. See example in *Example of drawing molecule surface using OEFlowerArcFxn*.

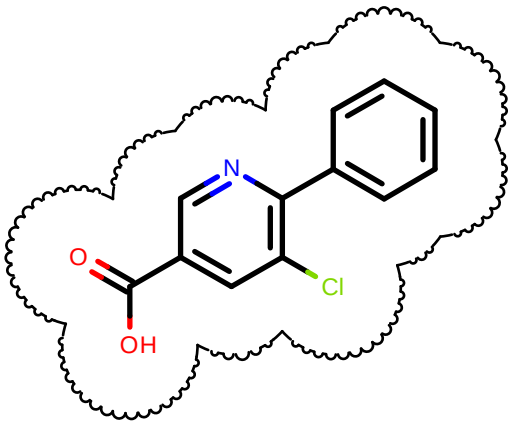


Fig. 47: **Example of drawing molecule surface using OEFlowerArcFxn**

See also

- *OEDrawFlowerSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEFlowerArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
: The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Flower* style.

See example in *Example of drawing molecule surface using OEFlowerArcFxn*.

image
: The image on which the surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEFlowerArcFxn* object is dynamically allocated and owned by the caller.

4.2.13 OENecklaceArcFxn

```
class OENecklaceArcFxn : public OESurfaceArcFxnBase
```

This class represents *OENecklaceArcFxn*. See example in *Example of drawing molecule surface using OENecklaceArcFxn*.

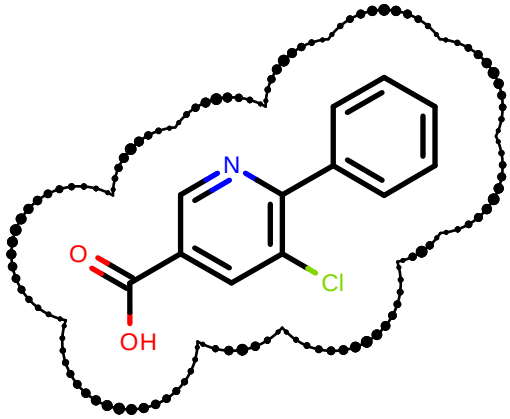


Fig. 48: **Example of drawing molecule surface using OENecklaceArcFxn**

See also

- *OEDrawNecklaceSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OENecklaceArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Necklace* style.

See example in *Example of drawing molecule surface using OENecklaceArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OENecklaceArcFxn* object is dynamically allocated and owned by the caller.

4.2.14 OENullArcFxn

```
class OENullArcFxn : public OESurfaceArcFxnBase
```

This class represents *OENullArcFxn*.

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

operator()

```
bool operator()(OEDepict::OEImageBase &, const OESurfaceArc &) const
```

No-op *i.e.* no surface arc is drawn.

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OENullArcFxn* object is dynamically allocated and owned by the caller.

4.2.15 OEOliveBranchArcFxn

```
class OEOliveBranchArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEOliveBranchArcFxn*. See example in *Example of drawing molecule surface using OEOliveBranchArcFxn*.

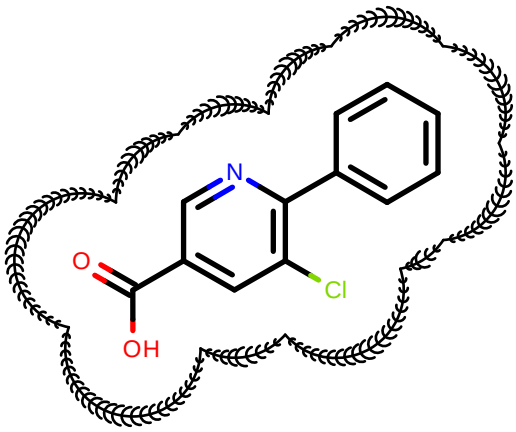


Fig. 49: **Example of drawing molecule surface using OEOliveBranchArcFxn**

See also

- *OEDrawOliveBranchSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEOliveBranchArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.OliveBranch* style.

See example in *Example of drawing molecule surface using OEOliveBranchArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEOliveBranchArcFxn* object is dynamically allocated and owned by the caller.

4.2.16 OEPearlsArcFxn

```
class OEPearlsArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEPearlsArcFxn*. See example in *Example of drawing molecule surface using OEPearlsArcFxn*.

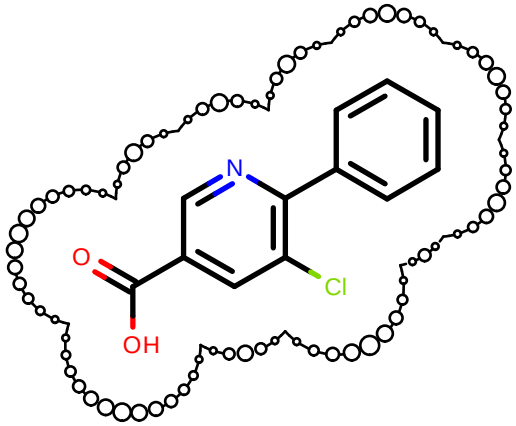


Fig. 50: **Example of drawing molecule surface using OEPearlsArcFxn**

See also

- *OEDrawPearlsSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEPearlsArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Pearls* style.

See example in *Example of drawing molecule surface using OEPearlsArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEPearlsArcFxn* object is dynamically allocated and owned by the caller.

4.2.17 OERaceTrackArcFxn

```
class OERaceTrackArcFxn : public OESurfaceArcFxnBase
```

This class represents *OERaceTrackArcFxn*. See example in *Example of drawing molecule surface using OERaceTrackArcFxn*.

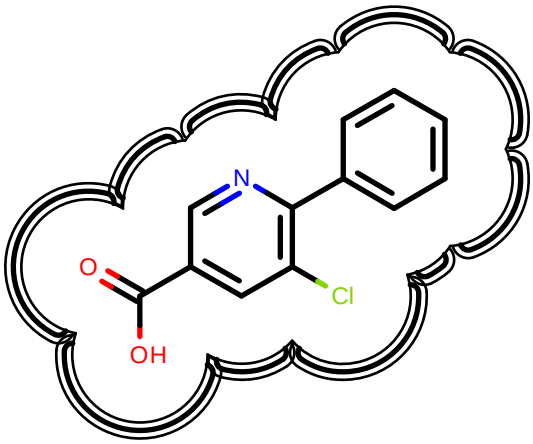


Fig. 51: **Example of drawing molecule surface using OERaceTrackArcFxn**

See also

- *OEDrawRaceTrackSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OERaceTrackArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.RaceTrack* style.

See example in *Example of drawing molecule surface using OERaceTrackArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OERaceTrackArcFxn* object is dynamically allocated and owned by the caller.

4.2.18 OERailroadTrackArcFxn

```
class OERailroadTrackArcFxn : public OESurfaceArcFxnBase
```

This class represents *OERailroadTrackArcFxn*. See example in *Example of drawing molecule surface using OERailroadTrackArcFxn*.

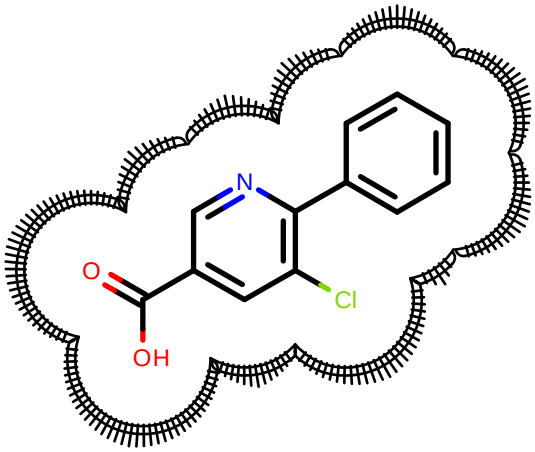


Fig. 52: **Example of drawing molecule surface using OERailroadTrackArcFxn**

See also

- *OEDrawRailroadTrackSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OERailroadTrackArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
: The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.RailroadTrack* style.

See example in *Example of drawing molecule surface using OERailroadTrackArcFxn*.

image
: The image on which the surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OERailroadTrackArcFxn* object is dynamically allocated and owned by the caller.

4.2.19 OESawArcFxn

```
class OESawArcFxn : public OESurfaceArcFxnBase
```

This class represents *OESawArcFxn*. See example in *Example of drawing molecule surface using OESawArcFxn*.

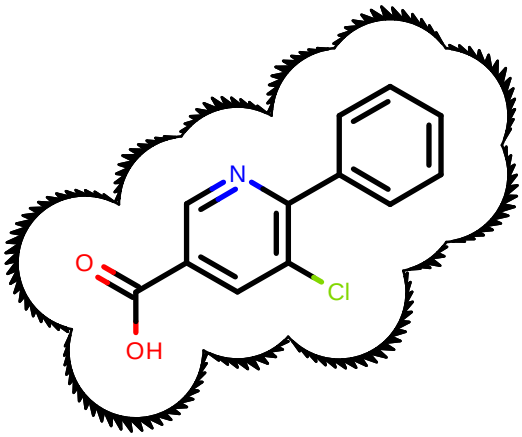


Fig. 53: **Example of drawing molecule surface using OESawArcFxn**

See also

- *OEDrawSawSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OESawArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Saw* style.

See example in *Example of drawing molecule surface using OESawArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OESawArcFxn* object is dynamically allocated and owned by the caller.

4.2.20 OESimpsonArcFxn

```
class OESimpsonArcFxn : public OESurfaceArcFxnBase
```

This class represents *OESimpsonArcFxn*. See example in *Example of drawing molecule surface using OESimpsonArcFxn*.

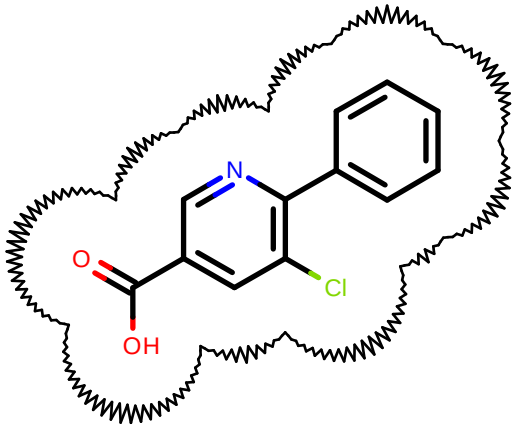


Fig. 54: **Example of drawing molecule surface using OESimpsonArcFxn**

See also

- *OEDrawSimpsonSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OESimpsonArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Simpson* style.

See example in *Example of drawing molecule surface using OESimpsonArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OESimpsonArcFxn* object is dynamically allocated and owned by the caller.

4.2.21 OEStitchArcFxn

```
class OEStitchArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEStitchArcFxn*. See example in *Example of drawing molecule surface using OEStitchArcFxn*.

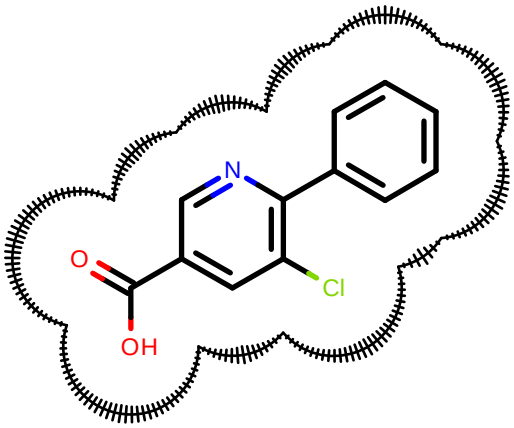


Fig. 55: **Example of drawing molecule surface using OEStitchArcFxn**

See also

- *OEDrawStitchSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEStitchArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
: The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Stitch* style.

See example in *Example of drawing molecule surface using OEStitchArcFxn*.

image
: The image on which the surface arc is drawn.

arc
: The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEStitchArcFxn* object is dynamically allocated and owned by the caller.

4.2.22 OESunArcFxn

```
class OESunArcFxn : public OESurfaceArcFxnBase
```

This class represents *OESunArcFxn*. See example in *Example of drawing molecule surface using OESunArcFxn*.

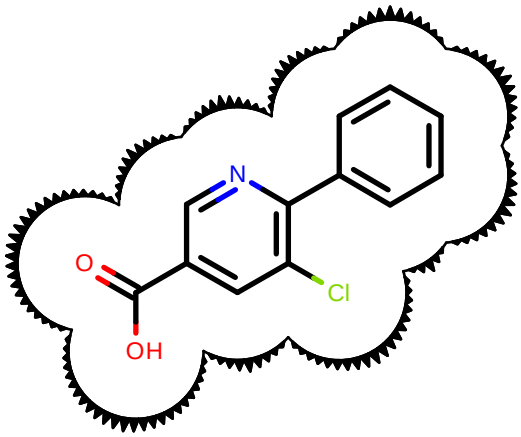


Fig. 56: **Example of drawing molecule surface using OESunArcFxn**

See also

- *OEDrawSunSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OESunArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Sun* style.

See example in *Example of drawing molecule surface using OESunArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OESunArcFxn* object is dynamically allocated and owned by the caller.

4.2.23 OESurfaceArcFxnBase

```
class OESurfaceArcFxnBase
```

This class represents *OESurfaceArcFxnBase* which is an abstract class that defines the interface used for drawing molecule surfaces.

The following classes derive from this class:

- *OEAlphaRainbowArcFxn*
- *OEBrickRoadArcFxn*
- *OECastleArcFxn*
- *OECogArcFxn*
- *OEDefaultArcFxn*
- *OEEyelashArcFxn*
- *OEFlowerArcFxn*
- *OENecklaceArcFxn*

- *OENullArcFxn*
- *OEOliveBranchArcFxn*
- *OEPearlsArcFxn*
- *OERaceTrackArcFxn*
- *OERailroadTrackArcFxn*
- *OESawArcFxn*
- *OESimpsonArcFxn*
- *OEStitchArcFxn*
- *OESunArcFxn*
- *OEWreathArcFxn*

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const = 0
```

It is a virtual const method that is implemented in the concrete derived classes to draw surface arcs.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

4.2.24 OEWreathArcFxn

```
class OEWreathArcFxn : public OESurfaceArcFxnBase
```

This class represents *OEWreathArcFxn*. See example in *Example of drawing molecule surface using OEWreathArcFxn*.

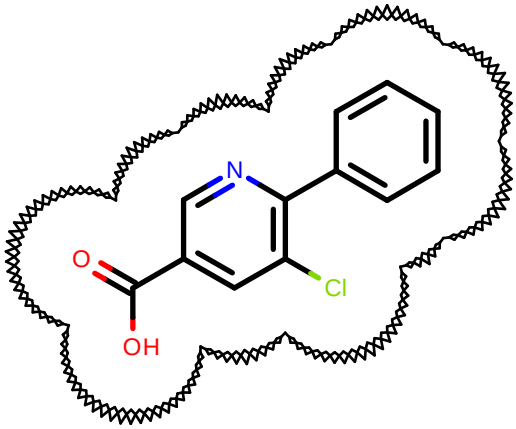


Fig. 57: **Example of drawing molecule surface using OEWreathArcFxn**

See also

- *OEDrawWreathSurfaceArc* function
- *OESetSurfaceArcFxn* function
- *OEDraw2DSurface* function

The following methods are publicly inherited from *OESurfaceArcFxnBase*:

operator()	*CreateCopy*

Constructors

```
OEWreathArcFxn(const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Default constructor.

pen
The graphical properties of the arc.

operator()

```
bool operator()(OEDepict::OEImageBase& image, const OESurfaceArc& arc) const
```

Draws a surface arc with the *OESurfaceArcStyle.Wreath* style.

See example in *Example of drawing molecule surface using OEWreathArcFxn*.

image
The image on which the surface arc is drawn.

arc
The object that stores the parameters of the surface arc.

See also

- *OESurfaceArc* class

CreateCopy

```
OESurfaceArcFxnBase* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEWreathArcFxn* object is dynamically allocated and owned by the caller.

4.3 OEGrapheme Constants

4.3.1 OE2DPropMapSetup

This namespace encodes symbolic constants used as bit-masks to indicate which property map parameters are being created when invoking the *OEConfigure2DPropMap* function.

All

The combination of following constants:

- *OE2DPropMapSetup.Legend*
- *OE2DPropMapSetup.MaxValue*
- *OE2DPropMapSetup.MinValue*
- *OE2DPropMapSetup.NegativeColor*
- *OE2DPropMapSetup.PositiveColor*
- *OE2DPropMapSetup.RadiusRatio*
- *OE2DPropMapSetup.Resolution*

Legend

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the location of the legend.

```
Contents of parameter -legend
    Aliases : -l
    Type : string
    Allow list : false
    Default : Bottom
    Simple : true
    Required : false
    Legal values : Hidden Top Bottom Left Right
    Brief : Location of the legend
    Detail
```

See also

- *OELegendLocation* namespace
- *OE2DPropMap.SetLegendLocation* method

MaxValue

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the maximum value of the property map.

```
Contents of parameter -maxvalue
    Aliases : -maxv
    Type : float
    Allow list : false
    Default : (parameter does not have a default)
    Simple : true
    Required : false
    Brief : Maximum property limit
```

See also

- *OE2DPropMap.SetMaxValue* method

MinValue

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the minimum value of the property map.

```
Contents of parameter -minvalue
    Aliases : -minv
    Type : float
    Allow list : false
    Default : (parameter does not have a default)
    Simple : true
    Required : false
    Brief : Minimum property limit
```

See also

- *OE2DPropMap.SetMinValue* method

NegativeColor

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the color used to represent negative values in the property map.

```
Contents of parameter -negcolor
    Aliases : -negc
    Type : string
    Allow list : false
    Default : red
    Simple : true
    Required : false
    Legal values : black blue bluetint brown cyan darkblue darkbrown
    darkcyan darkgreen darkgrey darkmagenta darkorange darkpurple darkred
    darkrose darksalmon darkyellow gold green greenblue greentint grey
    hotpink lightblue lightbrown lightgreen lightgrey lightorange
    lightpurple lightsalmon limegreen magenta mediumblue mediumbrown
    mediumgreen mediumorange mediumpurple mediumsalmon mediumyellow
    olivebrown olivegreen olivegrey orange pink pinktint purple red
    redorange royalblue seagreen skyblue violet white yellow yellowtint
    Brief : Color of the negative values
```

See also

- *OE2DPropMap.SetNegativeColor* method

PositiveColor

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the color used to represent positive values in the property map.

```
Contents of parameter -poscolor
    Aliases : -posc
    Type : string
    Allow list : false
    Default : blue
    Simple : true
    Required : false
    Legal values : black blue bluetint brown cyan darkblue darkbrown
    darkcyan darkgreen darkgrey darkmagenta darkorange darkpurple darkred
    darkrose darksalmon darkyellow gold green greenblue greentint grey
    hotpink lightblue lightbrown lightgreen lightgrey lightorange
    lightpurple lightsalmon limegreen magenta mediumblue mediumbrown
    mediumgreen mediumorange mediumpurple mediumsalmon mediumyellow
```

(continues on next page)

(continued from previous page)

```
    olivebrown olivegreen olivegrey orange pink pinktint purple red
    redorange royalblue seagreen skyblue violet white yellow yellowtint
    Brief : Color of the positive values
```

See also

- *OE2DPropMap.SetPositiveColor* method

RadiusRatio

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the Gaussian radius of the property map.

```
Contents of parameter -radiusratio
    Aliases : -rad
    Type : float
    Allow list : false
    Default : 1.0
    Simple : true
    Required : false
    Legal ranges :
        2.0 to 0.5
    Brief : Ratio to scale the gaussian radius
```

See also

- *OE2DPropMap.SetRadiusRatio* method

Resolution

Passing this constant to the *OEConfigure2DPropMap* function results in generating the following default interface to configure the resolution of the property map.

```
Contents of parameter -resolution
    Aliases : -res
    Type : int
    Allow list : false
    Default : 2
    Simple : true
    Required : false
    Legal ranges :
        20 to 2
    Brief : Resolution of the property map
```

See also

- *OE2DPropMap.SetResolution* method

4.3.2 OEActiveSiteRenderStyle

This namespace contains constants representing various active site representation styles.

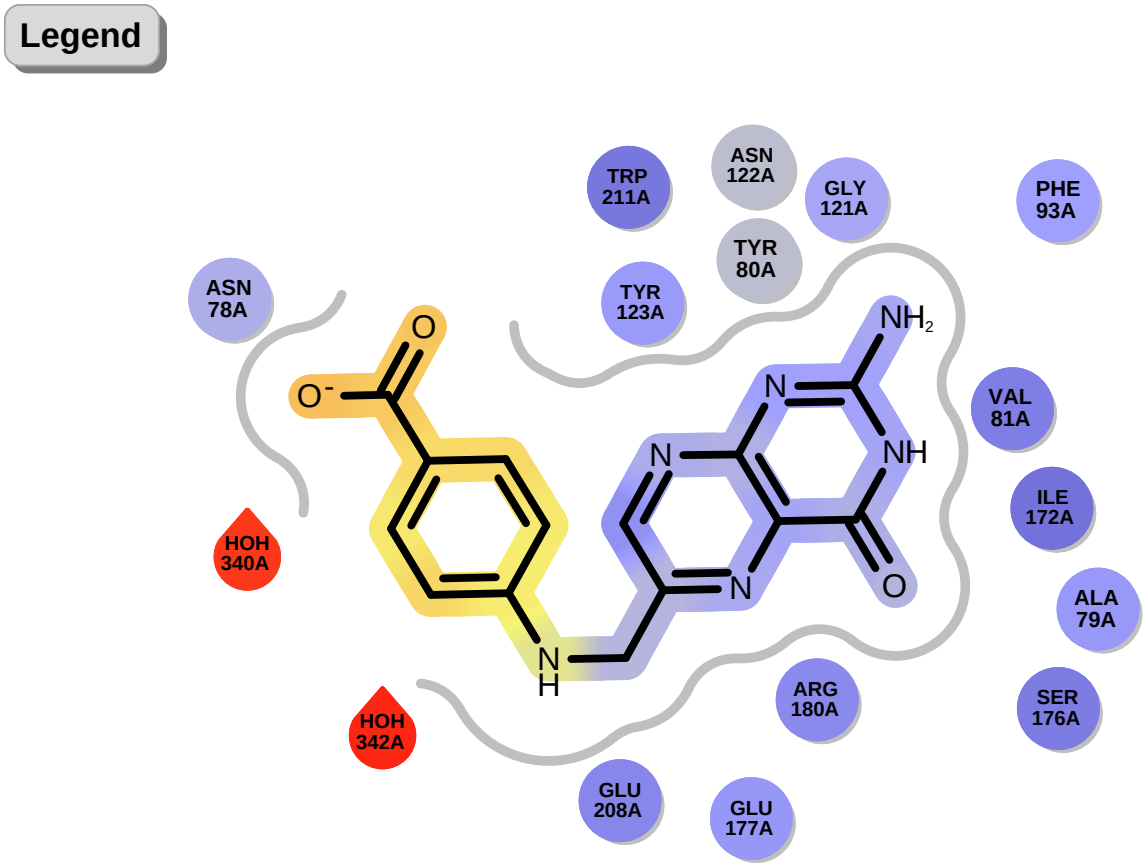
See also

- *OERenderActiveSiteMaps* function

Default

The label style is *OEActiveSiteRenderStyle.InteractionMap*.

BFactorMap



See also

- *OERenderBFactorMap* function
- *OEDrawBFactorMapLegend* function

Code Example

- Visualizing Protein-Ligand B-Factor Map **OpenEye Python Cookbook** recipe

ContactMap

There is no image available for the PDF version of this book.

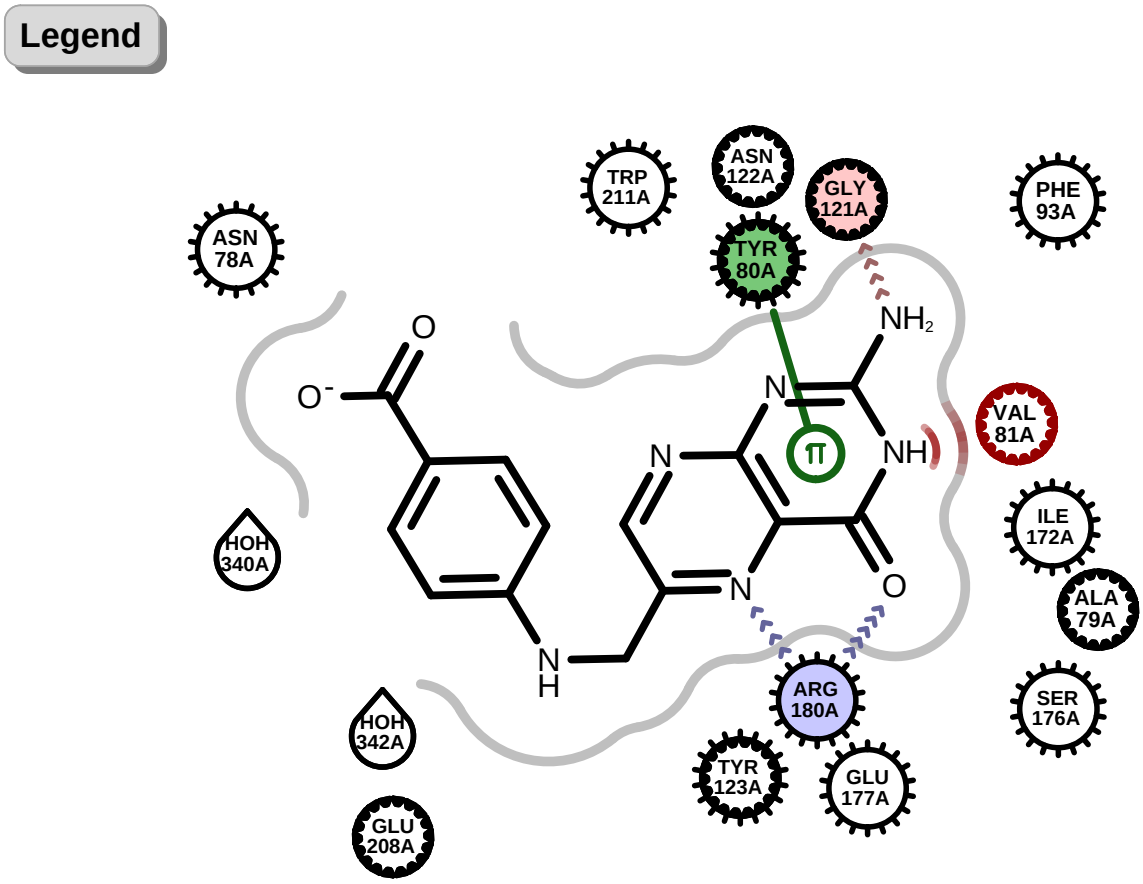
See also

- *OERenderContactMap* function

Code Example

- Visualizing Protein-Ligand Contacts **OpenEye Python Cookbook** recipe

InteractionMap



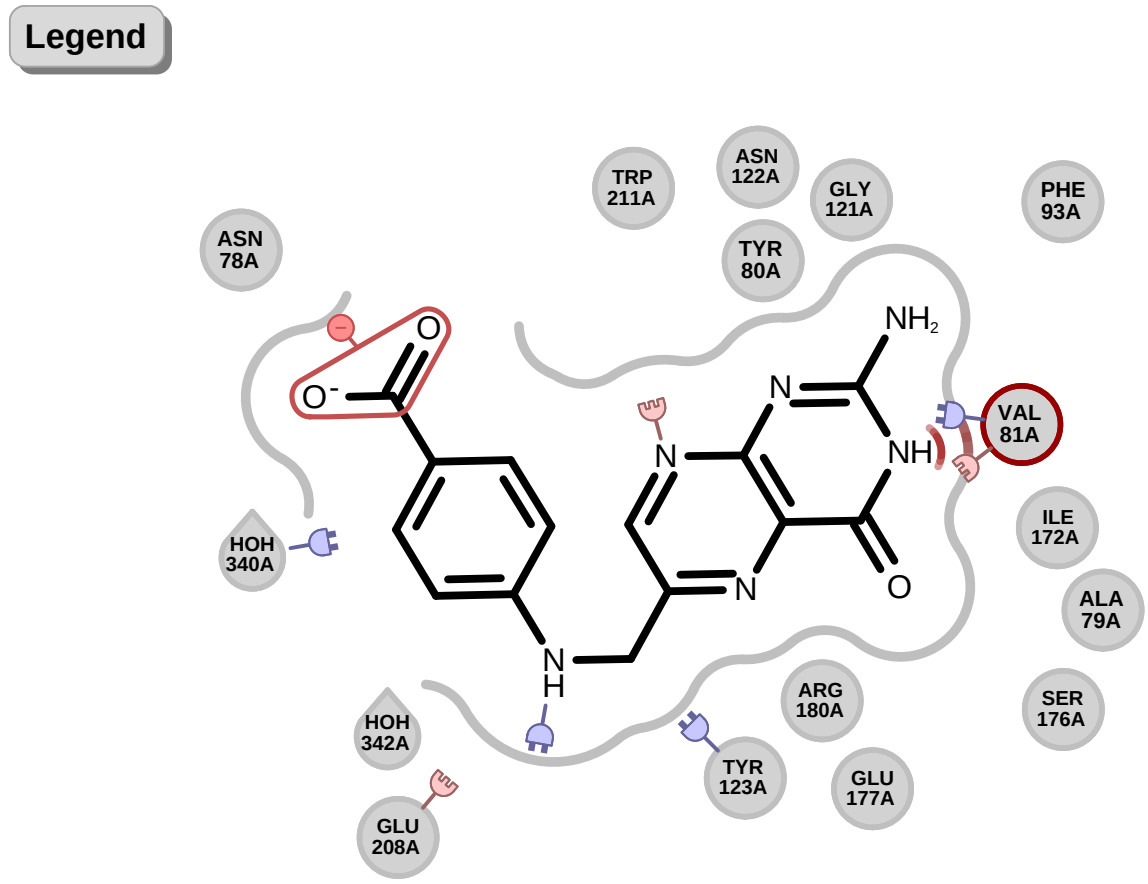
See also

- *OERenderActiveSite* function
- *OEDrawActiveSiteLegend* function
- *Depicting Active Site Interactions* example

Code Example

- Visualizing Protein-Ligand Interactions **OpenEye Python** Cookbook recipe

UnpairedMap



See also

- *OERenderUnpairedInteractionMap* function
- *OEDrawUnpairedInteractionMapLegend* function

Code Example

- Visualizing Protein-Ligand Unpaired Interactions **OpenEye Python Cookbook** recipe

4.3.3 OECircleStyle

This namespace contains constants representing various circle styles.

See also

- *OEDrawCircle* function

The *OECircleStyle* namespace contains the following constants:

AlphaRainbow

See *Figure: Example of a circle drawn with the 'AlphaRainbow' style*.

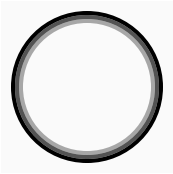


Fig. 58: **Example of a circle drawn with the 'AlphaRainbow' style**

See also

- *OEDrawAlphaRainbowCircle* function

BrickRoad

See *Figure: Example of a circle drawn with the 'BrickRoad' style*.

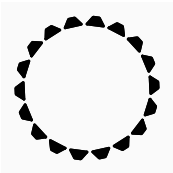


Fig. 59: **Example of a circle drawn with the 'BrickRoad' style**

See also

- *OEDrawBrickRoadCircle* function

Castle

See *Figure: Example of a circle drawn with the 'Castle' style.*

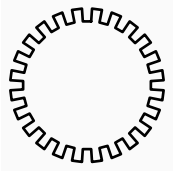


Fig. 60: **Example of a circle drawn with the 'Castle' style**

See also

- *OEDrawCastleCircle* function

Cog

See *Figure: Example of a circle drawn with the 'Cog' style.*

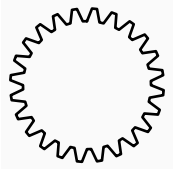


Fig. 61: **Example of a circle drawn with the 'Cog' style**

See also

- *OEDrawCogCircle* function

Default

See *Figure: Example of a circle drawn with the 'Default' style.*

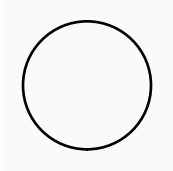


Fig. 62: **Example of a circle drawn with the 'Default' style**

See also

- *OEDrawDefaultCircle* function

Eyelash

See *Figure: Example of a circle drawn with the 'Eyelash' style.*

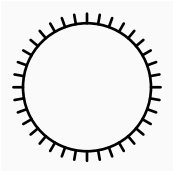


Fig. 63: **Example of a circle drawn with the 'Eyelash' style**

See also

- *OEDrawEyelashCircle* function

Flower

See *Figure: Example of a circle drawn with the 'Flower' style.*

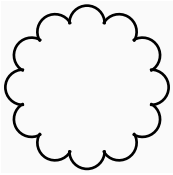


Fig. 64: **Example of a circle drawn with the 'Flower' style**

See also

- *OEDrawFlowerCircle* function

GreekKey

See *Figure: Example of a circle drawn with the 'GreekKey' style.*



Fig. 65: **Example of a circle drawn with the 'GreekKey' style**

See also

- *OEDrawGreekKeyCircle* function

Necklace

See *Figure: Example of a circle drawn with the 'Necklace' style.*

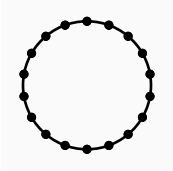


Fig. 66: **Example of a circle drawn with the 'Necklace' style**

See also

- *OEDrawNecklaceCircle* function

OliveBranch

See *Figure: Example of a circle drawn with the 'OliveBranch' style.*



Fig. 67: **Example of a circle drawn with the 'OliveBranch' style**

See also

- *OEDrawOliveBranchCircle* function

Pearls

See *Figure: Example of a circle drawn with the 'Pearls' style*.

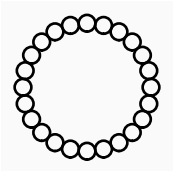


Fig. 68: **Example of a circle drawn with the 'Pearls' style**

See also

- *OEDrawPearlsCircle* function

RaceTrack

See *Figure: Example of a circle drawn with the 'RaceTrack' style*.

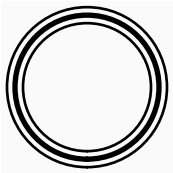


Fig. 69: **Example of a circle drawn with the 'RaceTrack' style**

See also

- *OEDrawRaceTrackCircle* function

RailroadTrack

See *Figure: Example of a circle drawn with the 'RailroadTrack' style*.

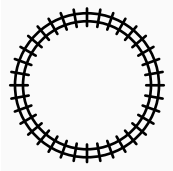


Fig. 70: **Example of a circle drawn with the 'RailroadTrack' style**

See also

- *OEDrawRailroadTrackCircle* function

Saw

See *Figure: Example of a circle drawn with the 'Saw' style*.

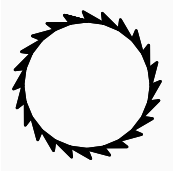


Fig. 71: **Example of a circle drawn with the 'Saw' style**

See also

- *OEDrawSawCircle* function

Simpson

See *Figure: Example of a circle drawn with the 'Simpson' style*.

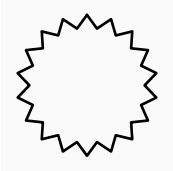


Fig. 72: **Example of a circle drawn with the 'Simpson' style**

See also

- *OEDrawSimpsonCircle* function

Stitch

See *Figure: Example of a circle drawn with the 'Stitch' style*.

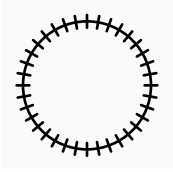


Fig. 73: **Example of a circle drawn with the 'Stitch' style**

See also

- *OEDrawStitchCircle* function

Sun

See *Figure: Example of a circle drawn with the 'Sun' style*.

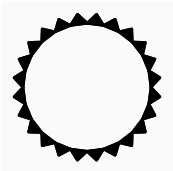


Fig. 74: **Example of a circle drawn with the 'Sun' style**

See also

- *OEDrawSunCircle* function

Wreath

See *Figure: Example of a circle drawn with the 'Wreath' style*.

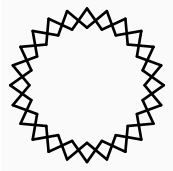


Fig. 75: **Example of a circle drawn with the 'Wreath' style**

See also

- *OEDrawWreathCircle* function

4.3.4 OEColorAtomStyle

This namespace contains constants representing various styles color atoms can be depicted by the *OERenderShapeQuery* function.

Default

The default color atom style is *OEColorAtomStyle.Circle*.

Hidden

No color atoms are depicted.

Circle

Each color atom is represented by a circle on the query (*i.e.* reference) molecule. If more than one color atom occupies the same space in 3D, the color atoms are represented by half, third, *etc* circles accordingly. See image in *Figure: Example of visualizing color atoms by circles*. The color of each circle is defined by the *OEColorForceFieldDisplay* object.

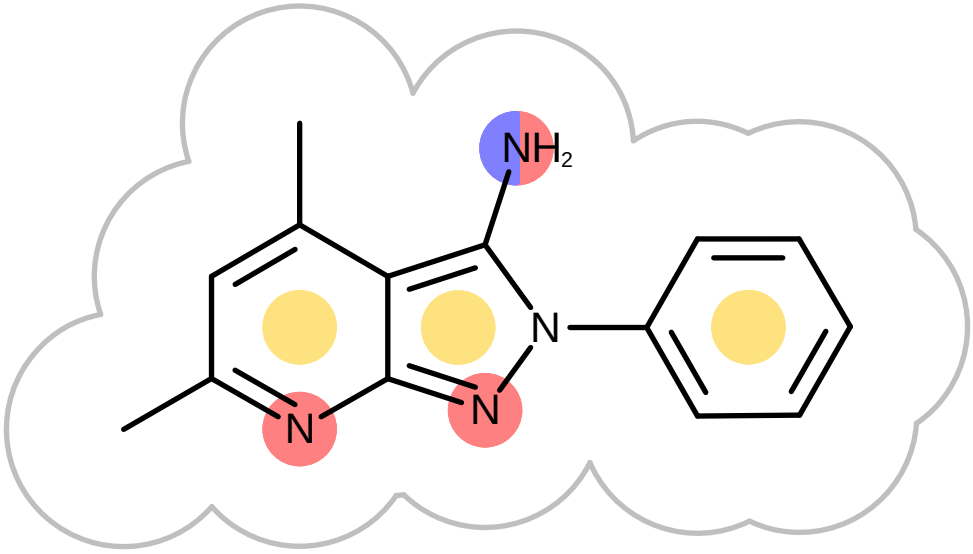


Fig. 76: **Example of visualizing color atoms by circles**

4.3.5 OELegendLocation

This namespace contains constants representing the position of the legend displayed with the property map:

See also

- *OE2DPropMap.GetLegendLocation* method
- *OE2DPropMap.SetLegendLocation* method

The *OELegendLocation* namespace contains the following constants:

Default

The default legend location is *OELegendLocation.Bottom*.

Bottom

See example in *Figure: Example of property map with its legend located at the bottom.*

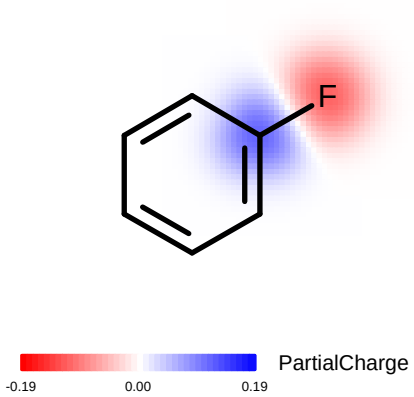


Fig. 77: **Example of property map with the legend located at the bottom**

Hidden

See example in *Figure: Example of property map with no legend*.

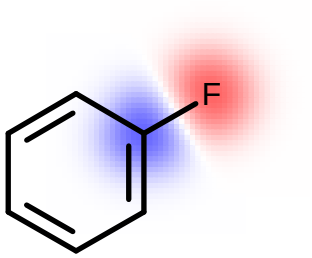


Fig. 78: **Example of property map with no legend**

Left

See example in *Figure: Example of property map with its legend located on the left*.

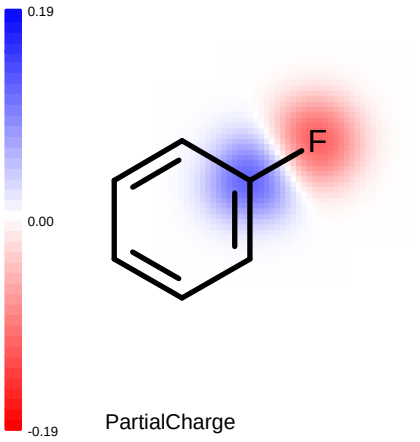


Fig. 79: **Example of property map with the legend located on the left**

Right

See example in *Figure: Example of property map with its legend located on the right*.

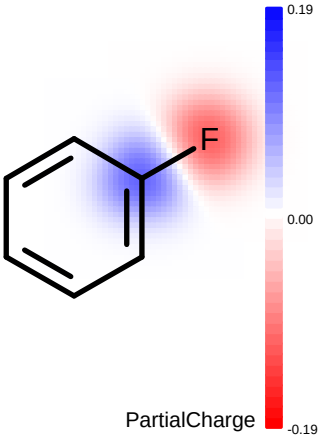


Fig. 80: **Example of property map with the legend located on the right**

Top

See example in *Figure: Example of property map with its legend located at the top*

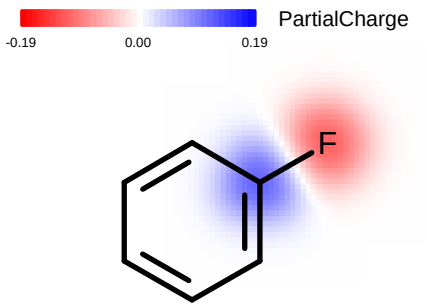


Fig. 81: **Example of property map with the legend located at the top**

4.3.6 OEPatternDirection

The *OEPatternDirection* namespace is used to customize various circle and arc drawing functions.

See also

- *OEDrawBrickRoadCircle* function
- *OEDrawEyelashCircle* function
- *OEDrawBrickRoadSurfaceArc* function
- *OEDrawEyelashSurfaceArc* function

The *OEPatternDirection* namespace contains the following constants:

Default

The default pattern direction is *OEPatternDirection.Outside*.

Inside

See example in *Figure: Example of circle and arc drawing with patterns 'inside'*.

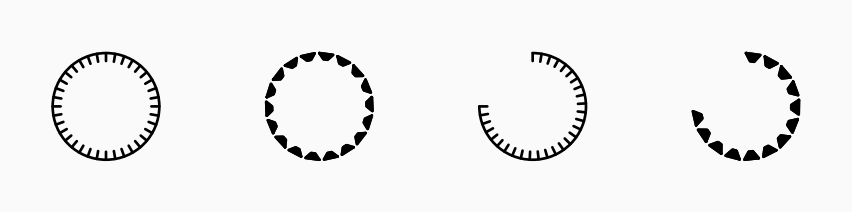


Fig. 82: **Example of circle and arc drawing with patterns 'inside'**

Outside

See example in *Figure: Example of circle and arc drawing with patterns 'outside'*.

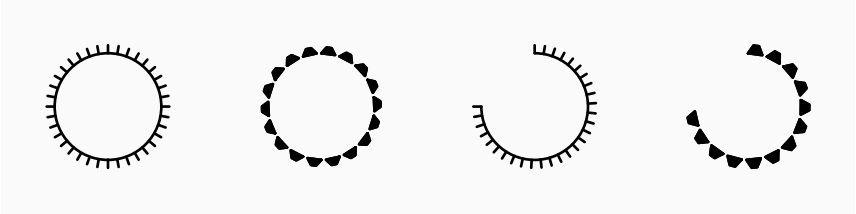


Fig. 83: **Example of circle and arc drawing with patterns 'outside'**

4.3.7 OEPeptideLabelStyle

Warning

This is a deprecated API.

This namespace contains constants representing various label styles for peptide depiction.

See also

- *OEPeptideDisplayOptions.GetLabelStyle* method
- *OEPeptideDisplayOptions.SetLabelStyle* method
- *OEDrawPeptide* function

Default

SingleLetter

ThreeLetters

4.3.8 OEPlotMarkerStyle

This namespace contains constants representing various plot marker styles.

See also

- *OEPlotMarker* class

The *OEPlotMarkerStyle* namespace contains the following constants:

This namespace contains constants.

Default

The default marker style is *OEPlotMarkerStyle.Circle*.

Circle

Square

4.3.9 OEShapeOverlapDisplayStyle

This namespace contains constants representing various styles that can be used to depict shape overlays when calling the *OERenderShapeQuery* function.

See also

- *OEShapeOverlapDisplayOptions.SetOverlapDisplayStyle* method

Default

The default shape overlap display style is *OEShapeOverlapDisplayStyle.PropertyMap*.

PropertyCloud

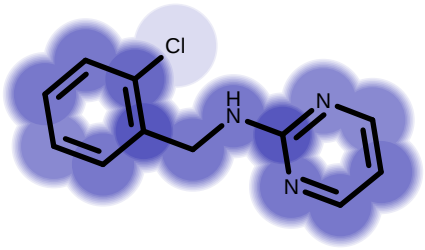


Fig. 84: **Example of visualizing shape overlaps with PropertyCloud style**

PropertyMap

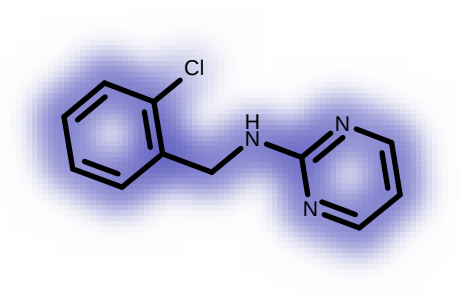


Fig. 85: **Example of visualizing shape overlaps with PropertyMap style**

4.3.10 OESurfaceArcScale

This namespace contains constants that determine the range of the radius scaling factor that can be used to generate the arcs of 2D molecule surfaces.

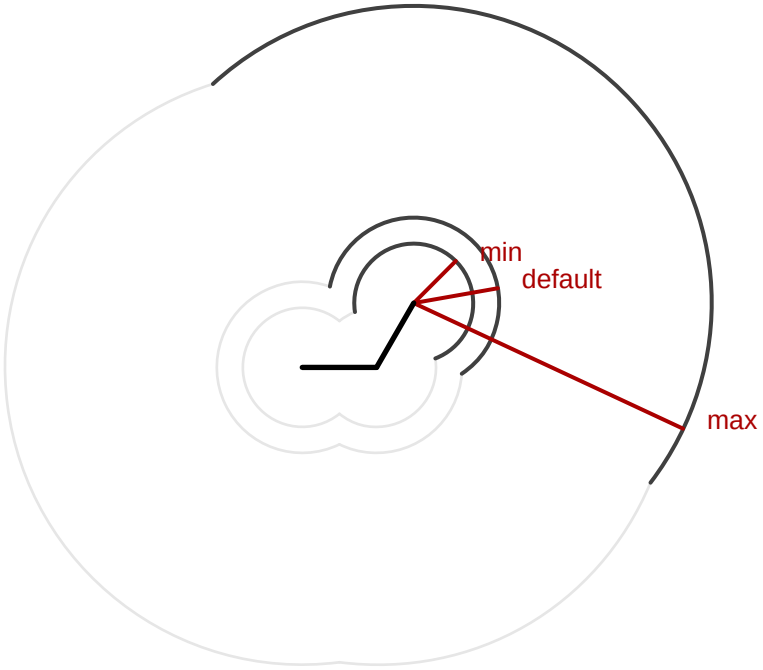


Fig. 86: **Example of various radius scales used to generate arcs**

See also

- *OEGet2DSurfaceArcs* function
- *OEGetMoleculeSurfaceScale* function

Default

Specifies the default radius scaling used to generate the arcs of the molecule surface when calling the *OEGet2DSurfaceArcs* function.

Maximum

Specifies the maximum radius scaling that can be used to generate the arcs of the molecule surface of a specific atom when calling the *OEGet2DSurfaceArcs* function.

Minimum

Specifies the minimum radius scaling that can be used to generate the arcs of the molecule surface of a specific atom when calling the *OEGet2DSurfaceArcs* function.

4.3.11 OESurfaceArcStyle

This namespace contains constants representing various arc styles.

See also

- *OEDrawSurfaceArc* function

The *OESurfaceArcStyle* namespace contains the following constants:

AlphaRainbow

See *Figure: Example of an arc drawn with the 'AlphaRainbow' style*.



Fig. 87: **Example of an arc drawn with the 'AlphaRainbow' style**

See also

- *OEDrawAlphaRainbowSurfaceArc* function

BrickRoad

See *Figure: Example of an arc drawn with the 'BrickRoad' style*.

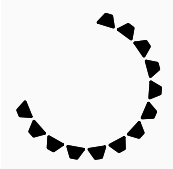


Fig. 88: **Example of an arc drawn with the 'BrickRoad' style**

See also

- *OEDrawBrickRoadSurfaceArc* function

Castle

See *Figure: Example of an arc drawn with the 'Castle' style*.

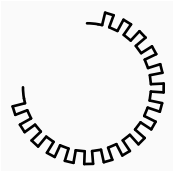


Fig. 89: **Example of an arc drawn with the 'Castle' style**

See also

- *OEDrawCastleSurfaceArc* function

Cog

See *Figure: Example of an arc drawn with the 'Cog' style*.

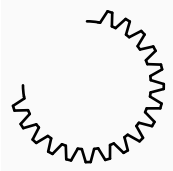


Fig. 90: **Example of an arc drawn with the 'Cog' style**

See also

- *OEDrawCogSurfaceArc* function

Default

See *Figure: Example of an arc drawn with the 'Default' style*.

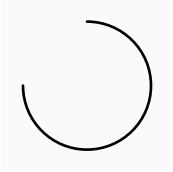


Fig. 91: **Example of an arc drawn with the 'Default' style**

See also

- *OEDrawDefaultSurfaceArc* function

Eyelash

See *Figure: Example of an arc drawn with the 'Eyelash' style*.

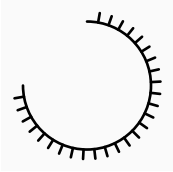


Fig. 92: **Example of an arc drawn with the 'Eyelash' style**

See also

- *OEDrawEyelashSurfaceArc* function

Flower

See *Figure: Example of an arc drawn with the 'Flower' style.*

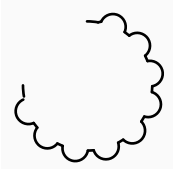


Fig. 93: **Example of an arc drawn with the 'Flower' style**

See also

- *OEDrawFlowerSurfaceArc* function

Necklace

See *Figure: Example of an arc drawn with the 'Necklace' style.*



Fig. 94: **Example of an arc drawn with the 'Necklace' style**

See also

- *OEDrawNecklaceSurfaceArc* function

OliveBranch

See *Figure: Example of an arc drawn with the 'OliveBranch' style.*



Fig. 95: **Example of an arc drawn with the 'OliveBranch' style**

See also

- *OEDrawOliveBranchSurfaceArc* function

Pearls

See *Figure: Example of an arc drawn with the 'Pearls' style*.

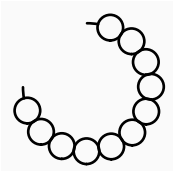


Fig. 96: **Example of an arc drawn with the 'Pearls' style**

See also

- *OEDrawPearlsSurfaceArc* function

RaceTrack

See *Figure: Example of an arc drawn with the 'RaceTrack' style*.



Fig. 97: **Example of an arc drawn with the 'RaceTrack' style**

See also

- *OEDrawRaceTrackSurfaceArc* function

RailroadTrack

See *Figure: Example of an arc drawn with the 'RailroadTrack' style*.



Fig. 98: **Example of an arc drawn with the 'RailroadTrack' style**

See also

- *OEDrawRailroadTrackSurfaceArc* function

Saw

See *Figure: Example of an arc drawn with the 'Saw' style*.



Fig. 99: **Example of an arc drawn with the 'Saw' style**

See also

- *OEDrawSawSurfaceArc* function

Simpson

See *Figure: Example of an arc drawn with the 'Simpson' style*.

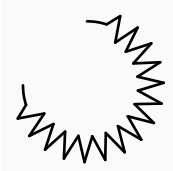


Fig. 100: **Example of an arc drawn with the 'Simpson' style**

See also

- *OEDrawSimpsonSurfaceArc* function

Stitch

See *Figure: Example of an arc drawn with the 'Stitch' style*.

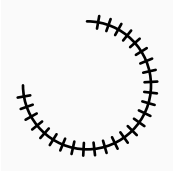


Fig. 101: **Example of an arc drawn with the 'Stitch' style**

See also

- *OEDrawStitchSurfaceArc* function

Sun

See *Figure: Example of an arc drawn with the 'Sun' style*.

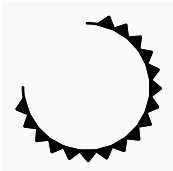


Fig. 102: **Example of an arc drawn with the 'Sun' style**

See also

- *OEDrawSunSurfaceArc* function

Wreath

See *Figure: Example of an arc drawn with the 'Wreath' style*.

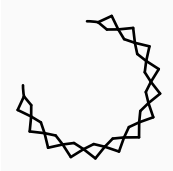


Fig. 103: **Example of an arc drawn with the 'Wreath' style**

See also

- *OEDrawWreathSurfaceArc* function

4.4 OEGrapheme Functions

4.4.1 OEAddComplexSurfaceArcs

```
bool OEAddComplexSurfaceArcs(OEChem::OEMolBase& ligand,
                             const OEChem::OEMolBase& protein,
                             const OESurfaceArcFxnBase& solventExposedFxn =␣
↪OEDefaultSolventArcFxn(),
                             const OESurfaceArcFxnBase& buriedFxn =␣
↪OEDefaultBuriedArcFxn(),
                             const OEComplexSurfaceArcFxnBase& cavityFxn =␣
↪OEDefaultCavityArcFxn(),
                             const OEComplexSurfaceArcFxnBase& voidFxn =␣
↪OEDefaultVoidArcFxn())
```

Takes a ligand-protein complex and generates the 2D molecule surface of the ligand, where the arcs of the surface are annotated based on the distance calculated between the accessible surface of the ligand and the protein. See *Figure: Example of surface drawing with the OEAddComplexSurfaceArcs function*.

ligand
The molecule for which the 2D surface is generated. Must have 3D coordinates and VDW radii assigned.

protein
The macromolecule whose contact with the ligand influences the surface drawing. Must have 3D coordinates and VDW radii assigned.

solventExposedFxn
This functor is used to draw those arc segments of the 2D molecule surface where the ligand is exposed to the solvent in the 3D complex. By default, the *OEDefaultSolventArcFxn* functor is used.

buriedFxn
This functor is used to draw those arc segments of the 2D molecule surface where the ligand and the protein tightly fit in the 3D complex. By default, the *OEDefaultBuriedArcFxn* functor is used.

cavityFxn
This functor is used to draw those arc segments of the 2D molecule surface where there is a cavity detected between the ligand and the protein in the 3D complex. By default, the *OEDefaultCavityArcFxn* functor is used.

voidFxn
This functor is used to draw those arc segments of the 2D molecule surface where there is a small gap detected between the ligand and the protein in the 3D complex. By default, the *OEDefaultVoidArcFxn* functor is used.

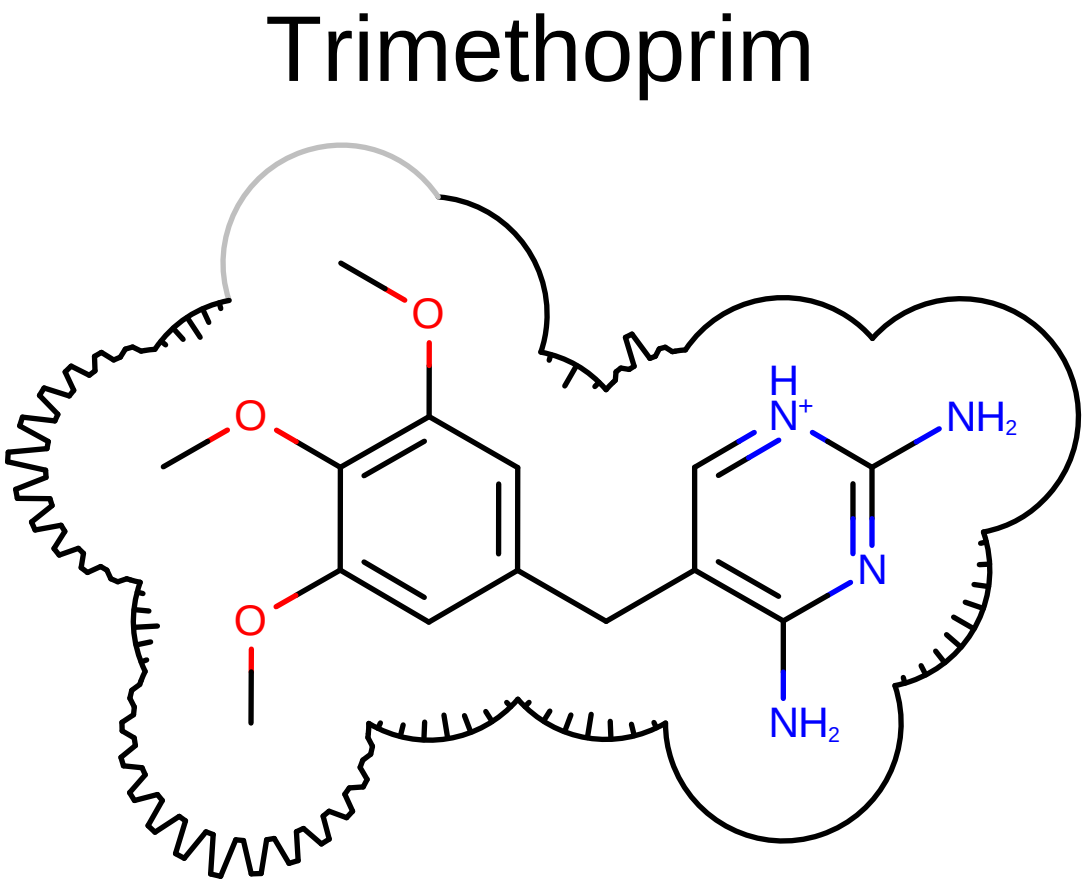


Fig. 104: **Example of surface drawing with the OEAddComplexSurfaceArcs function**

See also

- *Drawing a Ligand-Protein Complex Surface* chapter

4.4.2 OEAddGlyph

```
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEAtomGlyphBase& atomglyph,
                const OEChem::OEMatchBase& match)
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEAtomGlyphBase& atomglyph,
                const OEChem::OEAtomBondSet& abset)
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEAtomGlyphBase& atomglyph,
                const OESystem::OEUnaryPredicate<OEChem::OEAtomBase>& pred)
```

Annotates atoms with the style implemented in the given *OEAtomGlyphBase* object. The atoms being annotated can be specified either by an OEMatchBase or an OEAtomBondSet object or by an atom predicate.

disp
The OE2DMolDisplay object whose atoms are being annotated.

atomglyph
Specifies the style of the annotation. Each atom is being annotated by calling the *OEAtomGlyphBase.RenderGlyph* method.

match
OEAddGlyph function annotates the target atoms of the OEMatchBase object.

abset
OEAddGlyph function annotates the atoms returned by the `GetAtoms` method.

pred
OEAddGlyph function annotates only atoms for which the given atom predicate returns *true*.

See also

- *Annotating Atoms and Bonds* chapter
- *OEAtomGlyphBase* class
- *OEAtomGlyphCircle* class

```
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEBondGlyphBase& bondglyph,
                const OEChem::OEMatchBase& match)
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEBondGlyphBase& bondglyph,
                const OEChem::OEAtomBondSet& abset)
void OEAddGlyph(OEDepict::OE2DMolDisplay& disp,
                const OEBondGlyphBase& bondglyph,
                const OESystem::OEUnaryPredicate<OEChem::OEBondBase>& pred)
```

Annotates bonds with the style implemented in the given *OEBondGlyphBase* object. The bond being annotated can be specified either by an OEMatchBase or an OEAtomBondSet object or by a bond predicate.

disp
The OE2DMolDisplay object whose bonds are being annotated.

bondglyph
Specifies the style of the annotation. Each bond is being annotated by calling the *OEBondGlyphBase.RenderGlyph* method.

match
OEAddGlyph function annotates the target bonds of the OEMatchBase object.

abset
OEAddGlyph function annotates the bonds returned by the `GetBonds` method.

pred
OEAddGlyph function annotates only bonds for which the given bond predicate returns *true*.

See also

- *Annotating Atoms and Bonds* chapter
- *OEBondGlyphBase* class
- *OEBondGlyphArrow*
- *OEBondGlyphCircle*
- *OEBondGlyphCross* class
- *OEBondGlyphCurvedArrow* class
- *OEBondGlyphScissors* class

4.4.3 OEAddLigandHighlighting

Highlights atoms and/or bonds of the displayed ligand.

Table 4: **The overloaded versions of the OEAddLigandHighlighting function**

Link	Description
OEAddLigandHighlighting(disp, highlight, atompred)	highlights ligand atoms with OEHighlightBase
OEAddLigandHighlighting(adisp, highlight, bondpred)	highlights ligand bonds with OEHighlightBase
OEAddLigandHighlighting(adisp, highlight, atompred, bondpred)	highlights ligand atoms and bonds with OEHighlightBase
OEAddLigandHighlighting(adisp, highlight, match)	highlights ligand atoms and bonds of a match with OEHighlightBase
OEAddLigandHighlighting(adisp, highlight, abset)	highlights ligand atoms and bonds of a set with OEHighlightBase

Parameters :

adisp
The *OE2DActiveSiteDisplay* object of which ligand atoms add/or ligand bonds being highlighted.

highlight
The OEHighlightBase object that specifies the style of the highlighting.

atompred
OEAddLigandHighlighting function highlights only ligand atoms for which the given atom predicate returns *true*.

bondpred
OEAddLigandHighlighting function highlights only ligand bonds for which the given bond predicate returns *true*.

match
OEAddLigandHighlighting function highlights the target atoms and bonds of the OEMatchBase object.

abset
Stores the ligand atoms and bonds being highlighted.

See also

- OEHighlightByColor class
- OEHighlightByCogwheel class
- OEHighlightByBallAndStick class
- OEHighlightByStick class
- OEHighlightByLasso class

1. Highlighting using **atom** and **bond predicates**:

```
void OEAddLigandHighlighting(OE2DActiveSiteDisplay& adisp,
                             const OEDepict::OEHighlightBase& highlight,
                             const OESystem::OEUnaryPredicate<OEChem::OEAtomBase>&␣
↪atompred)
```

Highlights ligand atoms with the style implemented in the given OEHighlightBase object. The atoms being highlighted are specified by given atom predicate.

```
void OEAddLigandHighlighting(OE2DActiveSiteDisplay& adisp,
                             const OEDepict::OEHighlightBase& highlight,
                             const OESystem::OEUnaryPredicate<OEChem::OEBondBase>&␣
↪bondpred)
```

Highlights ligand bonds with the style implemented in the given OEHighlightBase object. The bonds being highlighted are specified by given bond predicates.

```
void OEAddLigandHighlighting(OE2DActiveSiteDisplay& adisp,
                             const OEDepict::OEHighlightBase& highlight,
                             const OESystem::OEUnaryPredicate<OEChem::OEAtomBase>&␣
↪atompred,
                             const OESystem::OEUnaryPredicate<OEChem::OEBondBase>&␣
↪bondpred)
```

Highlights ligand atoms and bonds with the style implemented in the given OEHighlightBase object. The atoms and bonds being highlighted are specified by given atom and bond predicates, respectively.

Example: (See image generated in *Example of highlighting 6-membered rings of a ligand using 'ball and stick' style* and *Example of highlighting 6-membered rings of a ligand using 'color' style*)

```
private class Pred6MemAromAtom : OEUnaryAtomPred
{
    public override bool Call(OEAtomBase atom)
    {
        return atom.IsAromatic() && OEChem.OEAtomIsInAromaticRingSize(atom, 6);
    }

    public override OEUnaryAtomBoolFunc CreateCopy()
    {
        OEUnaryAtomBoolFunc copy = new Pred6MemAromAtom();
        copy.ReleaseOwnership();
        return copy;
    }
}

private class Pred6MemAromBond : OEUnaryBondPred
{

    public override bool Call(OEBondBase bond)
    {
        return bond.IsAromatic() && OEChem.OEBondIsInAromaticRingSize(bond, 6);
    }

    public override OEUnaryBondBoolFunc CreateCopy()
    {
        OEUnaryBondBoolFunc copy = new Pred6MemAromBond();
        copy.ReleaseOwnership();
        return copy;
    }
}

private static void OEAddHighlighting_Predicate(OE2DActiveSiteDisplay adisp)
{
    OEHighlightByBallAndStick highlight = new OEHighlightByBallAndStick(OEChem.
↪OEBlueTint);

    OEGrapheme.OEAddLigandHighlighting(adisp, highlight, new Pred6MemAromAtom());
    OEGrapheme.OEAddLigandHighlighting(adisp, highlight, new Pred6MemAromBond());
}

private static void OEAddHighlighting_AtomAndBondPredicate(OE2DActiveSiteDisplay␣
↪adisp)
{
    OEHighlightByColor highlight = new OEHighlightByColor(OEChem.OEDarkGreen);
    OEGrapheme.OEAddLigandHighlighting(adisp, highlight,
                                       new Pred6MemAromAtom(), new␣
↪Pred6MemAromBond());
}
```

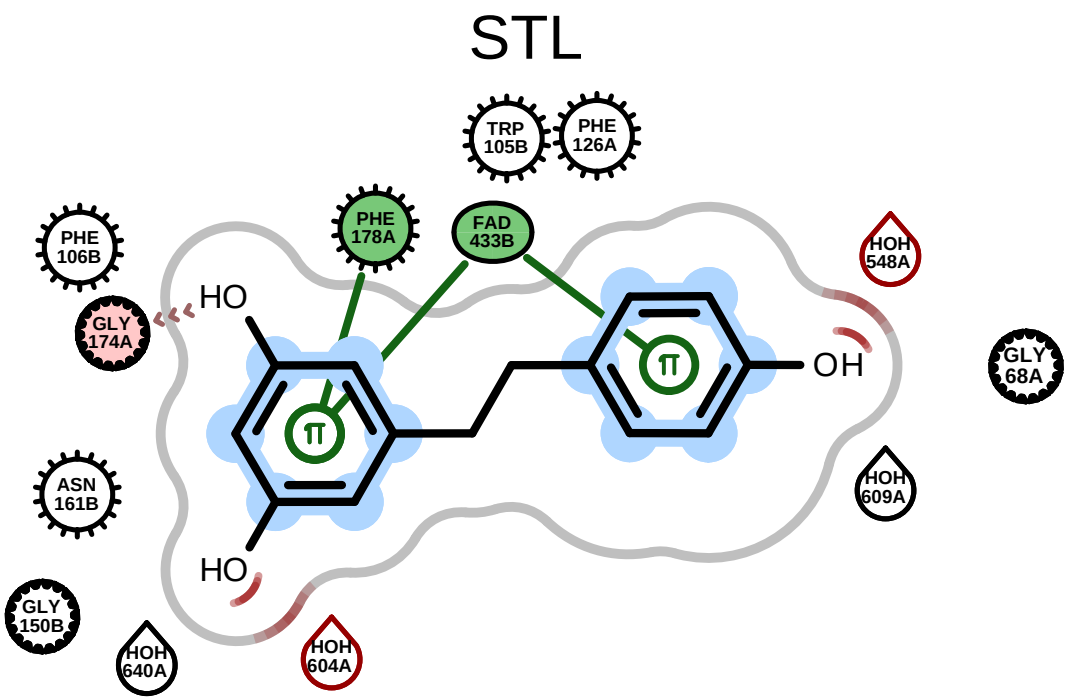


Fig. 105: **Example of highlighting 6-membered rings of a ligand using 'ball and stick' style**

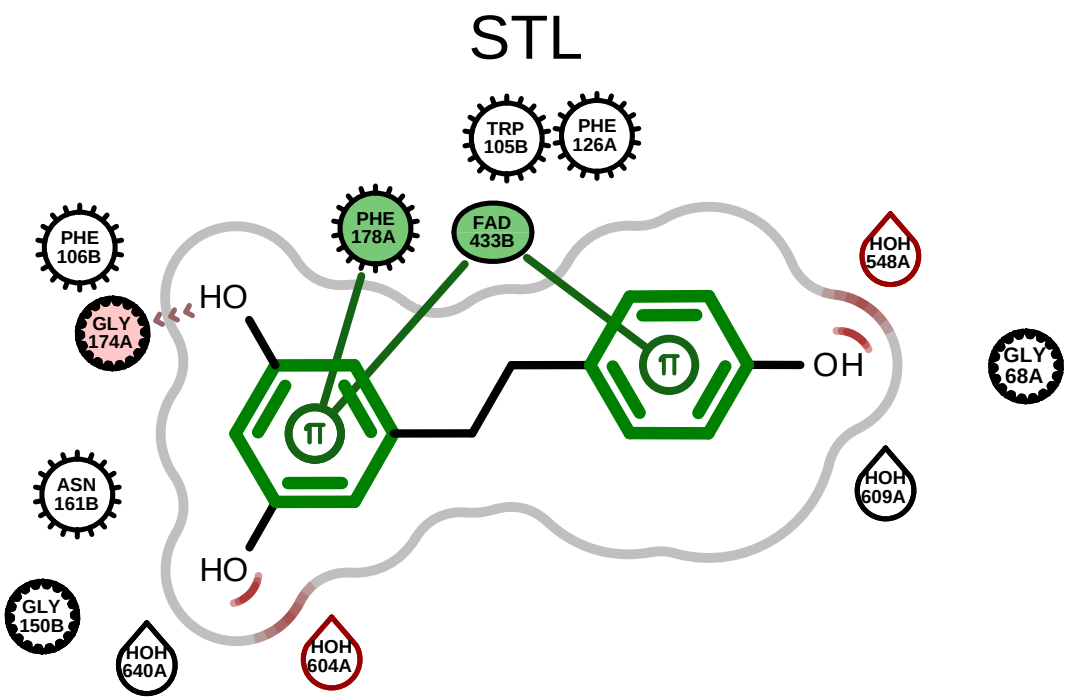


Fig. 106: **Example of highlighting 6-membered rings of a ligand using 'color' style**

See also

- Predicate Functors chapter in the *OEChem TK* manual

1. Highlighting using an OEMatchBase object that is initialized by substructure search or maximum common substructure search:

```
void OEAddLigandHighlighting(OE2DActiveSiteDisplay& adisp,
                             const OEDepict::OEHighlightBase& highlight,
                             const OEChem::OEMatchBase& match)
```

Example: (See image generated in *Example of highlighting 6-membered rings of a ligand using 'lasso' style*)

```
private static void OEAddHighlighting_OEMatch(OE2DActiveSiteDisplay adisp)
{
    OEMolBase ligand = adisp.GetDisplayedLigand();
    OESubSearch subs = new OESubSearch("a1aaaaa1");
    OEColorIter citer = OEChem.OEGetVividColors();
    bool unique = true;
    foreach (OEMatchBase match in subs.Match(ligand, unique))
    {
        if (citer.IsValid())
        {
```

(continues on next page)

(continued from previous page)

```
            OEColor hcolor = citer.Target();
            OEHighlightByLasso highlight = new OEHighlightByLasso(hcolor);
            highlight.SetConsiderAtomLabelBoundingBox(true);
            OEGrapheme.OEAddLigandHighlighting(adisp, highlight, match);
        }
    }
}
```

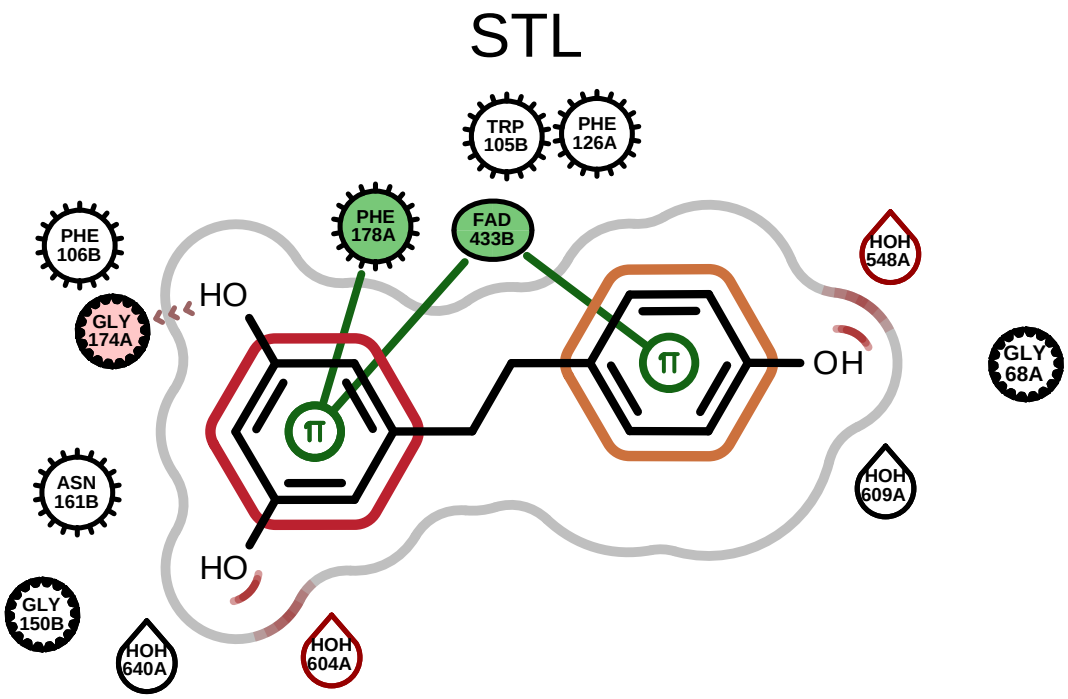


Fig. 107: **Example of highlighting 6-membered rings of a ligand using 'lasso' style**

1. Highlighting using an OEAtomBondSet object that stores the atoms and bonds being highlighted:

```
void OEAddLigandHighlighting(OE2DActiveSiteDisplay& adisp,
                             const OEDepict::OEHighlightBase& highlight,
                             const OEChem::OEAtomBondSet& abset)
```

Example: (See image generated in *Example of highlighting 6-membered rings of a ligand using 'cogwheel' style*)

Full listing.

```
private static void OEAddHighlighting_OEAtomBondSet(OE2DActiveSiteDisplay adisp)
{
    OEMolBase ligand = adisp.GetDisplayedLigand();
    OEAtomBondSet abset = new OEAtomBondSet(ligand.GetAtoms(new Pred6MemAromAtom()),
```

(continues on next page)

(continued from previous page)

```
                                                ligand.GetBonds(new␣
↪Pred6MemAromBond()));
    OEHighlightByCogwheel highlight = new OEHighlightByCogwheel(OEChem.OEPinkTint);
    highlight.SetInnerContour(false);
    OEGrapheme.OEAddLigandHighlighting(adisp, highlight, abset);
}
```

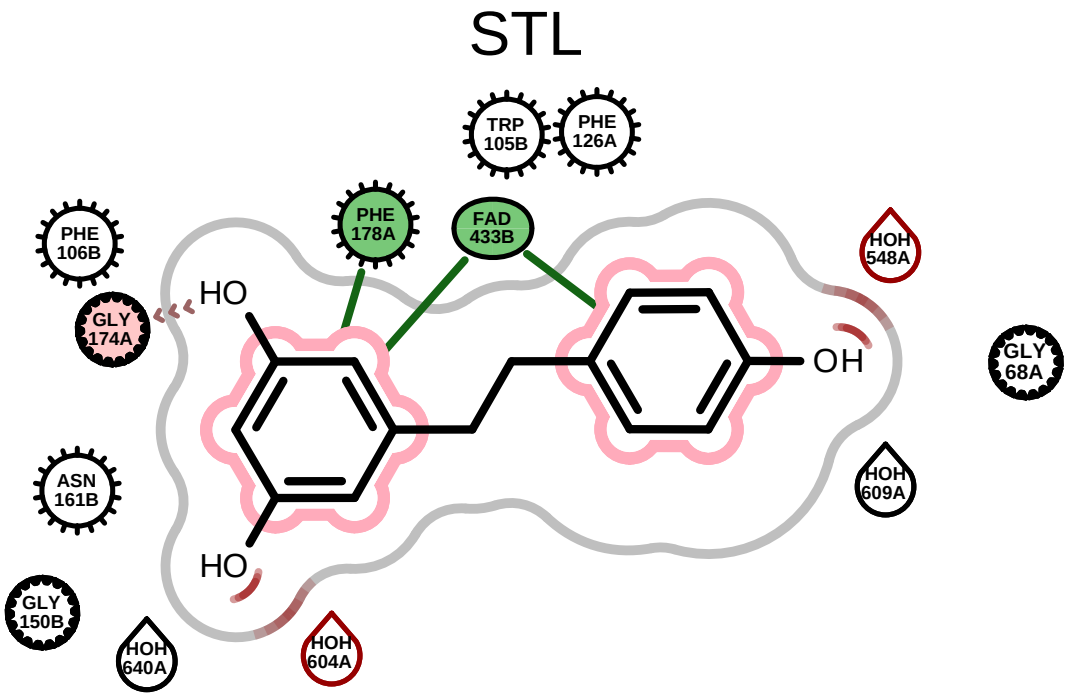


Fig. 108: **Example of highlighting 6-membered rings of a ligand using 'cogwheel' style**

4.4.4 OEAddResidueHighlighting

```
void OEAddResidueHighlighting(OE2DActiveSiteDisplay& adisp,
                              const OEDepict::OEPen& pen,
                              const OEChem::OEResidue& residue)
```

Highlights the given residues in the active site display.

adisp
: The *OE2DActiveSiteDisplay* object of which residues being highlighted.

pen
: The OEPen object that specifies the style of the highlighting.

residue
: The residue being highlighted.

Example: (See image generated in *Example of highlighting residues*)

The `OEGetLightColors` function in the below example returns eight colors that is used to highlight only the first eight residues in the generated image.

```
private static void OEAddHighlighting_OEResidue(OE2DActiveSiteDisplay adisp)
{
    OEColorIter citer = OEChem.OEGetLightColors();
    foreach (OEResidue residue in adisp.GetDisplayedResidues())
    {
        if (citer.IsValid())
        {
            OEColor color = citer.Target();
            OEPen pen = new OEPen(color, color, OEFill.On, 1.0);
            OEGrapheme.OEAddResidueHighlighting(adisp, pen, residue);
        }
    }
}
```

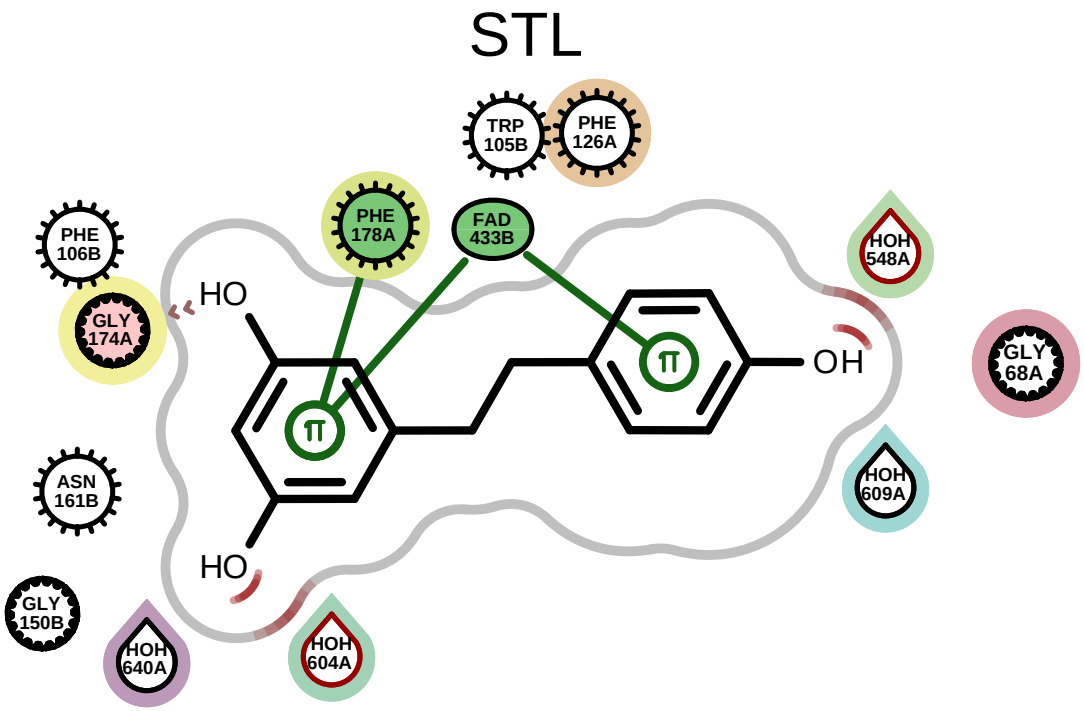


Fig. 109: **Example of highlighting residues**

4.4.5 OEClearSurfaceArcFxn

```
void OEClearSurfaceArcFxn(OEChem::OEAtomBase* atom)
```

Removes the surface drawing functor previously added to the atom.

See also

- *OEGet2DSurfaceArcs* function
- *OEGetSurfaceArcFxn* function
- *OEHasSurfaceArcFxn* function
- *OESetSurfaceArcFxn* function

4.4.6 OEConfigure2DPropMap

```
bool OEConfigure2DPropMap(OESystem::OEInterface& itf,
                          unsigned int config = OE2DPropMapSetup::All)
```

Configures the given interface to add property map style parameters.

itf
The interface being configured.

config
The option that specifies which parameters will be added to the interface This value has to be from the *OE2DPropMapSetup* namespace.

See also

- OEInterface class in the **OEChem TK** manual
- *OE2DPropMapSetup* namespace

4.4.7 OEDraw2DSurface

```
bool OEDraw2DSurface(OEDepict::OE2DMolDisplay& disp)
```

Calculates the arc segments of the molecule surface and draws the arcs into the molecule display using the functor previously added to the atoms (by calling *OESetSurfaceArcFxn*). See *Figure: Example of drawing a molecule surface using the OEDraw2DSurface function*.

disp
The molecule display on which the 2D molecule surface is drawn.

Note

If no surface drawing functor is set to the atoms, then the *OEDefaultArcFxn* functor will be used by default.

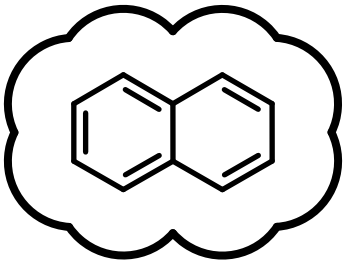


Fig. 110: **Example of drawing a molecule surface using the OEDraw2DSurface function**

See also

- *Drawing a Molecule Surface* chapter
- *OESetSurfaceArcFxn*

```
bool OEDraw2DSurface(OEDepict::OE2DMolDisplay& disp,
                     const OEDepict::OE2DMolDisplay& refdisp)
```

Allows a molecule surface to be generated on one molecule display but drawn into another display. See *Figure: Example of using OEDraw2DSurface with reference molecule display*.

disp
The molecule display on which the 2D molecule surface is drawn.

refdisp
The molecule display for which the 2D molecule surface is calculated.

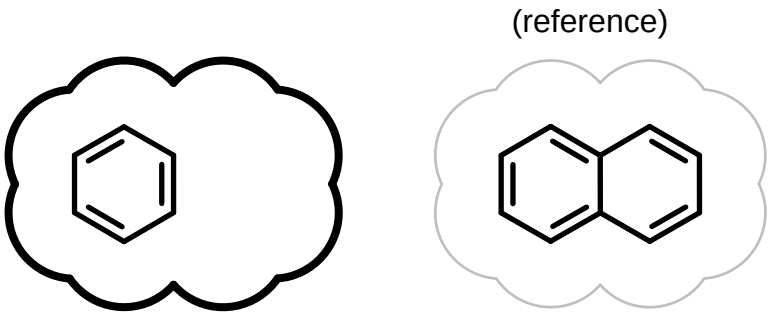


Fig. 111: **Example of using OEDraw2DSurface with reference molecule display**

See also

- *Drawing a Molecule Surface* chapter
- *OESetSurfaceArcFxn*

4.4.8 OEDrawActiveSiteLegend

```
void OEDrawActiveSiteLegend(OEDepict::OEImageBase& image,
                            const OE2DActiveSiteDisplay& adisp,
                            const OE2DActiveSiteLegendDisplayOptions& opts)
```

Draws legend for active site interaction map.

image
The image on which the color force field legend is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

opts
The *OE2DActiveSiteLegendDisplayOptions* object that determines how the legend is depicted.

The following code snippet shows how to use the *OEDrawActiveSiteLegend* function. The image created is shown in *Figure: Example of using the OEDrawActiveSiteLegend function*.

```
// initializing adisp OE2DActiveSiteDisplay object

uint rows = 2;
uint cols = 4;
OE2DActiveSiteLegendDisplayOptions opts = new OE2DActiveSiteLegendDisplayOptions(rows,␣
↪cols);

OEImage image = new OEImage(600.0, 150.0);
OEGrapheme.OEDrawActiveSiteLegend(image, adisp, opts);
```

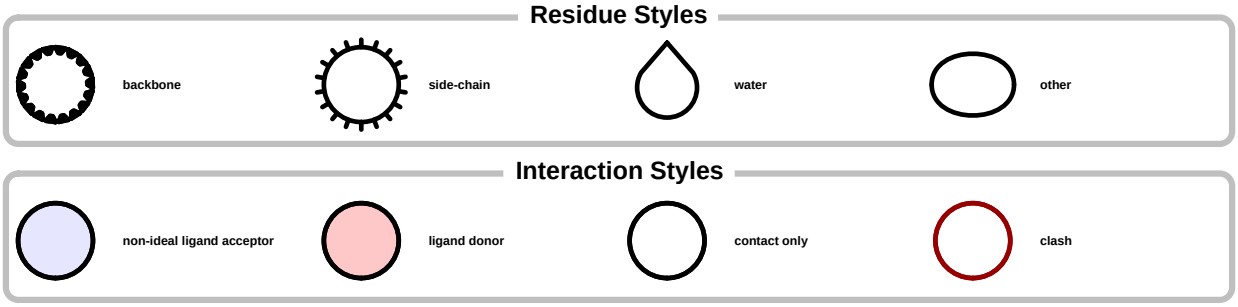


Fig. 112: **Example of using the OEDrawActiveSiteLegend function**

Note

The *OEDrawActiveSiteLegend* function renders only those residue "glyphs" that are used to depict the interactions of the given active site when calling the *OERenderActiveSite* function.

See also

- *OE2DActiveSiteDisplay* class
- *OE2DActiveSiteLegendDisplayOptions* class

Code Example

- Visualizing Protein-Ligand Interactions **OpenEye Python Cookbook** recipe
- *Depicting Active Site Interactions* example

4.4.9 OEDrawBFactorMapLegend

```
bool OEDrawBFactorMapLegend(OEDepict::OEImageBase& image,
                            const OE2DActiveSiteDisplay& adisp)
```

Draws the color gradient that is used to visualize B-factor information of an active site by the *OERenderBFactorMap* function.

image
The image in which the color gradient is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the B-factor information of an active site.

Example:

```
// initialize OE2DActiveSiteDisplay

OEImage cgimage = new OEImage(300.0, 100.0);
OEGrapheme.OEDrawBFactorMapLegend(cgimage, adisp);
OEDepict.OEWriteImage("DrawBFactorMapLegend.svg", cgimage);
```

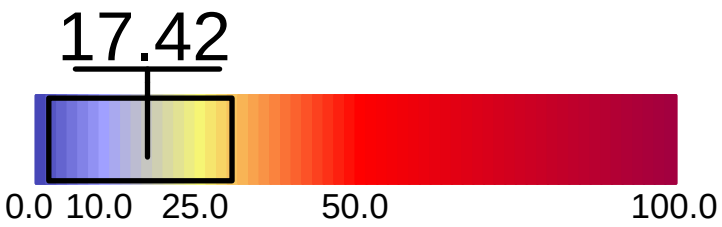


Fig. 113: **Example of using the OEDrawBFactorMapLegend function**

See also

- *OERenderBFactorMap* function
- *OEDrawColorGradient* function

Code Example

- Visualizing Protein-Ligand B-Factor **OpenEye Python Cookbook** recipe

4.4.10 OEDrawCircle

```
bool OEDrawCircle(OEDepict::OEImageBase& image, unsigned int style,
                  const OEDepict::OE2DPoint& center, double radius,
                  const OEDepict::OEPen& pen = OEDepict::OEBlackPen)
```

Draws a circle with the given style.

image
The image on which the circle is drawn.

style
Specifies the style of the circle. This value has to be from the *OECircleStyle* namespace.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

4.4.11 OEDrawColorForceFieldLegend

```
void OEDrawColorForceFieldLegend(OEDepict::OEImageBase& image,
                                 const OEColorForceFieldDisplay& cffdisp,
                                 const OEColorForceFieldLegendDisplayOptions& opts)
```

Draws a legend of the color force field.

image
The image on which the color force field legend is drawn.

cffdisp
The *OEColorForceFieldDisplay* object that holds the data necessary to depict the color force field.

opts
The *OEColorForceFieldLegendDisplayOptions* object that determine how the legend of color force field is depicted.

The following code snippet shows how to use the *OEDrawColorForceFieldLegend* function. The image created is shown in *Figure: Example of using the OEDrawColorForceFieldLegend function*.

```
OEColorForceField cff = new OEColorForceField();
cff.Init(OEColorFFType.ImplicitMillsDean);

OEColorForceFieldDisplay cffdisp = new OEColorForceFieldDisplay(cff);
uint rows = 2;
uint cols = 4;
OEColorForceFieldLegendDisplayOptions opts = new␣
↪OEColorForceFieldLegendDisplayOptions(rows, cols);
```

(continues on next page)

(continued from previous page)

```
OEImage image = new OEImage(450, 150);
OEGrapheme.OEDrawColorForceFieldLegend(image, cffdisp, opts);
```

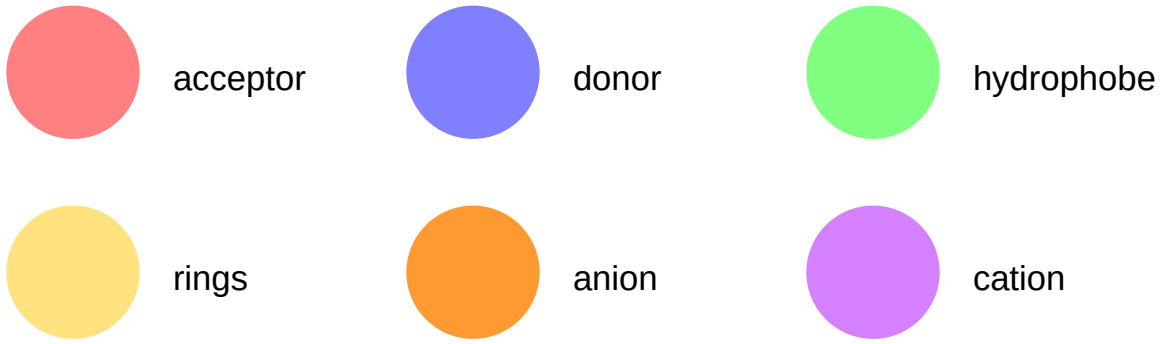


Fig. 114: **Example of using the OEDrawColorForceFieldLegend function**

See also

- *OEColorForceFieldDisplay* class
- *OEColorForceFieldLegendDisplayOptions* class

4.4.12 OEDrawColorGradient

```
void OEDrawColorGradient(OEDepict::OEImageBase& image,
                         const OESystem::OEColorGradientBase& colorg)
void OEDrawColorGradient(OEDepict::OEImageBase& image,
                         const OESystem::OEColorGradientBase& colorg,
                         const OEColorGradientDisplayOptions& opts)
```

Draws a color gradient.

image
The image on which the color gradient is drawn.

colorg
The color gradient being drawn.

opts
The *OEColorGradientDisplayOptions* object that stores the parameters that determine how the color gradient is drawn.

The following code snippet shows how to use the `OEDrawColorGradient` function. The image created is shown in *Figure: Example of using the OEDrawColorGradient function*.

```
OELinearColorGradient colorg = new OELinearColorGradient();
colorg.AddStop(new OEColorStop( 0.0, OEChem.OEYellow));
colorg.AddStop(new OEColorStop(+1.0, OEChem.OEOrange));
```

(continues on next page)

(continued from previous page)

```
colorg.AddStop(new OEColorStop(-1.0, OEChem.OEGreen));

OEImage image = new OEImage(400, 100);
OEGrapheme.OEDrawColorGradient(image, colorg);
OEDepict.OEWriteImage("DrawColorGradient.png", image);
```

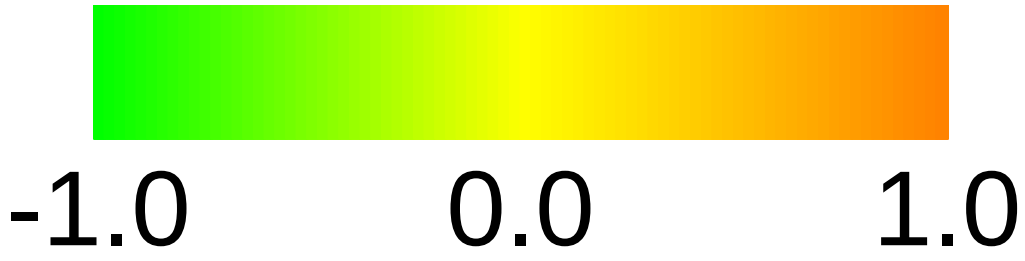


Fig. 115: **Example of using the OEDrawColorGradient function**

The orientation (*i.e.*, whether the color gradient is rendered vertically or horizontally) depends on the width and height of the OEImageBase object. If the width is greater than the height, then the color gradient is drawn horizontally; otherwise it is rendered vertically . See examples in *Figure: Example of rendering the same color gradient into different dimensions*.

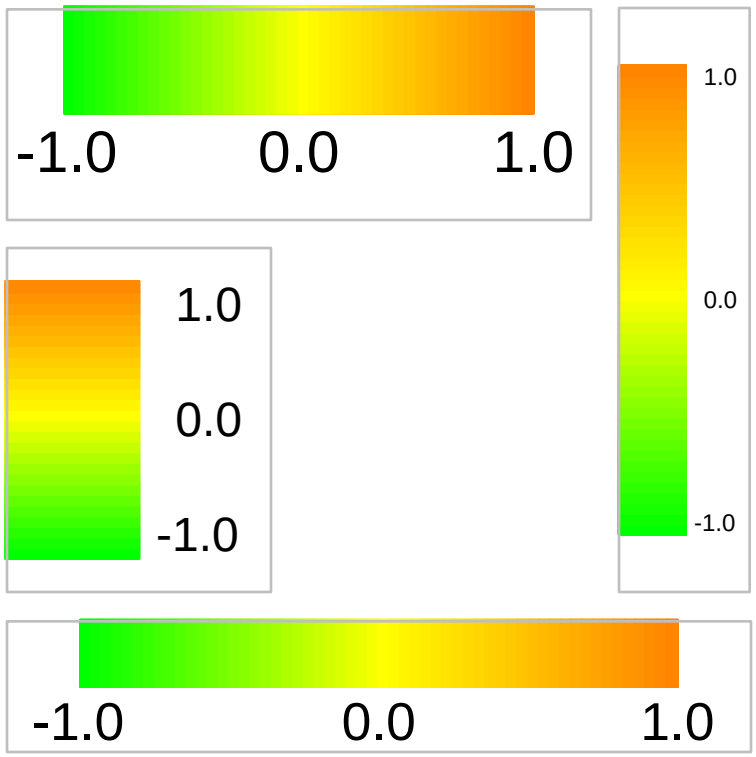


Fig. 116: **Example of rendering the same color gradient into different dimensions**

The *OEDrawColorGradient* function can depict any color gradient derived from the OEColorGradientBase base class.

See examples in *Figure: Example of rendering various color gradients*.

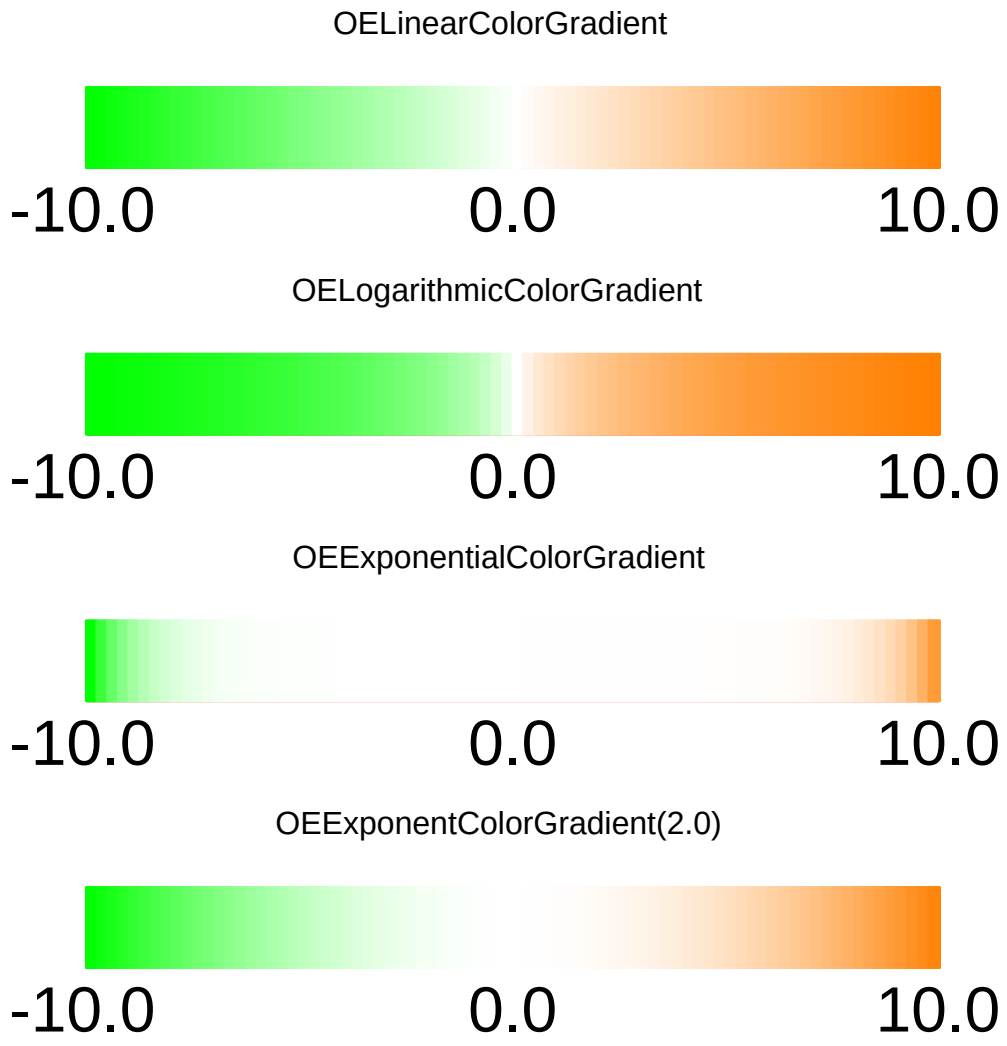


Fig. 117: **Example of rendering various color gradients**

See also

- *OEColorGradientDisplayOptions* class
- OEColorGradientBase base class
- OELinearColorGradient, OEExponentialColorGradient, OELogarithmicColorGradient, and OEExponentColorGradient concrete classes

4.4.13 OEDrawPeptide

```
bool OEDrawPeptide(OEDepict::OEImageBase& image,
                   const OEChem::OEMolBase& mol)
```

(continues on next page)

(continued from previous page)

```
bool OEDrawPeptide(OEDepict::OEImageBase& image,
                   const OEChem::OEMolBase& mol,
                   const OEPeptideDisplayOptions& opts)
```

Draws a molecule after performing substitutions in which standard amino acids components identified in the molecule are replaced by corresponding circular glyphs.

image
The image on which the molecule is drawn.

mol
The molecule being depicted.

opts
The *OEPeptideDisplayOptions* object that stores properties that determine the styles of the peptide depiction.

Note

- The *OEDrawPeptide* implementation does not support visualizing large proteins. The number of heavy atoms is limited to 250.
- The *OEDrawPeptide* only support standard L-amino acids and there D-amino acid enantiomers.

Example:

The following code snippet shows how to use the *OEDrawPeptide* function to generate an interactive SVG image. The image created is shown in *Figure: Example of using the OEDrawPeptide function*.

```
oemolistream ifs = new oemolistream();

uint flavor = OEIFlavor.Generic.Default |
              OEIFlavor.FASTA.EmbeddedSMILES;

ifs.SetFlavor(OEFormat.FASTA, flavor);
ifs.SetFormat(OEFormat.FASTA);

string fasta = ">Custom FASTA\nFAVS[[R4]COCC(C(=O)O)Cc1ccccc1]";
byte[] fastaBytes = Encoding.ASCII.GetBytes(fasta);
ifs.OpenByteArray(fastaBytes, (uint)fastaBytes.Length);

OEGraphMol mol = new OEGraphMol();
OEChem.OEReadMolecule(ifs, mol);
ifs.close();

OEImage image = new OEImage(400, 250);

OEPeptideDisplayOptions opts = new OEPeptideDisplayOptions();
opts.SetInteractive(true);
OEGrapheme.OEDrawPeptide(image, mol, opts);
OEDepict.OEWriteImage("DrawPeptide.svg", image);
```

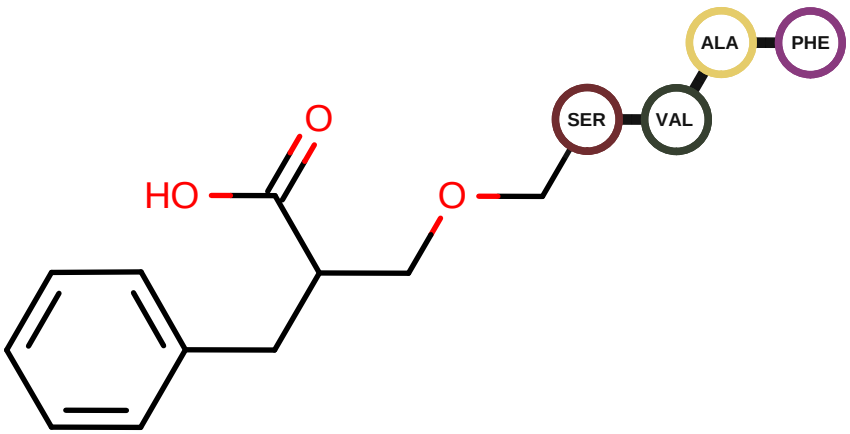


Fig. 118: **Example of using the OEDrawPeptide function**

Note

This *interactive* functionality is **only available** for `.svg` image format.

The generated `svg` image should be included into and HTML page with the SVG MIME type.

```
<object data="<imagename>.svg" type="image/svg+xml"></object>
```

See also

- *OEPeptideDisplayOptions* class
- *OEDrawResidues* function to depict residues of peptides
- *OEDrawMonomerGraph* and *OEHighlightMonomers* functions to depict molecules initialized from HELM
- Generating Interactive SVG Images chapter in **OEDepict TK** manual

4.4.14 OEDrawResidues

```
bool OEDrawResidues(OEDepict::OEImageBase& image,
                    const OEChem::OEMolBase& mol,
                    const bool interactive = false)
```

The *OEDrawResidues* function depicts the residue graph of a molecule with up to 300 heavy atoms. It is recommended to use when a molecule is imported from a PDB file that automatically assigns residues (`OEResidue`) to each atom.

image
The image on which the molecule is drawn.

mol
The molecule being depicted.

interactive
The parameter that determines whether the generated image will be interactive (only available for `SVG` images) revealing the residue on mouse hover.

Note

The *OEDrawResidues* does not support visualizing large proteins. The number of heavy atoms is limited to 250.

Example:

The following code snippet shows how to use the *OEDrawResidues* function to generate an interactive SVG image. The image created is shown in *Figure: Example of using the OEDrawResidues function*. If the molecule has no residue information, if the OEHasResidues function returns `false`, then a warning will be throw and no image will be generated.

The *OEDrawResidues* function allows the visualization of nonstandard amino acids. Nonstandard amino acid residues are represented with a black jagged circle with the residue labels shown at the center.

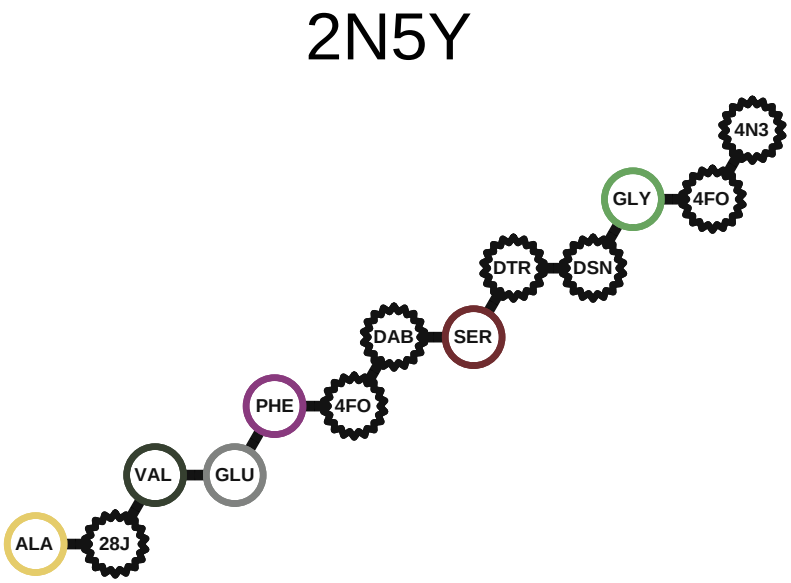


Fig. 119: **Example of using the OEDrawResidues function**

Note

This *interactive* functionality is **only available** for `.svg` image format.

The generated `svg` image should be included into and HTML page with the SVG MIME type.

```
<object data="<imagename>.svg" type="image/svg+xml"></object>
```

See also

- *OEDrawPeptide* function to depict a peptide with standard amino acids
- *OEDrawMonomerGraph* and *OEHighlightMonomers* functions to depict molecules initialized from HELM
- Generating Interactive SVG Images chapter in **OEDepict TK** manual

4.4.15 OEDrawROCSScores

```
void OEDrawROCSScores(OEDepict::OEImageBase& image,
                      const std::vector<double>& scores)
```

Renders the scores into a radial graph.

image
The image on which the radial graph of the scores is drawn.

scores
The vector of scores. The scores will expected in the following order - shape overlap score - color overlap score - 2D similarity score (optional)

Note

All scores are expected to be in the range of $[0.0, 1.0]$. This $[0.0, 1.0]$ range is transformed to $[0.0° - 360.0°]$ angles when representing the scores in the diagram.

```
OEImage image = new OEImage(200, 150);
OEDoubleVector scores = new OEDoubleVector();
scores.Add(0.75);
scores.Add(0.25);
scores.Add(0.50);
OEGrapheme.OEDrawROCSScores(image, scores);
OEDepict.OEWriteImage("DrawROCSScores.png", image);
```

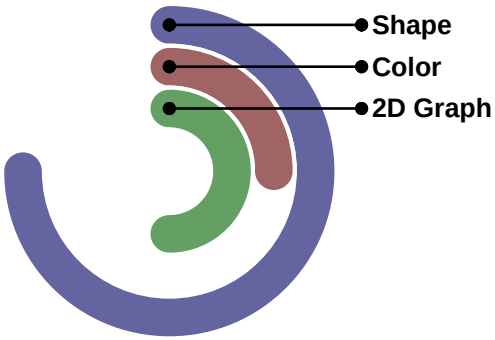


Fig. 120: **Example of using the OEDrawROCSScores function**

Code Example

- *Depicting Shape and Color Atom Overlaps* example
- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

4.4.16 OEDrawSurfaceArc

```
bool OEDrawSurfaceArc(OEDepict::OEImageBase& image,
                      unsigned int style,
                      const OEDepict::OE2DPoint& center,
                      double bgnAngle, double endAngle, double radius,
                      const OEDepict::OEPen& pen)
```

Draws an arc with the given style.

image
The image on which the arc is drawn.

style
Specifies the style of the arc. This value has to be from the *OESurfaceArcStyle* namespace.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

4.4.17 OEDrawUnpairedInteractionMapLegend

```
void OEDrawUnpairedInteractionMapLegend(OEDepict::OEImageBase& image,
                                        const OE2DActiveSiteDisplay& adisp,
                                        const OE2DActiveSiteLegendDisplayOptions& opts)
```

Draws legend for unpaired interaction map.

image
The image in which the color gradient is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the B-factor information of an active site.

opts
The *OE2DActiveSiteLegendDisplayOptions* object that determines how the legend is depicted.

Example:

```
// initailize OE2DActiveSiteDisplay

OEImage cgimage = new OEImage(400.0, 200.0);
OE2DActiveSiteLegendDisplayOptions lopts = new OE2DActiveSiteLegendDisplayOptions(3, 2);
OEGrapheme.OEDrawUnpairedInteractionMapLegend(cgimage, adisp, lopts);
OEDepict.OEWriteImage("DrawUnpairedMapLegend.svg", cgimage);
```

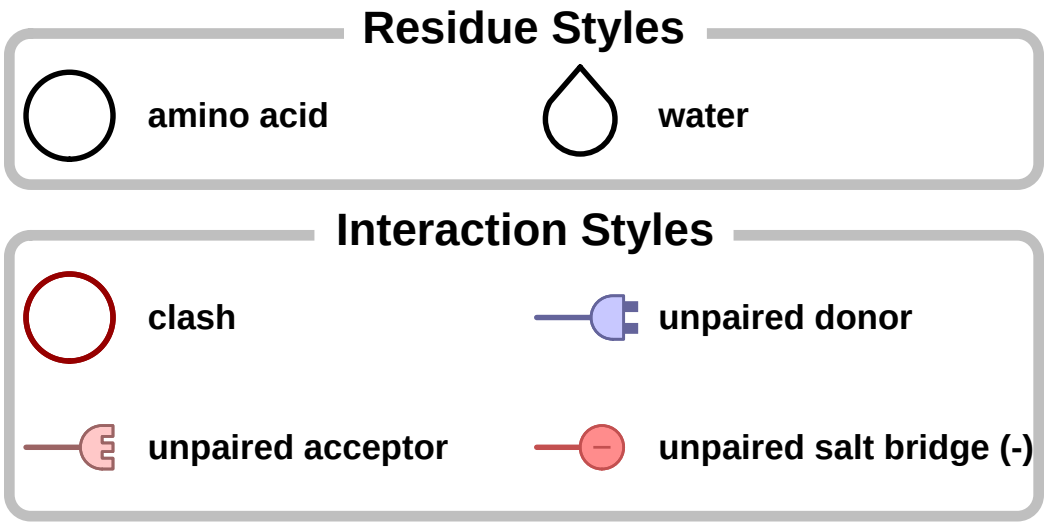


Fig. 121: **Example of using the OEDrawUnpairedInteractionMapLegend function**

See also

- *OERenderUnpairedInteractionMap* function

Code Example

- Visualizing Protein-Ligand Unpaired Interactions **OpenEye Python Cookbook** recipe

4.4.18 OEGet2DSurfaceArcs

```
OESystem::OEIterBase<OESurfaceArc>* OEGet2DSurfaceArcs(const OEDepict::OE2DMolDisplay&␣
↪disp,
                                                       double radiusScale =␣
↪OESurfaceArcScale::Default)
```

Returns an iterator over the *OESurfaceArc* objects from which a continuous 2D molecule surface can be drawn. See *Figure: Example of the data stored in the OESurfaceArc class*.

disp
The OE2DMolDisplay object for which the arcs of the 2D molecule surface are generated.

radiusScale
The multiplier used to modify the radius of the returned arcs. The scaling factor has to be in the range of

[*OESurfaceArcScale.Minimum*, *OESurfaceArcScale.Maximum*].

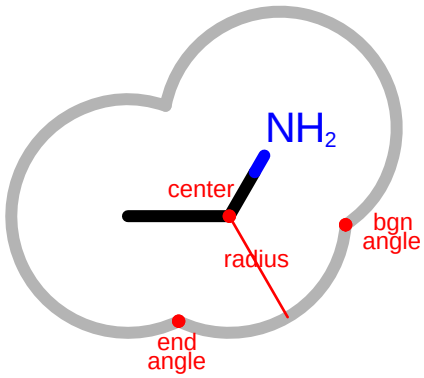


Fig. 122: **Example of the data stored in the OESurfaceArc class**

```
OESystem::OEIterBase<OESurfaceArc>* OEGet2DSurfaceArcs(const OEDepict::OE2DMolDisplay&␣
↪mdisp,
                                                       const OEDepict::OE2DAtomDisplay&␣
↪adisp,
                                                       double radiusScale =␣
↪OESurfaceArcScale::Default)
```

Returns an iterator over the *OESurfaceArc* object(s) generated for the given atom display. See *Figure: Example of the arcs returned for an atom display with various radius scales*.

mdisp
: The OE2DMolDisplay object to which the OE2DAtomDisplay object belongs.

adisp
: The OE2DAtomDisplay object for which the arc(s) of the 2D molecule surface are generated.

radiusScale
: The multiplier used to modify the radius of the returned arcs. The scaling factor has to be in the range of [*OESurfaceArcScale.Minimum*, *OESurfaceArcScale.Maximum*].

```
OESystem::OEIterBase<OESurfaceArc>* OEGet2DSurfaceArcs(const OEDepict::OE2DMolDisplay&␣
↪disp,
                                                       const std::vector<double>&␣
↪radiusScales)
```

Returns an iterator over the *OESurfaceArc* objects from which a continuous 2D molecule surface can be drawn.

disp
: The OE2DMolDisplay object for which the arcs of the 2D molecule surface are generated.

radiusScales
: The multipliers used to modify the radius of the returned arcs. The vector has to be `OEMolBase.GetMaxAtomIdx` long. Each scaling factor has to be in the range of [*OESurfaceArcScale.Minimum*, *OESurfaceArcScale.Maximum*].

See also

- *Drawing a Molecule Surface* chapter
- *OESurfaceArc* classes
- *OESurfaceArcScale* namespace

4.4.19 OEGetMoleculeSurfaceScale

```
double OEGetMoleculeSurfaceScale(const OEChem::OEMolBase& mol,
                                 const OEDepict::OE2DMolDisplayOptions& opts,
                                 double maxRadiusScale = OESurfaceArcScale::Default)
```

Estimates the scaling factor of the molecule depiction considering the size of the molecule surface being drawn around the molecule diagram. See *Figure: Example of using the OEGetMoleculeSurfaceScale function*.

mol
The molecule for which the scaling factor is determined.

opts
The OE2DMolDisplayOptions object that stores the parameters which influence the calculated scaling factor (such as the width and the height of the image).

maxRadiusScale
The maximum radius multiplier used to generate the arcs of the 2D molecule surface.

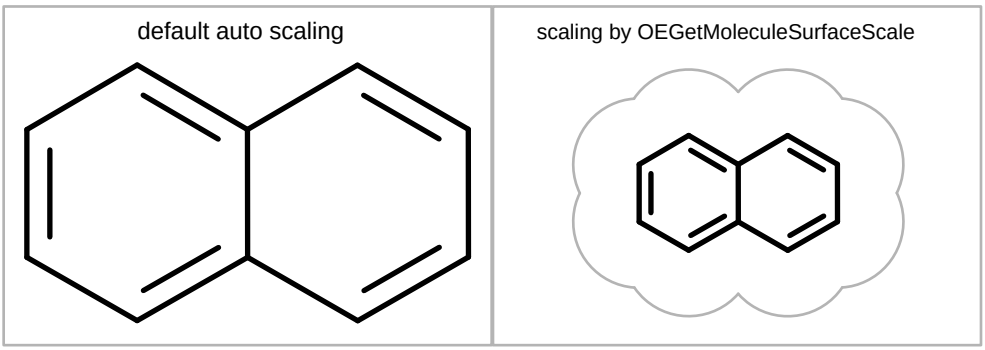


Fig. 123: **Example of using the OEGetMoleculeSurfaceScale function**

See also

- *OESurfaceArcScale* namespace

4.4.20 OEGetNearbyAtom

```
OEDepict::OENearestAtom OEGetNearbyAtom(const OE2DActiveSiteDisplay& disp,
                                        const OEDepict::OE2DPoint& point,
                                        double radiusscale = 1.0)
```

Returns an OENearestAtom object that stores a pointer of the atom which is depicted nearest to a given 2D display coordinates. By default, only atoms that not further than half bond away from the given point will be considered.

disp
The *OE2DActiveSiteDisplay* object of which atoms are being considered.

point
Specifies the 2D display coordinates.

radiusscale
The multiplier that can be used to increase or to decrease the radius around the given point to be considered. This value has to be in a range of $[0.25, 2.0]$.

Note

If there is no atom close to the given display coordinates, then the `OENearestBond.IsValid` method returns *false*.

See also

- OENearestAtom class
- *`OEGetNearestAtom`* function

4.4.21 OEGetNearbyBond

```
OEDepict::OENearestBond OEGetNearbyBond(const OE2DActiveSiteDisplay& disp,
                                        const OEDepict::OE2DPoint& point,
                                        double radiusscale = 1.0)
```

Returns an OENearestBond object that stores a pointer of the bond which is depicted nearest to a given 2D display coordinates. By default, only atoms that not further than half bond away from the given point will be considered.

disp
The *OE2DActiveSiteDisplay* object of which atoms are being considered.

point
Specifies the 2D display coordinates.

radiusscale
The multiplier that can be used to increase or to decrease the radius around the given point to be considered. This value has to be in a range of $[0.25, 2.0]$.

Note

If there is no bond close to the given display coordinates, then the `OENearestBond.IsValid` method returns *false*.

See also

- OENearestBond class
- *OEGetNearestBond* function

4.4.22 OEGetNearbyResidue

```
OENearestResidue OEGetNearbyResidue(const OE2DActiveSiteDisplay& disp,
                                    const OEDepict::OE2DPoint& point,
                                    double radiusscale = 1.0)
```

Returns an *OENearestResidue* object that stores the residue which is depicted nearest to a given 2D display coordinates. By default, only residues that not further than a radius distance from the given point will be considered.

disp
The *OE2DActiveSiteDisplay* object of which atoms are being considered.

point
Specifies the 2D display coordinates.

radiusscale
The multiplier that can be used to increase or to decrease the radius around the given point to be considered. This value has to be in a range of $[0.75, 3.0]$.

Note

If there is no residue close to the given display coordinates, then the *OENearestResidue.IsValid* method returns *false*.

See also

- *OENearestResidue* class
- *OEGetNearestResidue* function

4.4.23 OEGetNearestAtom

```
OEDepict::OENearestAtom OEGetNearestAtom(const OE2DActiveSiteDisplay& disp,
                                         const OEDepict::OE2DPoint& point)
```

Returns an OENearestAtom object that stores a pointer of the atom which is depicted nearest to a given 2D display coordinates.

disp
The *OE2DActiveSiteDisplay* object of which atoms are being considered.

point
Specifies the 2D display coordinates.

See also

- OENearestAtom class
- *OEGetNearbyAtom* function

4.4.24 OEGetNearestBond

```
OEDepict::OENearestBond OEGetNearestBond(const OE2DActiveSiteDisplay& disp,
                                         const OEDepict::OE2DPoint& point)
```

Returns an OENearestBond object that stores a pointer of the bond which is depicted nearest to a given 2D display coordinates.

disp
The *OE2DActiveSiteDisplay* object of which bonds are being considered.

point
Specifies the 2D display coordinates.

See also

- OENearestBond class
- *OEGetNearbyBond* function

4.4.25 OEGetNearestResidue

```
OENearestResidue OEGetNearestResidue(const OE2DActiveSiteDisplay& disp,
                                     const OEDepict::OE2DPoint& point)
```

Returns an *OENearestResidue* object that stores a residue which is depicted nearest to a given 2D display coordinates.

disp
The *OE2DActiveSiteDisplay* object of which residues are being considered.

point
Specifies the 2D display coordinates.

See also

- *OENearestResidue* class
- *OEGetNearbyResidue* function

4.4.26 OEGetSurfaceArcFxn

```
const OESurfaceArcFxnBase& OEGetSurfaceArcFxn(const OEChem::OEAtomBase* atom)
```

Returns the surface drawing functor that is associated with the given atom, *i.e.*, when calling the *OEDraw2DSurface* function this return functor is used to draw the arc segment of the given atom.

See also

- *OEClearSurfaceArcFxn* function
- *OEGet2DSurfaceArcs* function
- *OEHasSurfaceArcFxn* function
- *OESetSurfaceArcFxn* function

4.4.27 OEGraphemeGetArch

```
const char* OEGraphemeGetArch()
```

4.4.28 OEGraphemeGetLicensee

```
bool OEGraphemeGetLicensee(std::string& licensee)
```

4.4.29 OEGraphemeGetPlatform

```
const char* OEGraphemeGetPlatform()
```

4.4.30 OEGraphemeGetRelease

```
const char* OEGraphemeGetRelease()
```

4.4.31 OEGraphemeGetSite

```
bool OEGraphemeGetSite(std::string& site)
```

4.4.32 OEGraphemeGetVersion

```
unsigned int OEGraphemeGetVersion()
```

4.4.33 OEGraphemeIsLicensed

```
bool OEGraphemeIsLicensed(const char* feature = 0,
                          unsigned int* expdate = 0)
```

Determines whether a valid license file is present. This function may be called without a legitimate run-time license to determine whether it is safe to call any **Grapheme TK** functionality.

The 'feature' argument can be used to check for a valid license to **Grapheme TK** along with the following features: 'python', 'java', or 'clr' (for CSharp).

The 'expdate' argument can be used to obtain the expiration date of the current **Grapheme TK** license.

See also

- OEChemIsLicensed function

4.4.34 OEHasSurfaceArcFxn

```
bool OEHasSurfaceArcFxn(const OEChem::OEAtomBase* atom)
```

Returns whether or not a surface drawing functor is associated with the given atom.

See also

- *OEClearSurfaceArcFxn* function
- *OEGetSurfaceArcFxn* function
- *OESetSurfaceArcFxn* function

4.4.35 OEPrepareActiveSiteDepiction

```
bool OEPrepareActiveSiteDepiction(OEBio::OEInteractionHintContainer& asite)
```

Prepares an active site before depiction that involves suppressing hydrogens and calculate distances between the accessible surfaces of the ligand and the surrounding protein.

See also

- *Depicting Active Site Interactions* example

4.4.36 OEPrepareAlignedDepictionFrom3D

Table 5: **The overloaded versions of the OEPrepareAlignedDepiction-From3D function**

Link	Description
OEPrepareAlignedDepiction-From3D(fitmol2D, fitmol3D, refmol2D, refmol3D)	2D molecule alignment based on 3D
OEPrepareAlignedDepiction-From3D(fitmol2D, fitmol3D, refmol2D, refmol3D, opts)	2D molecule alignment based on 3D with options
OEPrepareAlignedDepiction-From3D(fitmol2D, fitmol3D, shapequery)	2D molecule alignment to 3D shape query

```
bool OEPrepareAlignedDepictionFrom3D(OEChem::OEMolBase& fitmol2D,
                                     const OEChem::OEMolBase& fitmol3D,
                                     const OEChem::OEMolBase& refmol2D,
                                     const OEChem::OEMolBase& refmol3D,
                                     bool clearCoords = true,
                                     bool suppressH = true)
```

Generates the 2D coordinates of a molecule driven by the 3D overlay of the 3D reference and the 3D fit molecules and the orientation of the 2D reference molecule.

fitmol2D
The molecule for which 2D coordinates are generated.

refmol2D
The 2D molecule whose orientation drives the 2D coordinate generation.

fitmol3D, refmol3D
The 3D molecule whose overlay drives the 2D coordinate generation.

clearCoords
If false and the *fitmol2D* has 2D coordinates, then these coordinates are used for the alignment. Otherwise the 2D coordinates of the fitted molecule are generated by calling the *OEPrepareDepictionFrom3D* function.

suppressH
If true, then explicit hydrogens are suppressed when generating the 2D coordinates of the fitted molecule. Only hydrogens that are necessary to faithfully represent tetrahedral stereochemistry will be kept.

```
bool OEPrepareAlignedDepictionFrom3D(OEChem::OEMolBase& fitmol2D,
                                     const OEChem::OEMolBase& fitmol3D,
                                     const OEChem::OEMolBase& refmol2D,
                                     const OEChem::OEMolBase& refmol3D,
                                     const OEAlignedDepictionFrom3DOptions& opts)
```

Generates the 2D coordinates of a molecule driven by the 3D overlay of the 3D reference and the 3D fit molecules and the orientation of the 2D reference molecule.

fitmol2D
The molecule for which 2D coordinates are generated.

refmol2D
The 2D molecule whose orientation drives the 2D coordinate generation.

fitmol3D, refmol3D
The 3D molecules whose overlay drives the 2D coordinate generation.

opts
The *OEAlignedDepictionFrom3DOptions* object that stores options controlling the generation of 2D coordinates.

The following code snippet shows how to use the *OEPrepareAlignedDepictionFrom3D* function. See generated images in *Table: 2D depiction based on 3D molecule overlay*.

```
OEGraphMol refmol2D = new OEGraphMol(refmol3D);
OEGrapheme.OEPrepareDepictionFrom3D(refmol2D);

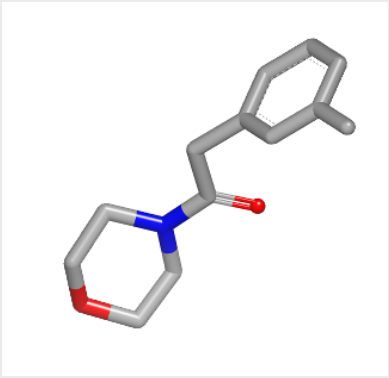
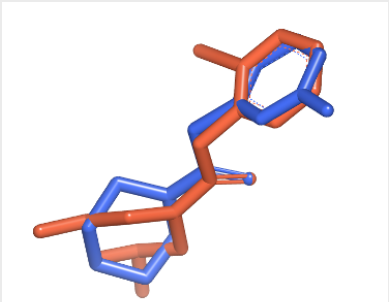
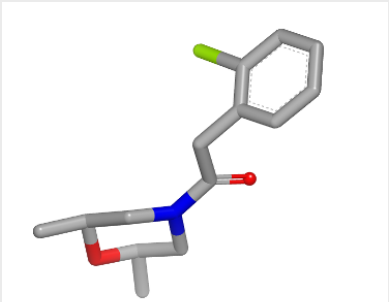
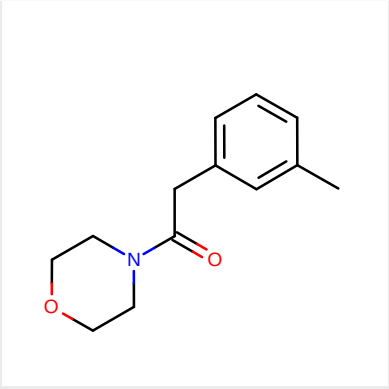
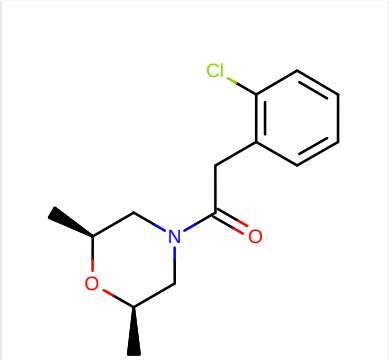
OEGraphMol fitmol2D = new OEGraphMol(fitmol3D);
OEGrapheme.OEPrepareAlignedDepictionFrom3D(fitmol2D, fitmol3D, refmol2D, refmol3D);


OE2DMolDisplayOptions opts = new OE2DMolDisplayOptions(width, height, OEScale.AutoScale);
double refscale = OEDepict.OEGetMoleculeScale(refmol2D, opts);
double fitscale = OEDepict.OEGetMoleculeScale(fitmol2D, opts);
opts.SetScale(Math.Min(refscale, fitscale));

OE2DMolDisplay refdisp = new OE2DMolDisplay(refmol2D, opts);
OEDepict.OERenderMolecule("OEPrepareAlignedDepictionFrom3D-ref-2D.png", refdisp);

OE2DMolDisplay fitdisp = new OE2DMolDisplay(fitmol2D, opts);
OEDepict.OERenderMolecule("OEPrepareAlignedDepictionFrom3D-fit-2D.png", fitdisp);
```

Table 6: **2D depiction based on 3D molecule overlay**

		
3D reference molecule	3D molecule overlay	3D fit molecule
		
2D reference molecule (generated by OEPrepareDepictionFrom3D)		2D fit molecule (**generated by OEPrepareAlignedDepiction-From3D**)

```
bool OEPrepareAlignedDepictionFrom3D(OEChem::OEMolBase& fitmol2D,
                                     const OEChem::OEMolBase& fitmol3D,
                                     const OEShapeQueryDisplay& qdisp)
```

Generates the 2D coordinates of a molecule driven by the 3D overlay of the shape query display object.

fitmol2D
The molecule for which 2D coordinates are generated.

refmol2D
The 2D molecule whose orientation drives the 2D coordinate generation.

qdisp
The *OEShapeQueryDisplay* object that drives the layout of the 2D coordinate generation.

See also

- *OEShapeQueryDisplay* class

4.4.37 OEPrepareDepictionFrom3D

```
bool OEPrepareDepictionFrom3D(OEChem::OEMolBase& mol,
                              bool suppressH = true)
```

Takes a 3D molecule and generates its 2D coordinates. The layout and orientation of the 2D coordinates are driven by the 3D coordinates. The molecule must have 3D coordinates assigned.

mol
The 3D molecule for which 2D coordinates are generated.

suppressH
If true, then explicit hydrogens are suppressed in the input molecule. Only hydrogens that are necessary to faithfully represent tetrahedral stereochemistry will be kept.

```
bool OEPrepareDepictionFrom3D(OEChem::OEMolBase& mol,
                              const OEDepictionFrom3DOptions& opts)
```

Takes a 3D molecule and generates its 2D coordinates using the given options. The layout and orientation of the 2D coordinates are driven by the 3D coordinates.

mol
The 3D molecule for which 2D coordinates are generated.

opts
The *OEDepictionFrom3DOptions* object that stores options controlling the generation of 2D coordinates.

See the 2D molecule depictions in *Table: 2D representation of the 3D molecule* and the corresponding 3D molecule in *Figure: Example of 3D molecule displayed in VIDA*.

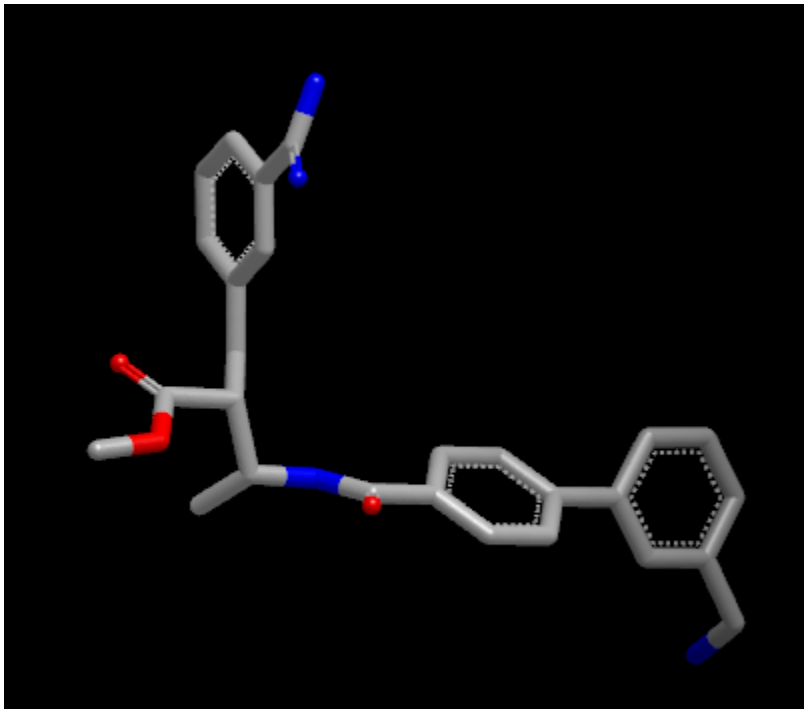
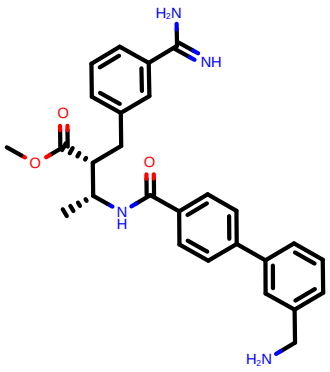
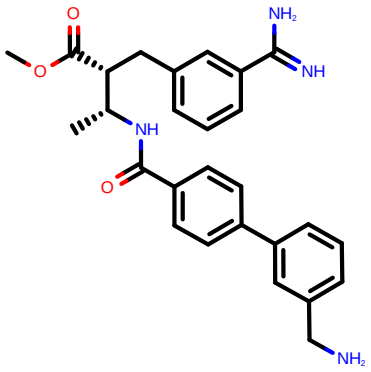


Fig. 124: **Example of 3D molecule displayed in VIDA**

Table 7: **2D representation of the 3D molecule**

OEPrepareDepictionFrom3D	OEPrepareDepiction
	

4.4.38 OERenderActiveSite

Table 8: **The overloaded versions of the OERenderActiveSite function**

Link	Description
OERenderActiveSite(image, adisp)	renders an active site into an image
OERenderActiveSite(filename, adisp)	writes an active site display into a file
OERenderActiveSite(stream, ext, adisp)	writes an active site display into an output stream

See also

- `OEDrawActiveSiteLegend` function
- `OERenderActiveSiteMaps` function
- `OERenderBFactorMap` function
- `OERenderContactMap` function
- `OERenderUnpairedInteractionMap` function

Code Example

- *Depicting Active Site Interactions* example
- Visualizing Protein-Ligand Interactions **OpenEye Python Cookbook** recipe

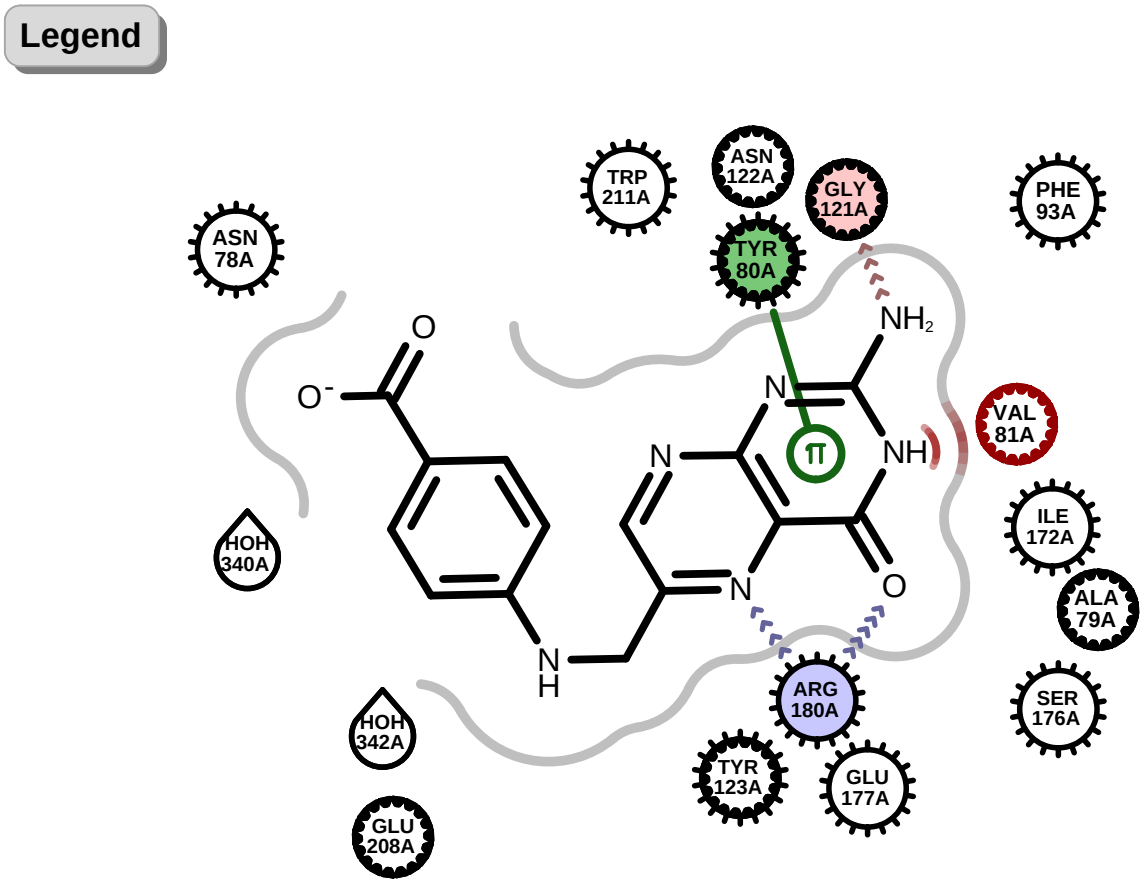


Fig. 125: **Example of using the OERenderActiveSite function**

```
bool OERenderActiveSite(OEDepict::OEImageBase& image,
                        const OE2DActiveSiteDisplay& adisp)
```

Renders the active site display into an image.

image
The image in which the active site is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

Example:

```
// initializing asite OEInteractionHintContainer(receptor, ligand) object

OEImage image = new OEImage(800.0, 600.0);
OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(image.GetWidth(),␣
↪image.GetHeight());
opts.SetRenderInteractiveLegend(true);
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);
```

(continues on next page)

(continued from previous page)

```
OEGrapheme.OERenderActiveSite(image, adisp);
OEDepict.OEWriteImage("OERenderActiveSite-image-adisp.png", image);
```

See also

- *OE2DActiveSiteDisplay* class
- *OE2DActiveSiteDisplayOptions* class

```
bool OERenderActiveSite(const std::string& filename,
                        const OE2DActiveSiteDisplay& adisp)
```

Writes the active site display into a file.

filename
The name of the file into which the active site display is being written. The extension of the filename determine the type of the image.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

Example:

```
// initializing asite OEInteractionHintContainer(receptor, ligand) object

OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(800.0, 600.0);
opts.SetRenderInteractiveLegend(true);
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

OEGrapheme.OERenderActiveSite("OERenderActiveSite-fname-adisp.png", adisp);
```

See also

- *OE2DActiveSiteDisplay* class
- *OE2DActiveSiteDisplayOptions* class

```
bool OERenderActiveSite(OEPlatform::oeostream& os,
                        const std::string& extension,
                        const OE2DActiveSiteDisplay& adisp)
```

Writes the active site display into a stream.

os
The stream into which the active site is being written.

ext
The extension which determine the type of the image.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

Example:

```
// initializing asite OEInteractionHintContainer(receptor, ligand) object

OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(800.0, 600.0);
opts.SetRenderInteractiveLegend(true);
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

oeofstream ofs = new oeofstream("OERenderActiveSite-stream-adisp.png");
OEGrapheme.OERenderActiveSite(ofs, "png", adisp);
ofs.close();
```

See also

- *OE2DActiveSiteDisplay* class
- *OE2DActiveSiteDisplayOptions* class

4.4.39 OERenderActiveSiteMaps

```
bool OERenderActiveSiteMaps(OEDepict::OEImageBase& image,
                            const OEBio::OEInteractionHintContainer& asite)
```

Renders four protein-ligand maps (interaction, unpaired, b-factor and contact) into separate tabs of an image.

image
The image in which the active site is drawn.

asite
The OEInteractionHintContainer object that holds the data for the active site.

Note

This functionality is **only available** for `.svg` image format.

Example:

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(receptor,␣
↪ligand);
OEBio.OEPerceiveInteractionHints(asite);
OEGrapheme.OEPrepareActiveSiteDepiction(asite);

OEImage image = new OEImage(800.0, 600.0);

OEGrapheme.OERenderActiveSiteMaps(image, asite);
OEDepict.OEWriteImage("RenderActiveSiteMap.svg", image);
```

There is no image available for the PDF version of this book.

See also

- *OERenderActiveSite* function
- *OERenderBFactorMap* function
- *OERenderContactMap* function
- *OERenderUnpairedInteractionMap* function

Code Example

- Visualizing Protein-Ligand Maps **OpenEye Python Cookbook** recipe

4.4.40 OERenderBFactorMap

```
bool OERenderBFactorMap(OEDepict::OEImageBase& image,
                        const OE2DActiveSiteDisplay& adisp)
```

Renders the B-factor information of an active site display into an image.

image
The image in which the active site is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the B-Factor information of an active site.

Example:

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(receptor,␣
↪ligand);
OEBio.OEPerceiveInteractionHints(asite);
OEGrapheme.OEPrepareActiveSiteDepiction(asite);

OEImage image = new OEImage(800.0, 600.0);
OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(image.
↪GetWidth(), image.GetHeight());
opts.SetRenderInteractiveLegend(true);
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

OEGrapheme.OERenderContactMap(image, adisp);
OEDepict.OEWriteImage("RenderContactMap.svg", image);
```

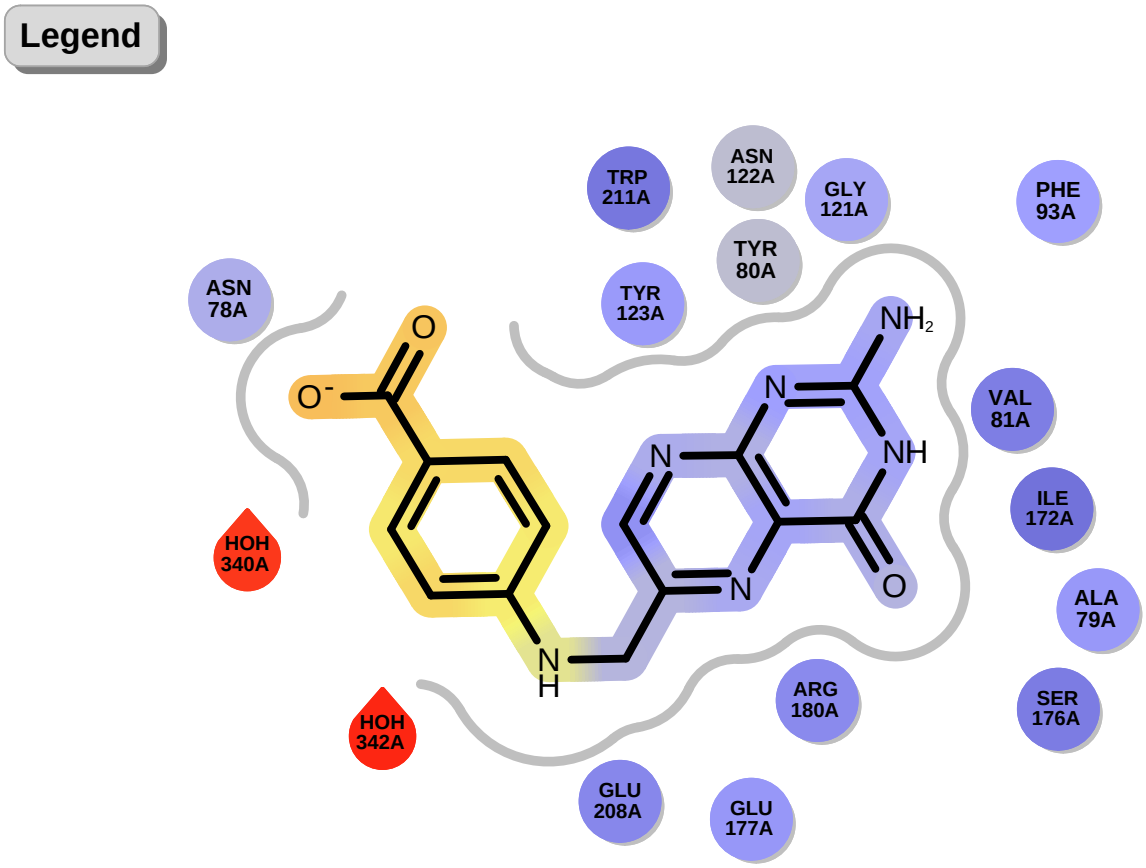


Fig. 126: **Example of using the OERenderBfactorMap function**

See also

- *OEDrawBFactorMapLegend* function
- *OERenderActiveSite* function
- *OERenderActiveSiteMaps* function
- *OERenderContactMap* function
- *OERenderUnpairedInteractionMap* function

Code Example

- Visualizing Protein-Ligand B-Factor **OpenEye Python Cookbook** example
- Visualizing Protein-Ligand B-Factor Map **OpenEye Python Cookbook** example

4.4.41 OERenderContactMap

```
bool OERenderContactMap(OEDepict::OEImageBase& image,
                        const OE2DActiveSiteDisplay& adisp)
```

Renders the protein-ligand contact map with *residue* toggle buttons. Clicking on any residue will highlight those ligand atoms that interact with the residue.

Note

This functionality is **only available** for `.svg` image format. In all other image formats it only depicts the residue circles.

The generated `svg` image should be included into and HTML page with the SVG MIME type.

```
<object data="<imagename>.svg" type="image/svg+xml"></object>
```

image
The image on which the contact map is rendered.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

The following code snippet shows how to use the *OERenderContactMap* function. The image created is shown in *Figure: Example of using the OERenderContactMap function*.

Example:

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(receptor, ligand);
OEBio.OEPerceiveInteractionHints(asite);
OEGrapheme.OEPrepareActiveSiteDepiction(asite);

OEImage image = new OEImage(800.0, 600.0);
OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(image.GetWidth(),␣
↪image.GetHeight());
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

OEGrapheme.OERenderContactMap(image, adisp);
OEDepict.OEWriteImage("RenderContactMap.svg", image);
```

There is no image available for the PDF version of this book.

See also

- *OERenderActiveSite* function
- *OERenderActiveSiteMaps* function
- *OERenderBFactorMap* function

- *OERenderUnpairedInteractionMap* function

Code Example

- Visualizing Protein-Ligand Contacts **OpenEye Python Cookbook** recipe

4.4.42 OERenderColorOverlap

```
void OERenderColorOverlap(OEDepict::OEImageBase& image,
                          const OEShapeOverlapDisplay& odisp)
```

Renders the color overlap between a reference and a fit molecule.

image
The image on which the fit molecule is rendered along with circles representing the satisfied and missed color atoms of the reference molecule.

odisp
The *OEShapeOverlapDisplay* object that holds the data necessary to depict the color overlap between the reference and the fit molecules.

The following code snippet shows how to use the *OERenderColorOverlap* function. The image created is shown in *Table: Example of using the OERenderColorOverlap function*.

```
OEColorForceField cff = new OEColorForceField();
cff.Init(OEColorFFType.ImplicitMillsDean);
OEShapeQueryDisplay qdisp = new OEShapeQueryDisplay(refmol, cff);

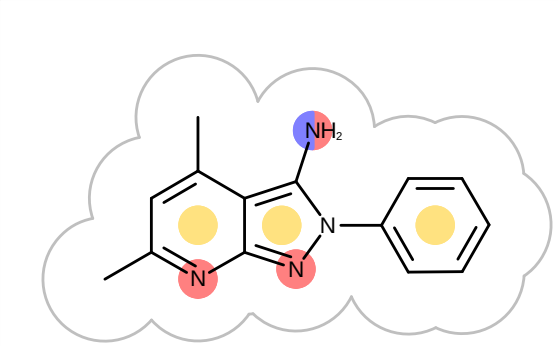
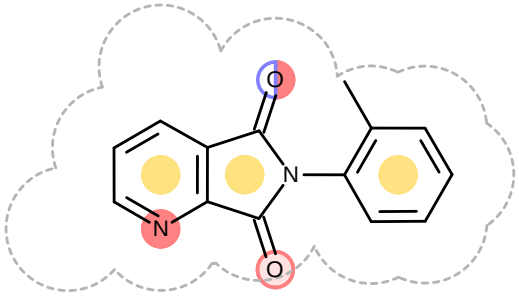
OEColorOverlapDisplayOptions opts = new OEColorOverlapDisplayOptions();
opts.SetTitleLocation(OETitleLocation.Hidden);
OEPen arcpen = new OEPen(OEChem.OEGrey, OEChem.OEGrey, OEFill.Off, 2.0, OEStipple.
↪ShortDash);
opts.SetQuerySurfaceArcFxn(new OEDefaultArcFxn(arcpen));
OEShapeOverlapDisplay odisp = new OEShapeOverlapDisplay(qdisp, fitmol, opts);

OEImage image = new OEImage(420.0, 280.0);
OEGrapheme.OERenderColorOverlap(image, odisp);
```

Note

The color atom matches between the reference and the fit molecule are represented by circles (or half circles if two color atoms occupy the same space in 3D). Each circle corresponds to a color atom in the reference molecule. The color of the circle indicates the fitness of the color atom match in 3D. The lighter the color, the smaller the overlap between the reference and fit color atoms in 3D. Unfilled circles represent unmatched reference color atoms. If there is a good color atom match exist for a reference color atom in 3D, then the circle representing the color atom is positioned to the matching fit color atom in 2D.

Table 9: **Example of using the OERenderColorOverlap function**

(A) reference molecule (shape query)	**(B)** fit molecule
	
shape query depicted by the **OERenderShapeQuery** function	color atom overlap depicted by the **OERenderColorOverlap** function

Warning

When calling the *OERenderColorOverlap* function **no overlay is performed to maximize the color overlap** between the reference and the fit molecule, *i.e.*, the function simply calculates the color overlap using the given 3D coordinates.

See also

- *OEColorOverlapDisplayOptions* class
- *OERenderShapeQuery* function
- *OERenderShapeOverlap* function

Code Example

- *Depicting Shape and Color Atom Overlaps* example
- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

4.4.43 OERenderRamachandranPlot

```
bool OERenderRamachandranPlot(OEDepict::OEImageBase& image,
                              const OERamachandranPlot& plot)
```

Renders the six Ramachandran plots into separate tabs of the given image.

image
The image on which the Ramachandran plots are rendered.

plot
The *OERamachandranPlot* object that stores the data of a Ramachandran plot.

Note

This functionality is **only available** for `.svg` image format.

Example:

```
OEImage image = new OEImage(800, 600);

OERamachandranPlot ramaplot = new OERamachandranPlot();
ramaplot.AddMolecule(protein);

OEGrapheme.OERenderRamachandranPlot(image, ramaplot);
```

4.4.44 OERenderShapeOverlap

Table 10: **The overloaded versions of the OERenderShapeOverlap function**

Link	Description
OERenderShapeOverlap(image, overlapdisp)	render shape overlap
OERenderShapeOverlap(molimage, overlapimage, overlapdisp)	render shape overlap into separate images

```
void OERenderShapeOverlap(OEDepict::OEImageBase& image,
                          const OEShapeOverlapDisplay& odisp)
```

Renders the shape overlap between a reference and a fit molecule.

image
The image on which the fit molecule is rendered along with the property map and the molecule surface of the reference molecule representing the shape overlap.

odisp
: The *OEShapeOverlapDisplay* object that holds the data necessary to depict the shape overlap between the reference and the fit molecules.

The following code snippet shows how to use the `OERenderShapeOverlap` function. The image created is shown in *Table: Example of using the OERenderShapeOverlap function*.

```
OEColorForceField cff = new OEColorForceField();
cff.Init(OEColorFFType.ImplicitMillsDean);
OEShapeQueryDisplay qdisp = new OEShapeQueryDisplay(refmol, cff);

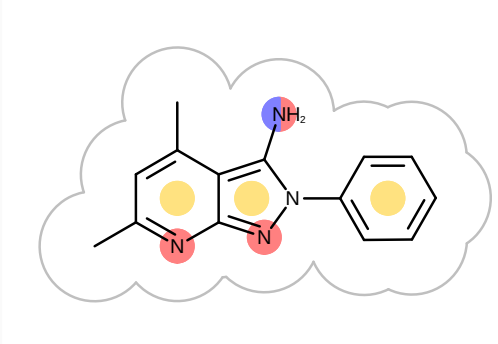
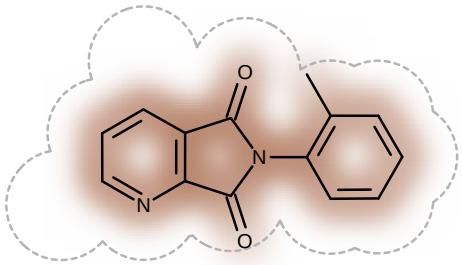
OEShapeOverlapDisplayOptions opts = new OEShapeOverlapDisplayOptions();
opts.SetTitleLocation(OETitleLocation.Hidden);
OEPen arcpen = new OEPen(OEChem.OEGrey, OEChem.OEGrey, OEFill.Off, 2.0,␣
↪OEStipple.ShortDash);
opts.SetQuerySurfaceArcFxn(new OEDefaultArcFxn(arcpen));
OEShapeOverlapDisplay odisp = new OEShapeOverlapDisplay(qdisp, fitmol, opts);

OEImage image = new OEImage(420.0, 280.0);
OEGrapheme.OERenderShapeOverlap(image, odisp);
```

Note

The shape overlap between the reference and the fit molecule is visualized by using a property map (*OE2DPropMap*), *i.e.*, a 2D grid, laid underneath the molecule structure, where the cells of the grid that are colored brown indicate good 3D shape overlap between the reference and the fit molecules. Additionally, clashes between the molecular graph of the fit molecule and 2D molecule surface of the reference structure imply shape mismatch in 3D.

Table 11: **Example of using the OERenderShapeOverlap function**

(A) reference molecule (shape query)	**(B)** fit molecule
	
shape query depicted by the **OERenderShapeQuery** function	shape overlap depicted by the **OERenderShapeOverlap** function

Warning

When calling the `OERenderShapeOverlap` function **no overlay is performed to maximize the color overlap** between the reference and the fit molecule, *i.e.*, the function simply calculates the shape overlap using the given 3D coordinates.

```
void OERenderShapeOverlap(OEDepict::OEImageBase& molimage,
                          OEDepict::OEImageBase& overlapimage,
                          const OEShapeOverlapDisplay& odisp)
```

Renders the shape overlap between a reference and a fit molecule into to separate images.

molimage
The image on which the fit molecule is rendered along with the surface of the reference molecule.

overlapimage
The image on which the property map is rendered representing the shape overlap.

odisp
The *OEShapeOverlapDisplay* object that holds the data necessary to depict the shape overlap between the reference and the fit molecules.

Note

The background of the molecule image is transparent. By superimposing the two images (molecule image on top), the result would be as visualizing the shape overlay in one image.

See also

- *OEShapeOverlapDisplayOptions* class
- *OERenderShapeQuery* function
- *OERenderColorOverlap* function

Code Example

- *Depicting Shape and Color Atom Overlaps* example
- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

4.4.45 OERenderShapeQuery

```
void OERenderShapeQuery(OEDepict::OEImageBase& image,
                        const OEShapeQueryDisplay& disp)
```

Renders the reference molecule of the shape overlap.

image
The image on which the reference molecule is rendered.

disp
The *OEShapeQueryDisplay* object that holds the data necessary to depict the reference molecule of a shape overlap.

The following code snippet shows how to use the *OERenderShapeQuery* function. The image created is shown in *Figure: Example of using the OERenderShapeQuery function*.

```
OEShapeQueryDisplayOptions opts = new OEShapeQueryDisplayOptions();
opts.SetTitleLocation(OETitleLocation.Hidden);
OEPen arcpen = new OEPen(OEChem.OEWhite, OEChem.OELightGrey, OEFill.On, 2.0);
opts.SetSurfaceArcFxn(new OEDefaultArcFxn(arcpen));

OEColorForceField cff = new OEColorForceField();
cff.Init(OEColorFFType.ImplicitMillsDean);
OEShapeQueryDisplay qdisp = new OEShapeQueryDisplay(refmol, cff, opts);

OEImage image = new OEImage(420.0, 280.0);
OEGrapheme.OERenderShapeQuery(image, qdisp);
```

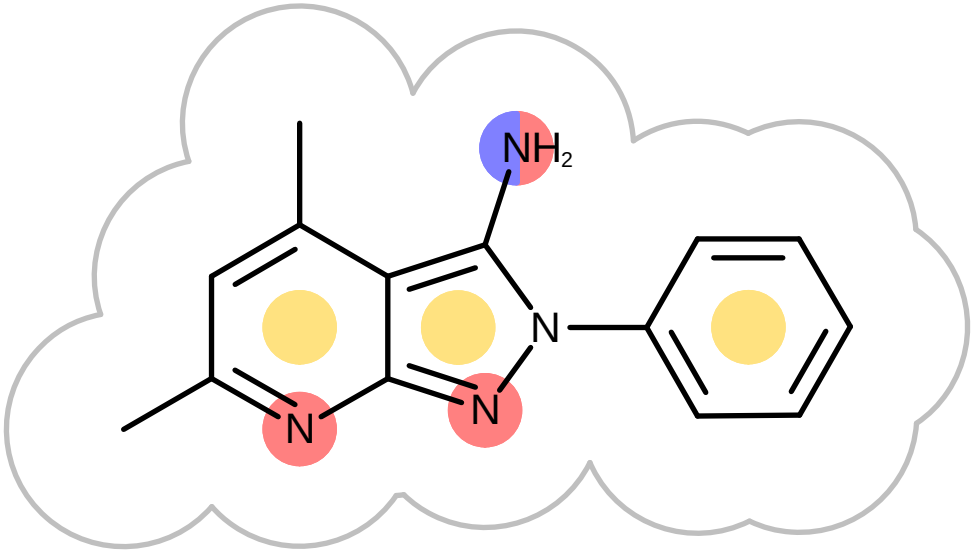


Fig. 127: **Example of using the OERenderShapeQuery function**

See also

- *OEShapeQueryDisplayOptions* class
- *OERenderColorOverlap* function
- *OERenderShapeOverlap* function

Code Example

- *Depicting Shape and Color Atom Overlaps* example
- Visualizing Shape and Color Overlap **OpenEye Python Cookbook** recipe

4.4.46 OERenderUnpairedInteractionMap

```
bool OERenderUnpairedInteractionMap(OEDepict::OEImageBase& image,
                                    const OE2DActiveSiteDisplay& adisp)
```

Renders the protein-ligand map with clash and unpaired interactions.

image
The image in which the active site is drawn.

adisp
The *OE2DActiveSiteDisplay* object that holds the data necessary to depict the interactions of an active site.

The *OERenderUnpairedInteractionMap* function currently visualizes the following interactions:

- Atom **clash** interaction (see OEClashInteractionHint class)
- **Unpaired** and **clash** types of the **hydrogen bonding** interaction (see OEHBondInteractionHint class `OEHBondInteractionHintType` namespace)
- **Unpaired** types of the **salt-bridge** interaction (see OESaltBridgeInteractionHint class and `OESaltBridgeInteractionHintType` namespace)

The following code snippet shows how to use the *OERenderUnpairedInteractionMap* function. The image created is shown in *Figure: Example of using the OERenderUnpairedInteraction function*.

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(receptor, ligand);
OEBio.OEPerceiveInteractionHints(asite);
OEGrapheme.OEPrepareActiveSiteDepiction(asite);

OEImage image = new OEImage(800.0, 600.0);
OE2DActiveSiteDisplayOptions opts = new OE2DActiveSiteDisplayOptions(image.GetWidth(),␣
↪image.GetHeight());
OE2DActiveSiteDisplay adisp = new OE2DActiveSiteDisplay(asite, opts);

OEGrapheme.OERenderUnpairedInteractionMap(image, adisp);
OEDepict.OEWriteImage("RenderUnpairedMap.svg", image);
```

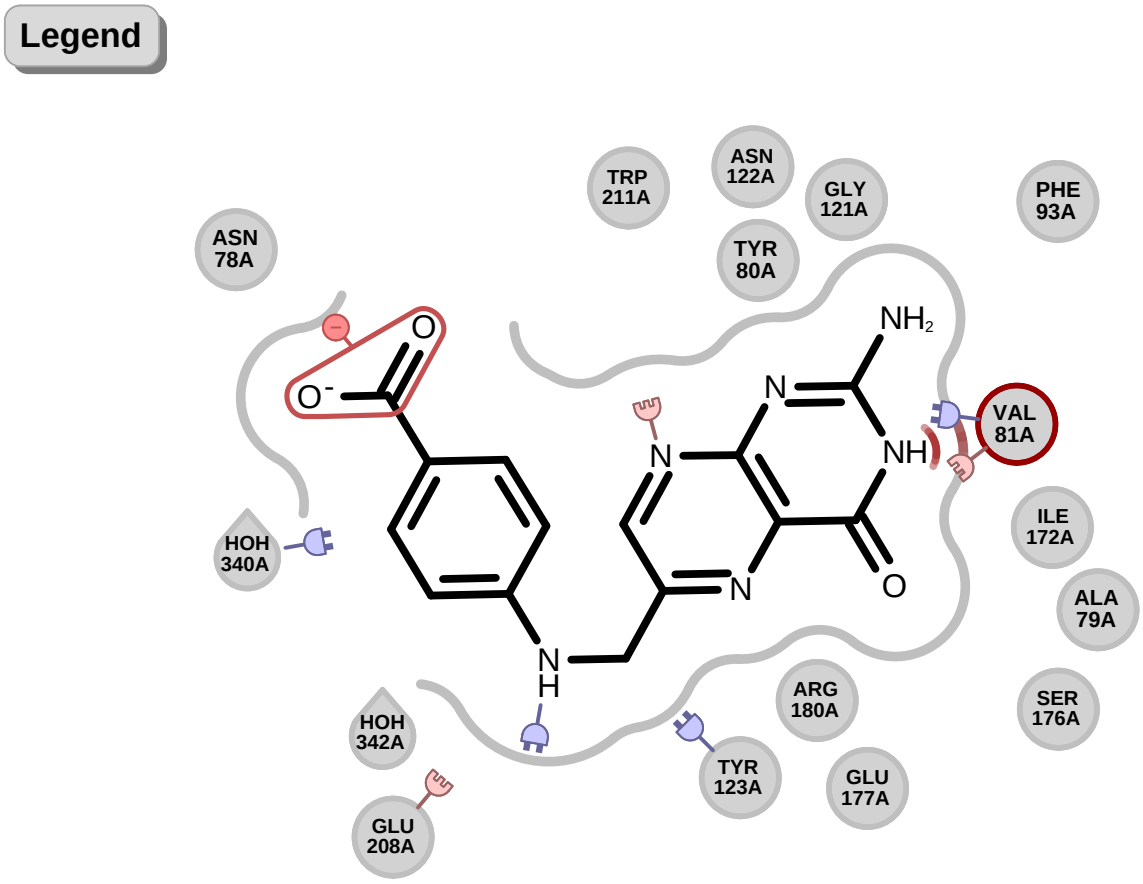


Fig. 128: **Example of using the OERenderUnpairedInteraction function**

Note

Please note that unpaired interactions have no direction in 3D, therefore the *linker* representing these unpaired interactions in 2D have no real spatial meaning either. Ligand linkers are directed away from the ligand, while protein linkers are directed towards the ligand when rendered in 2D.

See also

- *OEDrawUnpairedInteractionMapLegend* function
- *OERenderActiveSite* function
- *OERenderActiveSiteMaps* function
- *OERenderContactMap* function
- *OERenderBFactorMap* function

Code Example

- Visualizing Protein-Ligand Unpaired Interactions **OpenEye Python Cookbook** recipe

4.4.47 OESetSurfaceArcFxn

```
bool OESetSurfaceArcFxn(OEChem::OEMolBase& mol,
                        OEChem::OEAtomBase* atom,
                        const OESurfaceArcFxnBase& func)
```

Attaches the arc drawing functor to the given atom.

mol
The molecule the atom belongs to.

atom
The atom to which the functor will be attached.

func
The functor that will be used to draw the arc segment(s) of the molecule surface that correspond(s) to the given atom when the *OEDraw2DSurface* function in invoked.

See also

- *OEClearSurfaceArcFxn* function
- *OEGetSurfaceArcFxn* function
- *OEHasSurfaceArcFxn* function

4.4.48 OESetup2DPropMap

```
bool OESetup2DPropMap(OE2DPropMap& propmap,
                      const OESystem::OEInterface& itf)
```

Initializes an *OE2DPropMap* object from the parameters of a given interface.

4.5 OEGrapheme Basic Circle and Arc Drawing Functions

4.5.1 OEDrawAlphaRainbowCircle

```
bool OEDrawAlphaRainbowCircle(OEDepict::OEImageBase& image,
                              const OEDepict::OE2DPoint& center,
                              double radius,
                              const OEDepict::OEPen& pen,
                              unsigned int patternDirection =␣
↪OEPatternDirection::Outside)
```

Draws a circle with the *OECircleStyle.AlphaRainbow* style. See *Figure: OEDrawAlphaRainbowCircle*.

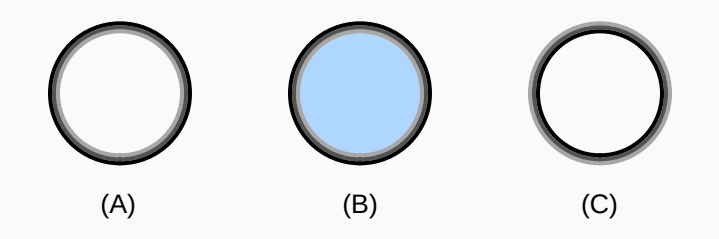


Fig. 129: **OEDrawAlphaRainbowCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See example (A) and (B) in *Figure: OEDrawAlphaRainbowCircle*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the circle. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawAlphaRainbowCircle*.

4.5.2 OEDrawAlphaRainbowSurfaceArc

```
bool OEDrawAlphaRainbowSurfaceArc(OEDepict::OEImageBase& image,
                                  const OEDepict::OE2DPoint& center,
                                  double bgnAngle, double endAngle,
                                  double radius,
                                  const OEDepict::OEPen& pen,
                                  double edgeAngle = 10.0,
                                  unsigned int patternDirection =␣
↪OEPatternDirection::Outside)
```

Draws an arc with the `OESurfaceArcStyle.AlphaRainbow` style. See *Figure: OEDrawAlphaRainbowSurfaceArc*.

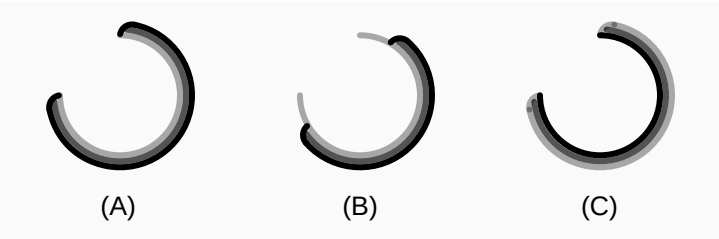


Fig. 130: **OEDrawAlphaRainbowSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawAlphaRainbowSurfaceArc*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the arc. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawAlphaRainbowSurfaceArc*.

See also

- *OEAlphaRainbowArcFxn* class

4.5.3 OEDrawBrickRoadCircle

```
bool OEDrawBrickRoadCircle(OEDepict::OEImageBase& image,
                           const OEDepict::OE2DPoint& center,
                           double radius, const OEDepict::OEPen& pen,
                           unsigned int patternDirection = OEPatternDirection::Outside,
                           double patternAngle = 20.0,
                           double patternWidthRatio = 0.15)
```

Draws a circle with the `OECircleStyle.BrickRoad` style. See *Figure: OEDrawBrickRoadCircle*.

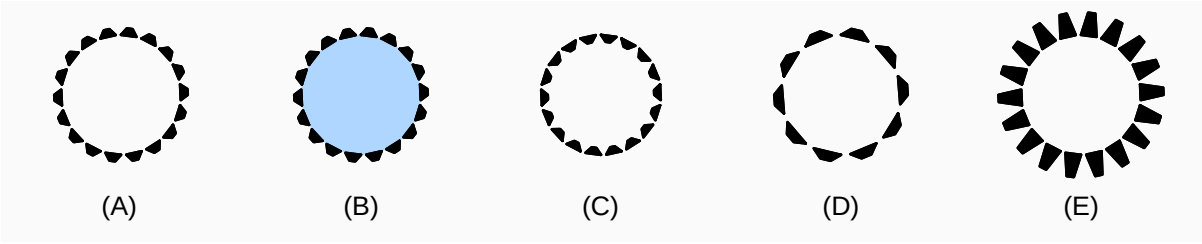


Fig. 131: **OEDrawBrickRoadCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawBrickRoadCircle*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the circle. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawBrickRoadCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawBrickRoadCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (E) in *Figure: OEDrawBrickRoadCircle*.

4.5.4 OEDrawBrickRoadSurfaceArc

```
bool OEDrawBrickRoadSurfaceArc(OEDepict::OEImageBase& image,
                               const OEDepict::OE2DPoint& center,
                               double bgnAngle, double endAngle,
                               double radius,
                               const OEDepict::OEPen& pen,
                               double edgeAngle = 10.0,
                               unsigned int patternDirection =␣
↪OEPatternDirection::Outside,
                               double patternAngle = 20.0,
                               double minPatternWidthRatio = 0.15,
                               double maxPatternWidthRatio = 0.15)
```

Draws an arc with the `OESurfaceArcStyle.BrickRoad` style.

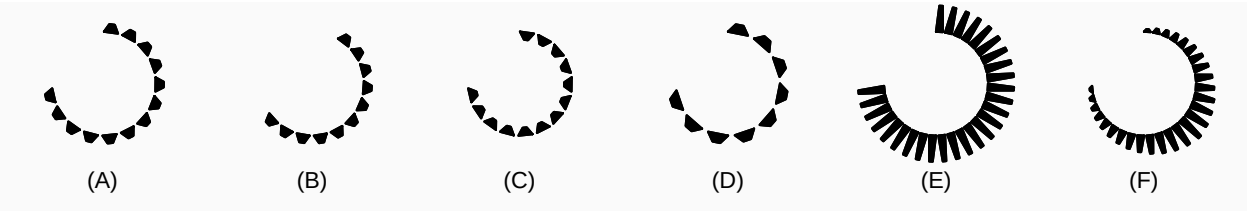


Fig. 132: **OEDrawBrickRoadSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$.

Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawBrickRoadSurfaceArc*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the arc. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawBrickRoadSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawBrickRoadSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (E) and (F) in *Figure: OEDrawBrickRoadSurfaceArc*.

See also

- *OEBrickRoadArcFxn* class

4.5.5 OEDrawCastleCircle

```
bool OEDrawCastleCircle(OEDepict::OEImageBase& image,
                        const OEDepict::OE2DPoint& center,
                        double radius, const OEDepict::OEPen& pen,
                        double patternAngle = 15.0,
                        double patternWidthRatio = 0.20)
```

Draws a circle with the `OECircleStyle.Castle` style.

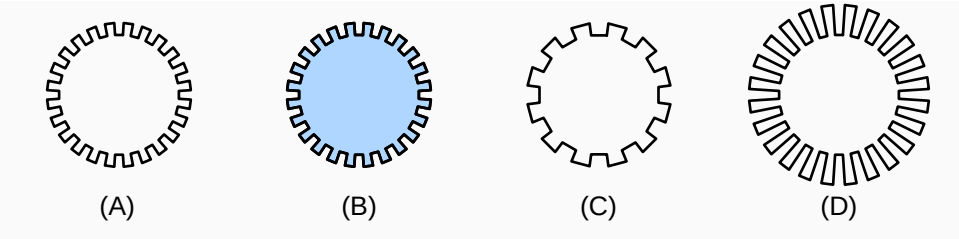


Fig. 133: **OEDrawCastleCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawCastleCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawCastleCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawCastleCircle*.

4.5.6 OEDrawCastleSurfaceArc

```
bool OEDrawCastleSurfaceArc(OEDepict::OEImageBase& image,
                            const OEDepict::OE2DPoint& center,
                            double bgnAngle, double endAngle,
                            double radius, const OEDepict::OEPen& pen,
                            double edgeAngle = 10.0,
                            double patternAngle = 15.0,
                            double minPatternWidthRatio = 0.20,
                            double maxPatternWidthRatio = 0.20)
```

Draws an arc with the *OESurfaceArcStyle.Castle* style.

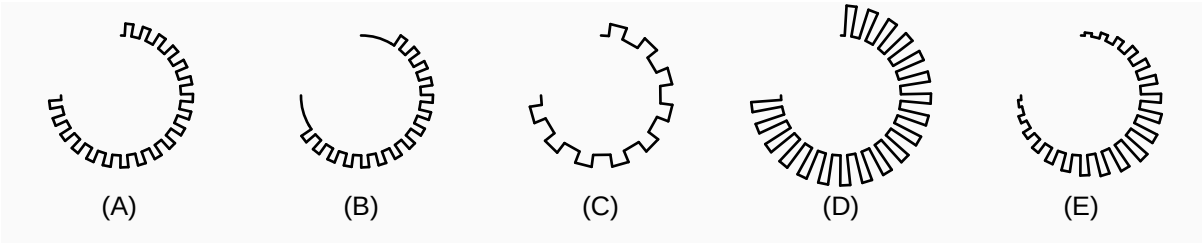


Fig. 134: **OEDrawCastleSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawCastleSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawCastleSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawCastleSurfaceArc*.

See also

- *OECastleArcFxn* class

4.5.7 OEDrawCogCircle

```
bool OEDrawCogCircle(OEDepict::OEImageBase& image,
                     const OEDepict::OE2DPoint& center, double radius,
                     const OEDepict::OEPen& pen,
                     double patternAngle = 15.0,
                     double patternWidthRatio = 0.20)
```

Draws a circle with the *OECircleStyle.Cog* style.

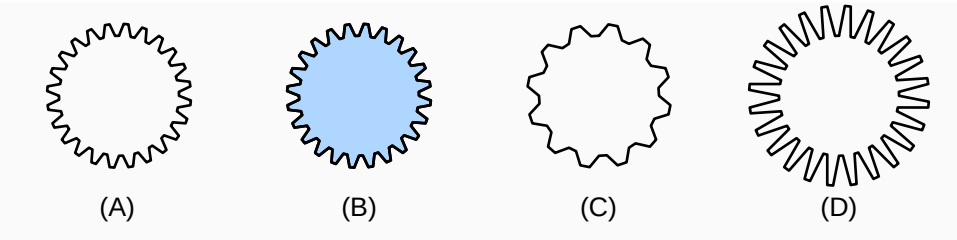


Fig. 135: **OEDrawCogCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawCogCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawCogCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawCogCircle*.

4.5.8 OEDrawCogSurfaceArc

```
bool OEDrawCogSurfaceArc(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double bgnAngle, double endAngle,
                         double radius, const OEDepict::OEPen& pen,
                         double edgeAngle = 10.0,
                         double patternAngle = 15.0,
                         double minPatternWidthRatio = 0.20,
                         double maxPatternWidthRatio = 0.20)
```

Draws an arc with the `OESurfaceArcStyle.Cog` style.

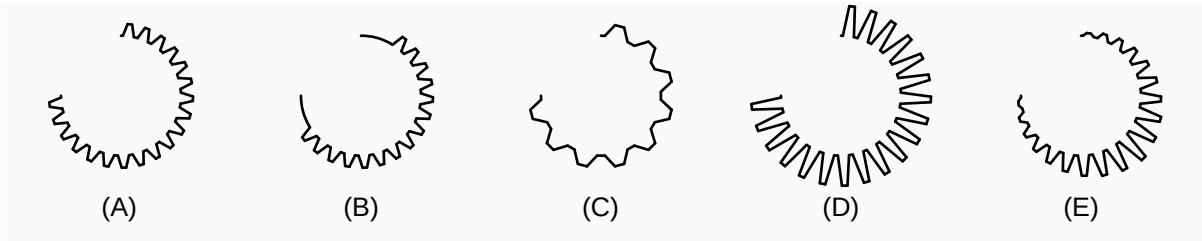


Fig. 136: **OEDrawCogSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawCogSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawCogSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawCogSurfaceArc*.

See also

- *OECogArcFxn* class

4.5.9 OEDrawDefaultCircle

```
bool OEDrawDefaultCircle(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double radius, const OEDepict::OEPen& pen)
```

Draws a circle with the *OECircleStyle.Default* style.

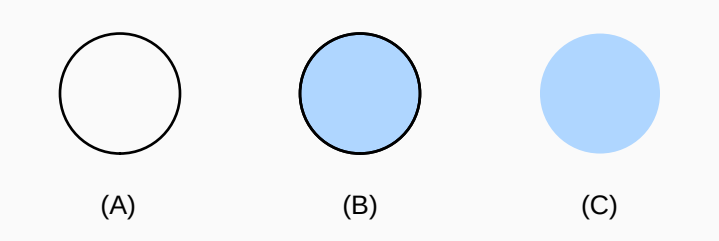


Fig. 137: **OEDrawDefaultCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples in *Figure: OEDrawDefaultCircle*.

4.5.10 OEDrawDefaultSurfaceArc

```
bool OEDrawDefaultSurfaceArc(OEDepict::OEImageBase& image,
                             const OEDepict::OE2DPoint& center,
                             double bgnAngle, double endAngle,
                             double radius, const OEDepict::OEPen& pen)
```

Draws an arc with the *OESurfaceArcStyle.Default* style.

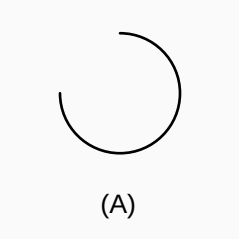


Fig. 138: **OEDrawDefaultSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

See also

- *OEDefaultArcFxn* class

4.5.11 OEDrawEyelashCircle

```
bool OEDrawEyelashCircle(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double radius, const OEDepict::OEPen& pen,
                         unsigned int patternDirection = OEPatternDirection::Outside,
                         double patternAngle = 10.0,
                         double patternWidthRatio = 0.15)
```

Draws a circle with the *OECircleStyle.Eyelash* style.

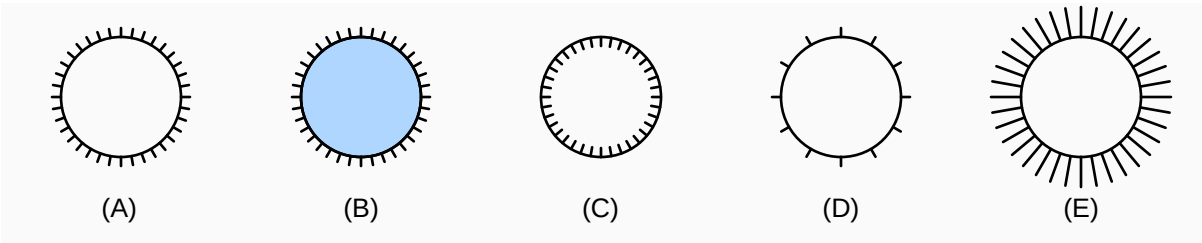


Fig. 139: **OEDrawEyelashCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawEyelashCircle*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the circle. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawEyelashCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawEyelashCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (E) in *Figure: OEDrawEyelashCircle*.

4.5.12 OEDrawEyelashSurfaceArc

```
bool OEDrawEyelashSurfaceArc(OEDepict::OEImageBase& image,
                             const OEDepict::OE2DPoint& center,
                             double bgnAngle, double endAngle,
                             double radius,
                             const OEDepict::OEPen& pen,
                             double edgeAngle = 10.0,
                             unsigned int patternDirection = OEPatternDirection::Outside,
                             double patternAngle = 10.0,
                             double minPatternWidthRatio = 0.15,
                             double maxPatternWidthRatio = 0.15)
```

Draws an arc with the `OESurfaceArcStyle.Eyelash` style.

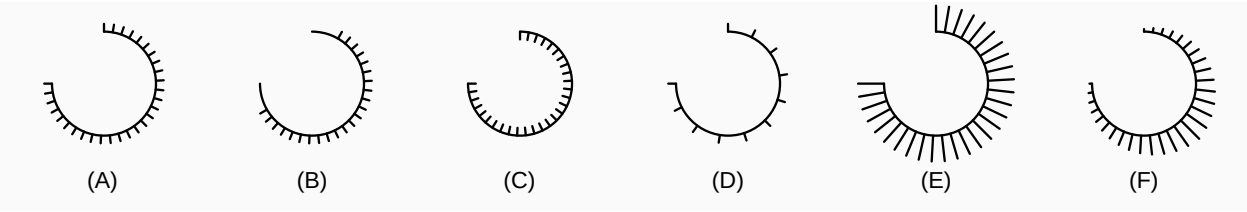


Fig. 140: **OEDrawEyelashSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawEyelashSurfaceArc*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the arc. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawEyelashSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawEyelashSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (E) and (F) in *Figure: OEDrawEyelashSurfaceArc*.

See also

- *OEEyelashArcFxn* class

4.5.13 OEDrawFlowerCircle

```
bool OEDrawFlowerCircle(OEDepict::OEImageBase& image,
                        const OEDepict::OE2DPoint& center,
                        double radius, const OEDepict::OEPen& pen,
                        double patternAngle = 30.0)
```

Draws a circle with the `OECircleStyle.Flower` style.

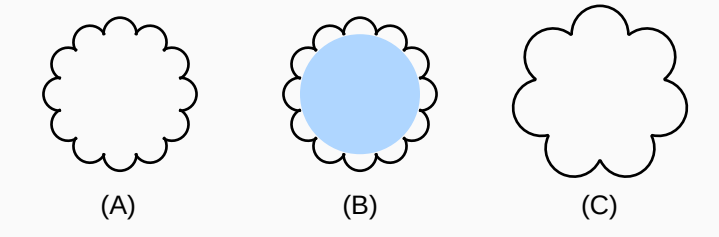


Fig. 141: **OEDrawFlowerCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawFlowerCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawFlowerCircle*.

4.5.14 OEDrawFlowerSurfaceArc

```
bool OEDrawFlowerSurfaceArc(OEDepict::OEImageBase& image,
                            const OEDepict::OE2DPoint& center,
                            double bgnAngle, double endAngle,
                            double radius, const OEDepict::OEPen& pen,
                            double edgeAngle = 10.0,
                            double patternAngle = 30.0,
                            double minPatternWidthRatio = 0.20,
                            double maxPatternWidthRatio = 0.20)
```

Draws an arc with the *OESurfaceArcStyle.Flower* style.

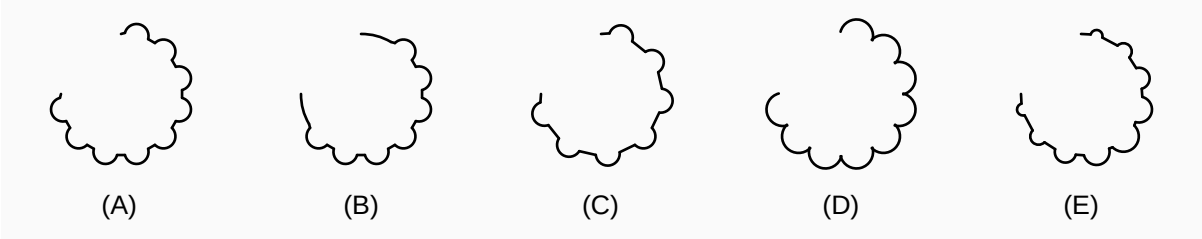


Fig. 142: **OEDrawFlowerSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawFlowerSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawFlowerSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawFlowerSurfaceArc*.

See also

- *OEFlowerArcFxn* class

4.5.15 OEDrawGreekKeyCircle

```
bool OEDrawGreekKeyCircle(OEDepict::OEImageBase& image,
                          const OEDepict::OE2DPoint& center,
                          double radius, const OEDepict::OEPen& pen,
                          double patternAngle = 15.0,
                          double patternWidthRatio = 0.20)
```

Draws a circle with the *OECircleStyle.GreekKey* style.

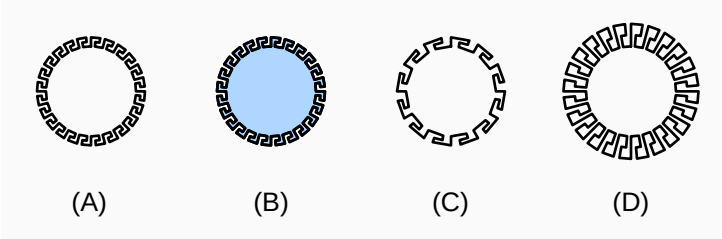


Fig. 143: **OEDrawGreekKeyCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawGreekKeyCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawGreekKeyCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawGreekKeyCircle*.

4.5.16 OEDrawNecklaceCircle

```
bool OEDrawNecklaceCircle(OEDepict::OEImageBase& image,
                          const OEDepict::OE2DPoint& center,
                          double radius, const OEDepict::OEPen& pen,
                          double patternAngle = 20.0,
                          double patternWidthRatio = 0.15)
```

Draws a circle with the `OECircleStyle.Necklace` style.

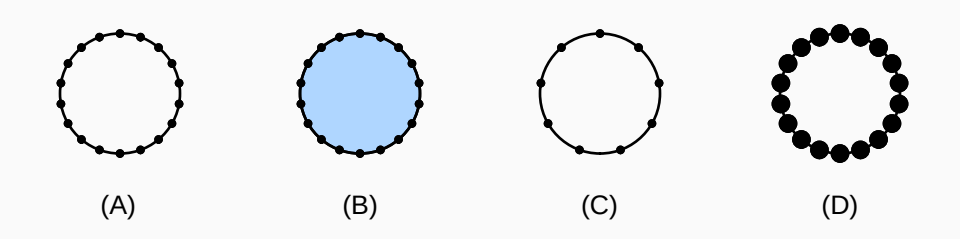


Fig. 144: **OEDrawNecklaceCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawNecklaceCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawNecklaceCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawNecklaceCircle*.

4.5.17 OEDrawNecklaceSurfaceArc

```
bool OEDrawNecklaceSurfaceArc(OEDepict::OEImageBase& image,
                              const OEDepict::OE2DPoint& center,
                              double bgnAngle, double endAngle,
                              double radius,
```

(continues on next page)

(continued from previous page)

```
                                const OEDepict::OEPen& pen,
                                double edgeAngle = 10.0,
                                double patternAngle = 20.0,
                                double minPatternWidthRatio = 0.15,
                                double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.Necklace* style.

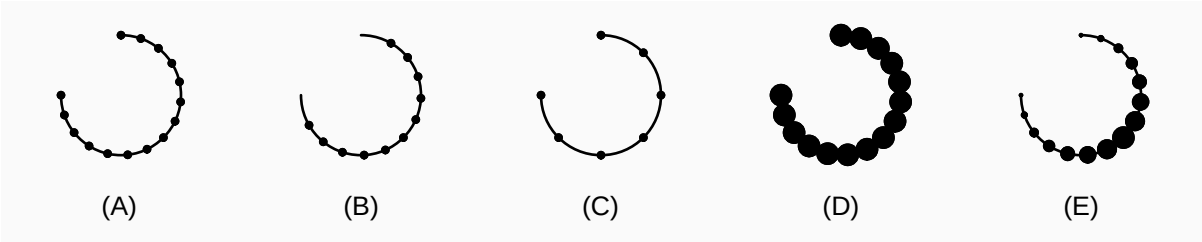


Fig. 145: **OEDrawNecklaceSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawNecklaceSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawNecklaceSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawNecklaceSurfaceArc*.

See also

- *OENecklaceArcFxn* class

4.5.18 OEDrawOliveBranchCircle

```
bool OEDrawOliveBranchCircle(OEDepict::OEImageBase& image,
                             const OEDepict::OE2DPoint& center,
                             double radius,
                             const OEDepict::OEPen& pen,
                             double patternAngle = 15.0,
                             double patternWidthRatio = 0.15)
```

Draws a circle with the *OECircleStyle.OliveBranch* style.

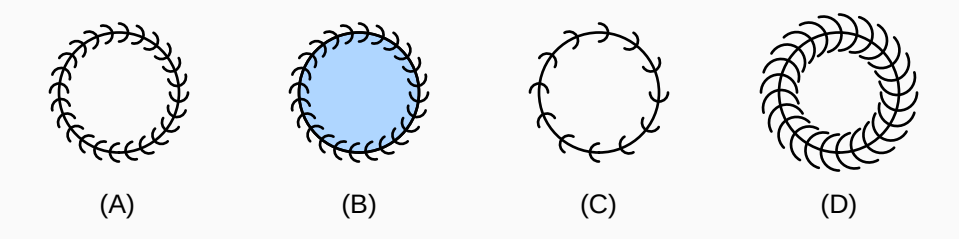


Fig. 146: **OEDrawOliveBranchCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawOliveBranchCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawOliveBranchCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawOliveBranchCircle*.

See also

- *OEOliveBranchArcFxn* class

4.5.19 OEDrawOliveBranchSurfaceArc

```
bool OEDrawOliveBranchSurfaceArc(OEDepict::OEImageBase& image,
                                 const OEDepict::OE2DPoint& center,
                                 double bgnAngle, double endAngle,
                                 double radius,
```

(continues on next page)

(continued from previous page)

```
                                    const OEDepict::OEPen& pen,
                                    double edgeAngle = 10.0,
                                    double patternAngle = 15.0,
                                    double minPatternWidthRatio = 0.15,
                                    double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.OliveBranch* style.

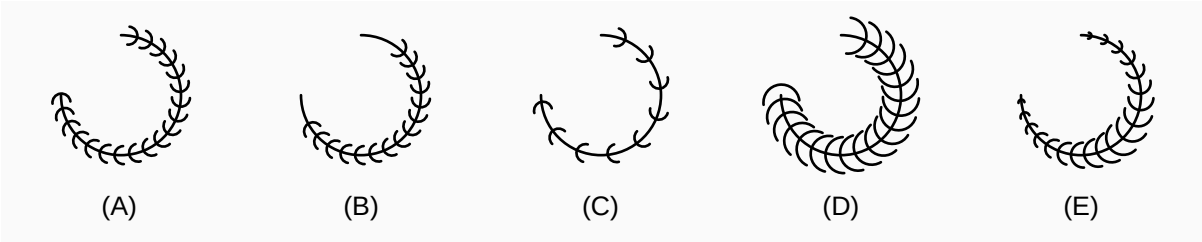


Fig. 147: **OEDrawOliveBranchSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawOliveBranchSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawOliveBranchSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawOliveBranchSurfaceArc*.

4.5.20 OEDrawPearlsCircle

```
bool OEDrawPearlsCircle(OEDepict::OEImageBase& image,
                        const OEDepict::OE2DPoint& center,
                        double radius, const OEDepict::OEPen& pen,
                        double patternAngle = 15.0)
```

Draws a circle with the *OECircleStyle.Pearls* style.

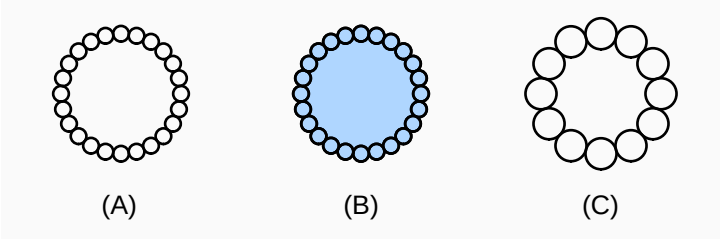


Fig. 148: **OEDrawPearlsCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawPearlsCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawPearlsCircle*.

4.5.21 OEDrawPearlsSurfaceArc

```
bool OEDrawPearlsSurfaceArc(OEDepict::OEImageBase& image,
                            const OEDepict::OE2DPoint& center,
                            double bgnAngle, double endAngle,
                            double radius, const OEDepict::OEPen& pen,
                            double edgeAngle = 10.0,
                            double patternAngle = 25.0,
                            double minPatternWidthRatio = 0.20,
                            double maxPatternWidthRatio = 0.20)
```

Draws an arc with the *OESurfaceArcStyle.Pearls* style.

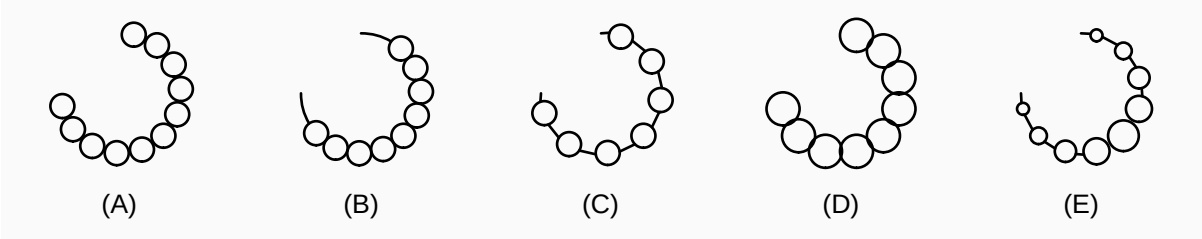


Fig. 149: **OEDrawPearlsSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawPearlsSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawPearlsSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawPearlsSurfaceArc*.

See also

- *OEPearlsArcFxn* class

4.5.22 OEDrawRaceTrackCircle

```
bool OEDrawRaceTrackCircle(OEDepict::OEImageBase& image,
                           const OEDepict::OE2DPoint& center,
                           double radius, const OEDepict::OEPen& pen)
```

Draws a circle with the *OECircleStyle.RaceTrack* style.

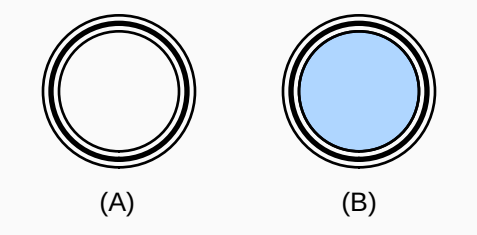


Fig. 150: **OEDrawRaceTrackCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawRaceTrackCircle*.

4.5.23 OEDrawRaceTrackSurfaceArc

```
bool OEDrawRaceTrackSurfaceArc(OEDepict::OEImageBase& image,
                               const OEDepict::OE2DPoint& center,
                               double bgnAngle, double endAngle,
                               double radius,
                               const OEDepict::OEPen& pen,
                               double edgeAngle = 10.0)
```

Draws an arc with the *OESurfaceArcStyle.RaceTrack* style.

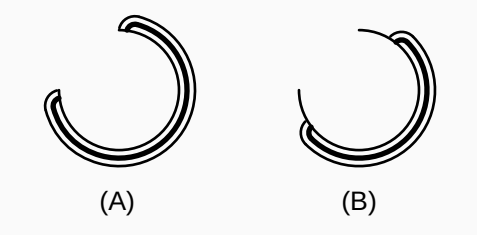


Fig. 151: **OEDrawRaceTrackSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawRaceTrackSurfaceArc*.

See also

- *OERaceTrackArcFxn* class

4.5.24 OEDrawRailroadTrackCircle

```
bool OEDrawRailroadTrackCircle(OEDepict::OEImageBase& image,
                               const OEDepict::OE2DPoint& center,
                               double radius,
                               const OEDepict::OEPen& pen,
                               double patternAngle = 10.0,
                               double patternWidthRatio = 0.15)
```

Draws a circle with the `OECircleStyle.RailroadTrack` style.

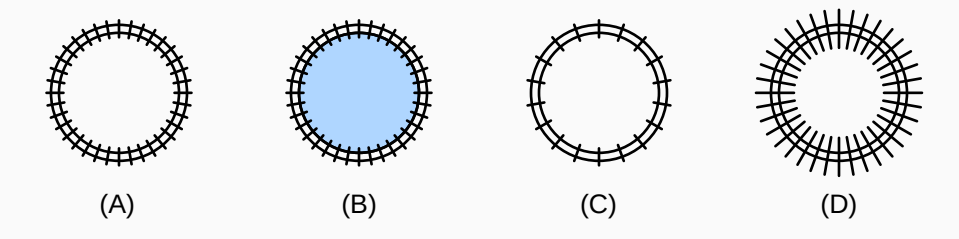


Fig. 152: **OEDrawRailroadTrackCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawRailroadTrackCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawRailroadTrackCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawRailroadTrackCircle*.

4.5.25 OEDrawRailroadTrackSurfaceArc

```
bool OEDrawRailroadTrackSurfaceArc(OEDepict::OEImageBase& image,
                                   const OEDepict::OE2DPoint& center,
                                   double bgnAngle, double endAngle,
                                   double radius,
                                   const OEDepict::OEPen& pen,
                                   double edgeAngle = 10.0,
                                   double patternAngle = 10.0,
                                   double minPatternWidthRatio = 0.15,
                                   double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.RailroadTrack* style.

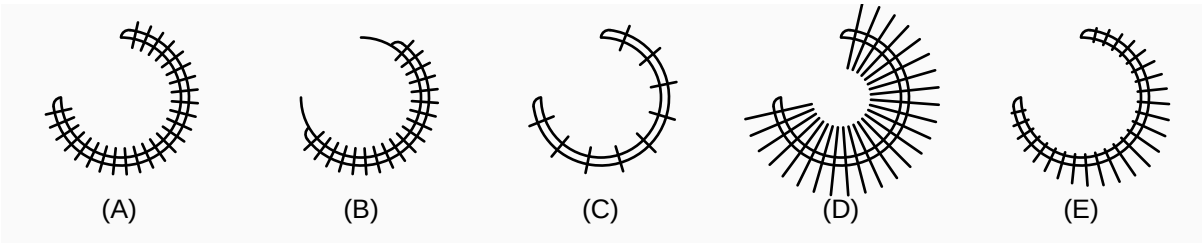


Fig. 153: **OEDrawRailroadTrackSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawRailroadTrackSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawRailroadTrackSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawRailroadTrackSurfaceArc*.

See also

- *OERailroadTrackArcFxn* class

4.5.26 OEDrawSawCircle

```
bool OEDrawSawCircle(OEDepict::OEImageBase& image,
                     const OEDepict::OE2DPoint& center, double radius,
                     const OEDepict::OEPen& pen,
                     unsigned int patternDirection = OEPatternDirection::Outside,
                     double patternAngle = 15.0,
                     double patternWidthRatio = 0.15)
```

Draws a circle with the `OECircleStyle.Saw` style.

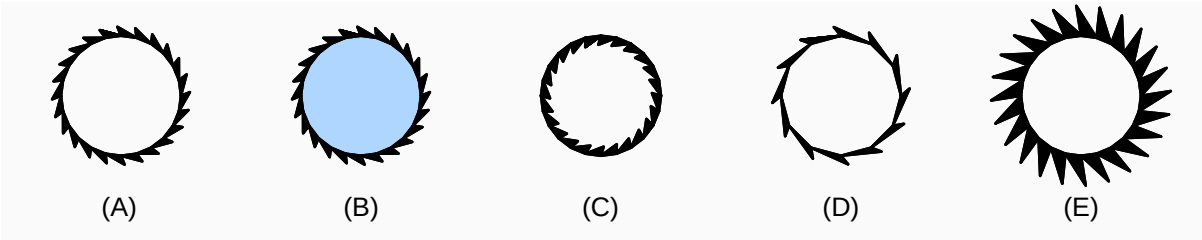


Fig. 154: **OEDrawSawCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawSawCircle*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the circle. This value has to be from the `OEPatternDirection` namespace.

See example (C) in *Figure: OEDrawSawCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawSawCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (E) in *Figure: OEDrawSawCircle*.

4.5.27 OEDrawSawSurfaceArc

```
bool OEDrawSawSurfaceArc(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double bgnAngle, double endAngle,
                         double radius, const OEDepict::OEPen& pen,
                         double edgeAngle = 10.0,
                         unsigned int patternDirection = OEPatternDirection::Outside,
                         double patternAngle = 15.0,
                         double minPatternWidthRatio = 0.15,
                         double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.Saw* style.

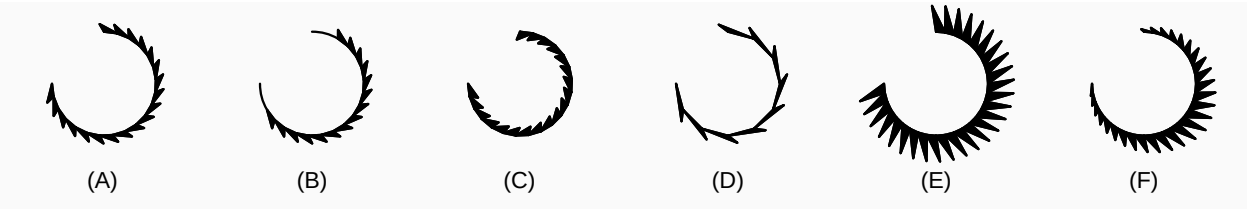


Fig. 155: **OEDrawSawSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawSawSurfaceArc*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the arc. This value has to be from the *OEPatternDirection* namespace.

See example (C) in *Figure: OEDrawSawSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawSawSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc

while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (E) and (F) in *Figure: OEDrawSawSurfaceArc*.

See also

- *OESawArcFxn* class

4.5.28 OEDrawSimpsonCircle

```
bool OEDrawSimpsonCircle(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double radius, const OEDepict::OEPen& pen,
                         double patternAngle = 10.0,
                         double patternWidthRatio = 0.20)
```

Draws a circle with the *OECircleStyle.Simpson* style.

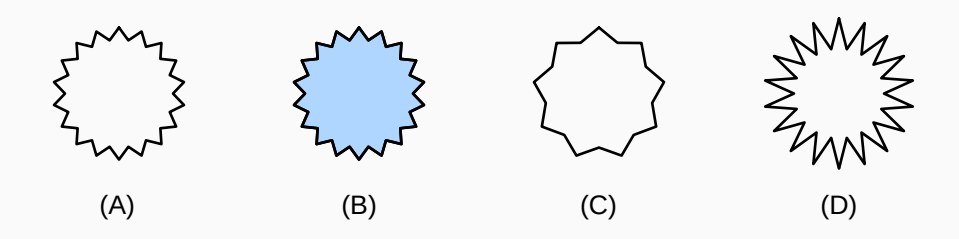


Fig. 156: **OEDrawSimpsonCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawSimpsonCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawSimpsonCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawSimpsonCircle*.

4.5.29 OEDrawSimpsonSurfaceArc

```
bool OEDrawSimpsonSurfaceArc(OEDepict::OEImageBase& image,
                             const OEDepict::OE2DPoint& center,
                             double bgnAngle, double endAngle,
                             double radius,
                             const OEDepict::OEPen& pen,
                             double edgeAngle = 10.0,
                             double patternAngle = 15.0,
                             double minPatternWidthRatio = 0.20,
                             double maxPatternWidthRatio = 0.20)
```

Draws an arc with the *OESurfaceArcStyle.Simpson* style.

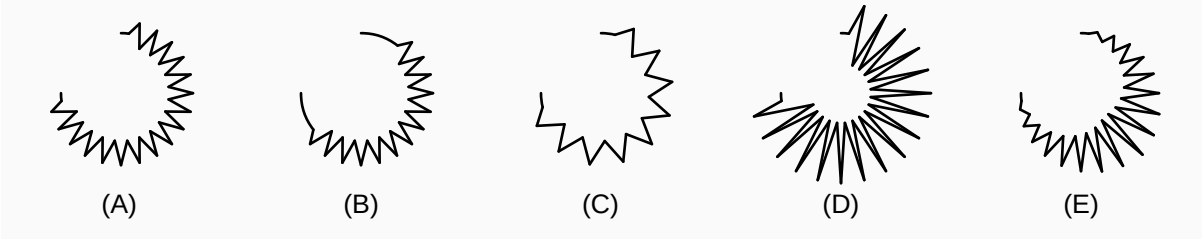


Fig. 157: **OEDrawSimpsonSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawSimpsonSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawSimpsonSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawSimpsonSurfaceArc*.

See also

- *OESimpsonArcFxn* class

4.5.30 OEDrawStitchCircle

```
bool OEDrawStitchCircle(OEDepict::OEImageBase& image,
                        const OEDepict::OE2DPoint& center,
                        double radius, const OEDepict::OEPen& pen,
                        double patternAngle = 10.0,
                        double patternWidthRatio = 0.20)
```

Draws a circle with the *OECircleStyle.Stitch* style.

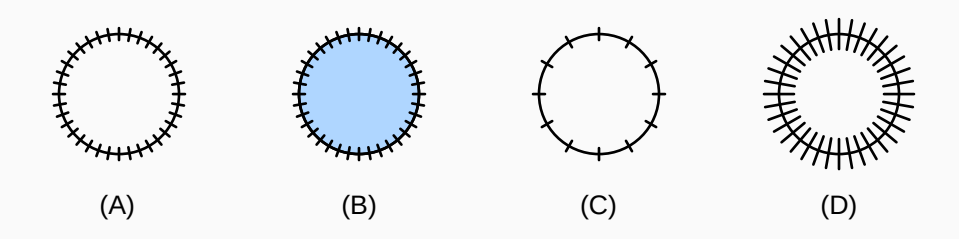


Fig. 158: **OEDrawStitchCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawStitchCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawStitchCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawStitchCircle*.

4.5.31 OEDrawStitchSurfaceArc

```
bool OEDrawStitchSurfaceArc(OEDepict::OEImageBase& image,
                            const OEDepict::OE2DPoint& center,
                            double bgnAngle, double endAngle,
                            double radius, const OEDepict::OEPen& pen,
                            double edgeAngle = 10.0,
```

(continues on next page)

(continued from previous page)

```
                                double patternAngle = 10.0,
                                double minPatternWidthRatio = 0.20,
                                double maxPatternWidthRatio = 0.20)
```

Draws an arc with the *OESurfaceArcStyle.Stitch* style.

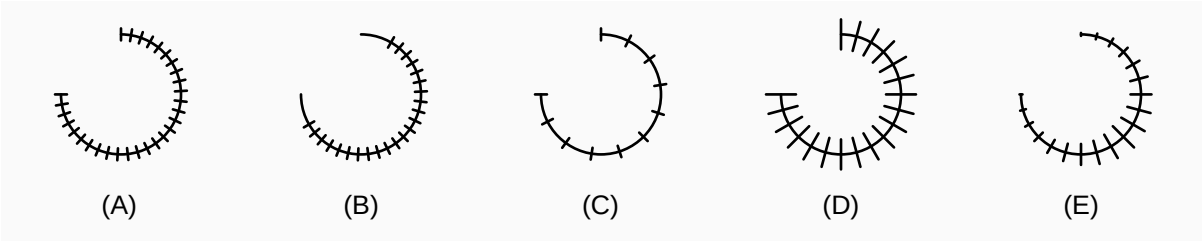


Fig. 159: **OEDrawStitchSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawStitchSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawStitchSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawStitchSurfaceArc*.

See also

- *OEStitchArcFxn* class

4.5.32 OEDrawSunCircle

```
bool OEDrawSunCircle(OEDepict::OEImageBase& image,
                     const OEDepict::OE2DPoint& center, double radius,
                     const OEDepict::OEPen& pen,
                     unsigned int patternDirection = OEPatternDirection::Outside,
                     double patternAngle = 15.0,
                     double patternWidthRatio = 0.15)
```

Draws a circle with the *OECircleStyle.Sun* style.

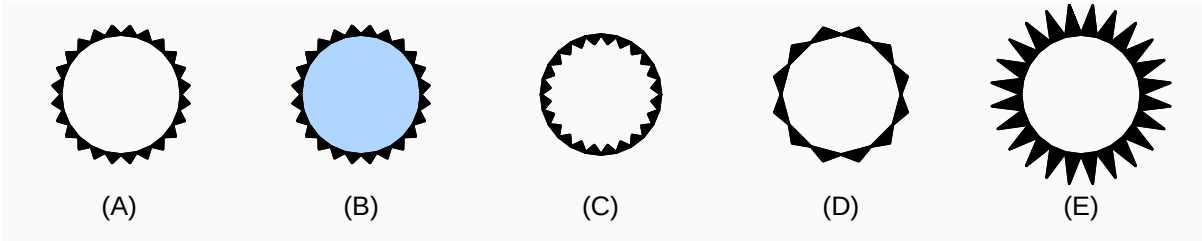


Fig. 160: **OEDrawSunCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawSunCircle*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the circle. This value has to be from the *OEPatternDirection* namespace.

See example (C) in *Figure: OEDrawSunCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawSunCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (E) in *Figure: OEDrawSunCircle*.

4.5.33 OEDrawSunSurfaceArc

```
bool OEDrawSunSurfaceArc(OEDepict::OEImageBase& image,
                         const OEDepict::OE2DPoint& center,
                         double bgnAngle, double endAngle,
                         double radius, const OEDepict::OEPen& pen,
```

(continues on next page)

(continued from previous page)

```
                        double edgeAngle = 10.0,
                        unsigned int patternDirection = OEPatternDirection::Outside,
                        double patternAngle = 15.0,
                        double minPatternWidthRatio = 0.15,
                        double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.Sun* style.

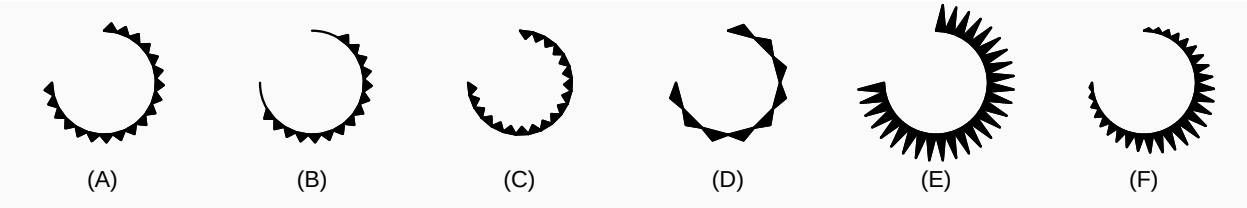


Fig. 161: **OEDrawSunSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawSunSurfaceArc*.

patternDirection
Specifies whether the pattern is drawn inside or outside of the arc. This value has to be from the *OEPatternDirection* namespace.

See example (C) in *Figure: OEDrawSunSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (D) in *Figure: OEDrawSunSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (E) and (F) in *Figure: OEDrawSunSurfaceArc*.

See also

- *OESunArcFxn* class

4.5.34 OEDrawWreathCircle

```
bool OEDrawWreathCircle(OEDepict::OEImageBase& image,
                        const OEDepict::OE2DPoint& center,
                        double radius, const OEDepict::OEPen& pen,
                        double patternAngle = 15.0,
                        double patternWidthRatio = 0.20)
```

Draws a circle with the *OECircleStyle.Wreath* style.

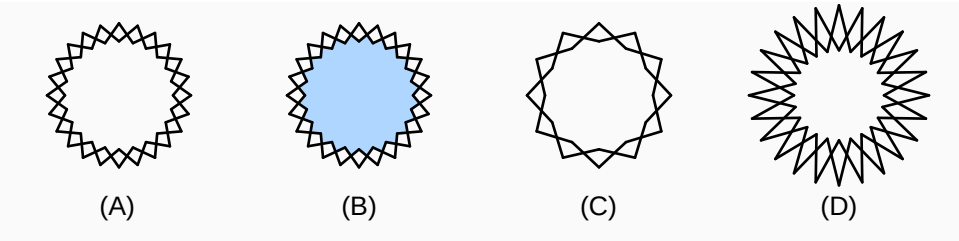


Fig. 162: **OEDrawWreathCircle**

image
The image on which the circle is drawn.

center
The center of the circle.

radius
The radius of the circle.

pen
The graphical properties of the circle.

See examples (A) and (B) in *Figure: OEDrawWreathCircle*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the circle. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawWreathCircle*.

patternWidthRatio
Specifies the width of the *pattern* of the circle relative to its radius. The ratio has to be in a range of $[0.01 - 1.00]$.

See example (D) in *Figure: OEDrawWreathCircle*.

4.5.35 OEDrawWreathSurfaceArc

```
bool OEDrawWreathSurfaceArc(OEDepict::OEImageBase& image,
                            const OEDepict::OE2DPoint& center,
                            double bgnAngle, double endAngle,
                            double radius, const OEDepict::OEPen& pen,
                            double edgeAngle = 10.0,
```

(continues on next page)

(continued from previous page)

```
                                double patternAngle = 20.0,
                                double minPatternWidthRatio = 0.15,
                                double maxPatternWidthRatio = 0.15)
```

Draws an arc with the *OESurfaceArcStyle.Wreath* style.

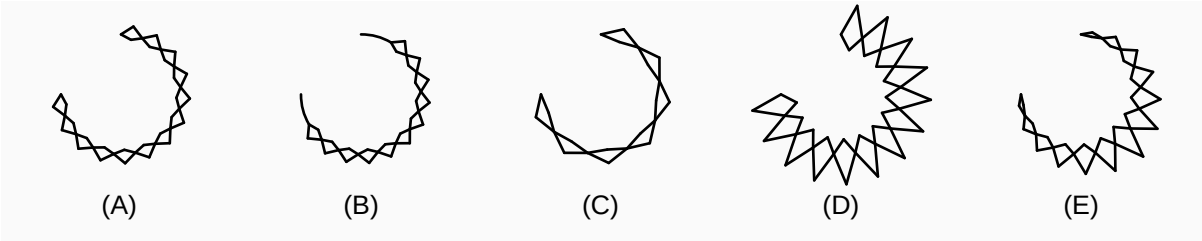


Fig. 163: **OEDrawWreathSurfaceArc**

image
The image on which the arc is drawn.

center
The center of the arc.

bgnAngle, endAngle
The two endpoints of the arc. Both angles are in degrees and their values have to be in a range of $[0.0° - 360.0°]$. Angles are interpreted such that $0.0°$ and $360.0°$ degrees are at the *12 o'clock* position, $90.0°$ degrees corresponds to *3 o'clock*, etc.

radius
The radius of the arc.

pen
The graphical properties of the arc.

edgeAngle
The angle (in degrees) between the end points of the arc and the points where the *pattern* of the arc starts.

See example (B) in *Figure: OEDrawWreathSurfaceArc*.

patternAngle
Specifies the angle (in degrees) between the *patterns* of the arc. The angle has to be in a range of $[1.0° - 90.0°]$

See example (C) in *Figure: OEDrawWreathSurfaceArc*.

minPatternWidthRatio, maxPatternWidthRatio
Specifies the width of the *pattern* of the arc relative to its radius. The *min* ratio is applied to the end of the arc while the *max* ratio specifies the width of the patterns at the middle of the arc. Both ratios have to be in a range of $[0.01 - 1.00]$.

See examples (D) and (E) in *Figure: OEDrawWreathSurfaceArc*.

See also

- *OEWreathArcFxn* class

CHAPTER

FIVE

PRELIMINARY API

5.1 Preliminary OEGrapheme Classes

5.1.1 OEHighlightMonomerDisplayOptions

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEHighlightMonomerDisplayOptions : public OEDepict::OE2DMolDisplayOptions
```

The *OEHighlightMonomerDisplayOptions* class is used to customize molecule depiction when using *OEHighlightMonomers* function.

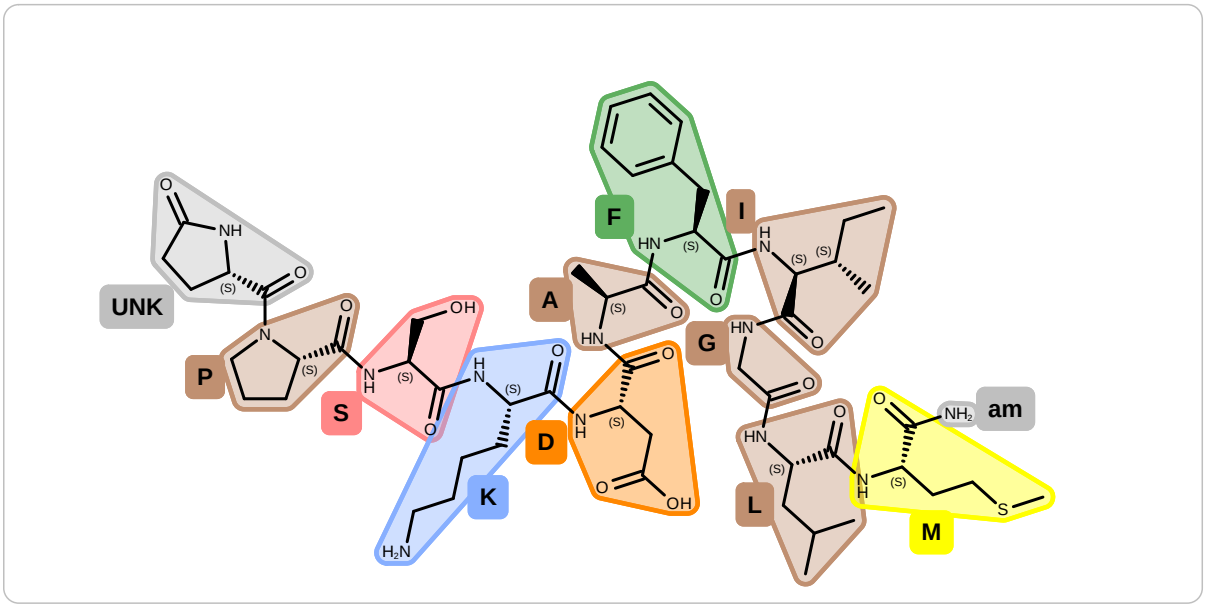


Fig. 1: **Example of using OEHighlightMonomers function**

The following methods are publicly inherited from OE2DMolDisplayOptions:

operator=	GetHydrogenStyle	SetBondLineAtomLabelGapScale
GetAromaticStyle	GetMargin	SetBondLineGapScale
GetAtomColor	GetScale	SetBondPropLabelFont
GetAtomColorStyle	GetSuperAtomLabelFont	SetBondPropLabelFontScale
GetAtomLabelFont	GetSuperAtomStyle	SetBondPropertyFunctor
GetAtomLabelFontScale	GetTitleFont	SetBondStereoStyle
GetAtomPropLabelFont	GetTitleFontScale	SetBondWidthScaling
GetAtomPropLabelFontScale	GetTitleHeight	SetDefaultBondPen
GetAtomPropertyFunctor	GetTitleLocation	SetDimensions
GetAtomStereoStyle	GetWidth	SetHeight
GetBackgroundColor	SetAromaticStyle	SetHydrogenStyle
GetBondColorStyle	SetAtomColor	SetMargin
GetBondLineAtomLabelGapScale	SetAtomColorStyle	SetMargins
GetBondLineGapScale	SetAtomLabelFont	SetScale
GetBondPropLabelFont	SetAtomLabelFontScale	SetSuperAtomLabelFont
GetBondPropLabelFontScale	SetAtomPropLabelFont	SetSuperAtomStyle
GetBondPropertyFunctor	SetAtomPropLabelFontScale	SetTitleFont
GetBondStereoStyle	SetAtomPropertyFunctor	SetTitleFontScale
GetBondWidthScaling	SetAtomStereoStyle	SetTitleHeight
GetDefaultBondPen	SetBackgroundColor	SetTitleLocation
GetHeight	SetBondColorStyle	SetWidth

Constructors

```
OEHighlightMonomerDisplayOptions()
```

Default constructor.

```
OEHighlightMonomerDisplayOptions(double width, double height,
                                 double scale)
```

Constructor that sets the image dimensions and scale.

```
OEHighlightMonomerDisplayOptions(const OEHighlightMonomerDisplayOptions& rhs)
```

Copy constructor.

GetMonomerHighlightColorFunctor

```
const OEMonomerColorBase& GetMonomerHighlightColorFunctor() const
```

Returns the functor that is used to determine the color used to highlight the monomers. By default, the *OEMonomerColorDefault* functor is used.

GetMonomerLabelColorFunctor

```
const OEMonomerColorBase& GetMonomerLabelColorFunctor() const
```

Returns the functor that is used to determine the color of the font use to draw the monomer code.

By default, the *OEMonomerColorMonochrome* functor is used to set all monomer code fonts to black.

SetMonomerHighlightColorFunctor

```
void SetMonomerHighlightColorFunctor(const OEMonomerColorBase& func)
```

Sets the functor that is used to determine the color used to highlight the monomers.

SetMonomerLabelColorFunctor

```
void SetMonomerLabelColorFunctor(const OEMonomerColorBase& func)
```

Sets the functor that is used to determine the color of the font used to draw the monomer code.

GetHighlightUnspecifiedStereo

```
bool GetHighlightUnspecifiedStereo() const
```

Returns whether monomers with unspecified stereo are highlighted.

SetHighlightUnspecifiedStereo

```
void SetHighlightUnspecifiedStereo(bool highlight)
```

Sets whether monomers with unspecified stereo are highlighted.

5.1.2 OELigandAlignerBase

Attention

This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
OESystem::OEUnaryFunction<OEChem::OEMolBase, bool, false, true>  OELigandAlignerBase;
```

OELigandAlignerBase is an abstract class. Functors deriving from this class can be used to perform custom ligand alignment when generating images of ligand-protein interactions.

The following classes derive from this abstract class:

- *OENoOpLigandAligner* class

See also

- *OE2DActiveSiteDisplayOptions.SetLigandAlignerFunctor* method

operator()

```
bool operator()(OEChem::OEMolBase&) const = 0
```

This is a virtual const method that must be implemented in the concrete derived classes.

CreateCopy

```
OELigandAlignerBase* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

5.1.3 OEMonomerColorBase

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
OESystem::OEUnaryFunction<OEChem::OEMonomerData, OESystem::OEColor, true, true> ␣
↪OEMonomerColorBase
```

OEMonomerColorBase is an abstract class that defines the API used to assign colors to monomers.

The following classes derive from this abstract class:

- *OEMonomerColorDefault* class
- *OEMonomerColorMonochrome* class

operator()

```
OESystem::OEColor operator()(const OEChem::OEMonomerData&) const
```

This is a virtual const method that must be implemented in the concrete derived classes.

CreateCopy

```
base_type* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

5.1.4 OEMonomerColorDefault

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMonomerColorDefault : public OEMonomerColorBase
```

The *OEMonomerColorDefault* class is used to assign colors to monomers for depiction.

The following methods are publicly inherited from *OEMonomerColorBase*:

CreateCopy	*operator()*

Constructors

```
OEMonomerColorDefault()
```

Default constructor.

operator()

```
OESystem::OEColor operator()(const OEChem::OEMonomerData &) const
```

Returns the color assigned to a specific monomer (`OEMonomerData`). Standard amino acids are colored by their properties (containing sulphur, aliphatic, aromatic, acidic, basic, hydroxylic, amidic). `OEBlack` is returned for all other monomers.

CreateCopy

```
base_type* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEMonomerColorDefault* object is dynamically allocated and owned by the caller.

5.1.5 OEMonomerColorMonochrome

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMonomerColorMonochrome : public OEMonomerColorBase
```

The *OEMonomerColorMonochrome* class is used to assign the same color to every monomer.

The following methods are publicly inherited from *OEMonomerColorBase*:

CreateCopy	*operator()*

Constructors

```
OEMonomerColorMonochrome()
```

Default constructor that sets the color to `OEBlack`.

```
OEMonomerColorMonochrome(const OESystem::OEColor& color)
```

Constructor with the given color (`OEColor`).

operator()

```
OESystem::OEColor operator()(const OEChem::OEMonomerData &) const
```

Returns the color that the functor was initialized with for each monomer input.

CreateCopy

```
base_type* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEMonomerColorMonochrome* object is dynamically allocated and owned by the caller.

5.1.6 OEMonomerGlyphStyleBase

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
OESystem::OEUnaryFunction<OEChem::OEMonomerData, unsigned, true, true> ␣
↪OEMonomerGlyphStyleBase;
```

OEMonomerGlyphStyleBase is an abstract class that assigns values from the *OECircleStyle* namespace to monomers for depiction.

The following classes derive from this abstract class:

- *OEMonomerGlyphStyleDefault* class

operator()

```
unsigned operator()(const OEChem::OEMonomerData&) const
```

This is a virtual const method that must be implemented in the concrete derived classes.

CreateCopy

```
base_type* CreateCopy() const = 0
```

Virtual const constructor which allows copying of concrete derived objects using a reference to this base class.

5.1.7 OEMonomerGlyphStyleDefault

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMonomerGlyphStyleDefault: public OEMonomerGlyphStyleBase
```

The *OEMonomerGlyphStyleDefault* class is used to assign values from the *OECircleStyle* namespace to each monomers for depiction.

The following methods are publicly inherited from *OEMonomerGlyphStyleBase*:

CreateCopy	*operator()*

Constructors

```
OEMonomerGlyphStyleDefault()
```

Default constructor.

operator()

```
unsigned operator()(const OEChem::OEMonomerData &) const
```

Returns *OECircleStyle.Default* if the monomer is a standard amino acid, and *OECircleStyle.Simpson* otherwise.

CreateCopy

```
base_type* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OEMonomerGlyphStyleDefault* object is dynamically allocated and owned by the caller.

5.1.8 OEMonomerGraphDisplayOptions

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OEMonomerGraphDisplayOptions
```

The *OEMonomerGraphDisplayOptions* class is used to customize molecule depiction when using *OEDrawMonomerGraph* function.

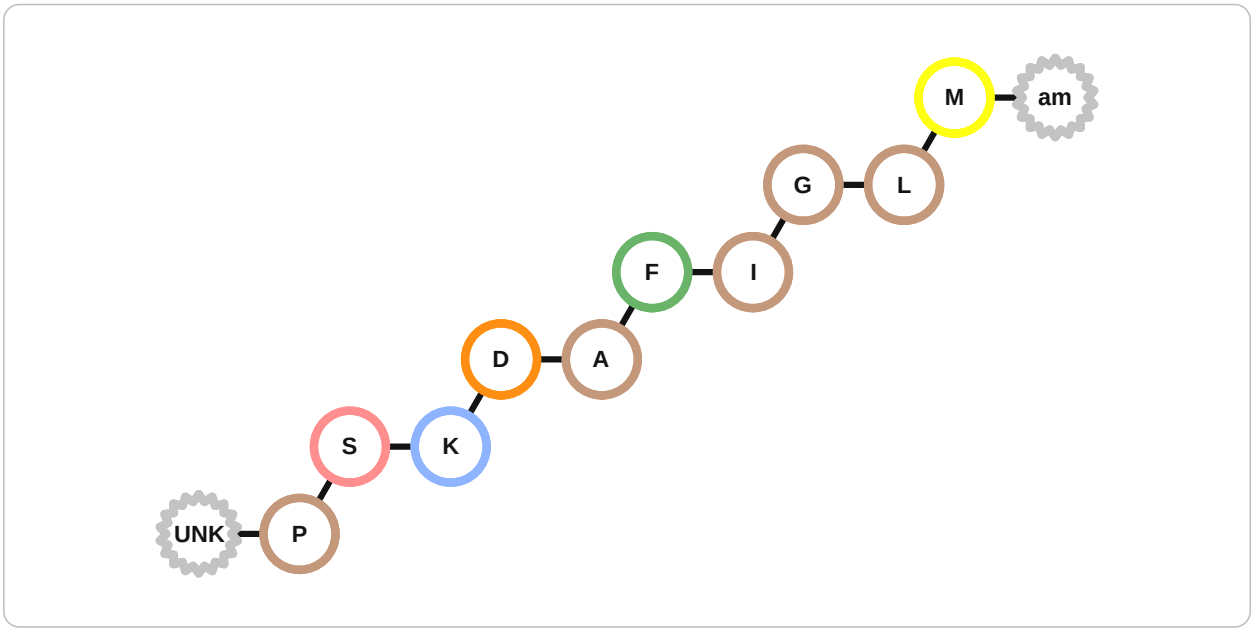


Fig. 2: **Example of using the OEDrawMonomerGraph function**

Constructors

```
OEMonomerGraphDisplayOptions()
```

Default constructor.

```
OEMonomerGraphDisplayOptions(const OEMonomerGraphDisplayOptions& rhs)
```

Copy constructor.

GetInteractiveEffect

```
unsigned GetInteractiveEffect() const
```

Returns the interactive effect of the generated svg image.

GetMonomerColorFunctor

```
const OEMonomerColorBase& GetMonomerColorFunctor() const
```

Returns the functor that is used to determine the color used to draw the glyph representing the monomer. By default, the *OEMonomerColorDefault* functor is used.

GetMonomerScale

```
double GetMonomerScale() const
```

Returns the scaling factor of monomer components in interactive mode.

GetMonomerGlyphStyleFunctor

```
const OEMonomerGlyphStyleBase& GetMonomerGlyphStyleFunctor() const
```

Returns the functor that is used to determine the circle style used to draw the glyph representing the monomer. By default, the *OEMonomerGlyphStyleDefault* functor is used.

GetMonomerLabelColorFunctor

```
const OEMonomerColorBase& GetMonomerLabelColorFunctor() const
```

Returns the functor that is used to determine the color of the font used to depict the monomer code. By default, the *OEMonomerColorMonochrome* functor is used to set all monomer code fonts to black.

SetInteractiveEffect

```
void SetInteractiveEffect(const unsigned effect)
```

Sets the interactive effect of the generated svg image. In interactive mode, the structures of the monomers that make up the molecule are shown on mouse-over or mouse-click.

See also

- OEInteractiveEffect namespace for available interactive effects.

SetMonomerColorFunctor

```
void SetMonomerColorFunctor(const OEMonomerColorBase& func)
```

Sets the functor that is used to determine the color used to draw the glyph representing the monomer.

SetMonomerScale

```
void SetMonomerScale(const double scale)
```

Sets the scaling factor of monomer components in interactive mode.

scale
: This value has to be in a range of [0.25, 0.75]. The default value is 0.50.

> **Note**
>
> This option has to be used along with the *OEMonomerGraphDisplayOptions.SetInteractiveEffect* method

SetMonomerGlyphStyleFunctor

```
void SetMonomerGlyphStyleFunctor(const OEMonomerGlyphStyleBase& func)
```

Sets the functor that is used to determine the circle style used to draw the glyph representing the monomer.

SetMonomerLabelColorFunctor

```
void SetMonomerLabelColorFunctor(const OEMonomerColorBase& func)
```

Sets the functor that is used to determine the color of the font used to depict the monomer code.

5.1.9 OENoOpLigandAligner

> **Attention**
>
> This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
class OENoOpLigandAligner : public OELigandAlignerBase
```

The *OENoOpLigandAligner* class is the default no-op ligand aligner. This means that the 2D coordinates of the ligand are only determined by the 3D coordinates of the bound ligand.

The following methods are publicly inherited from *OELigandAlignerBase*:

CreateCopy	*operator()*

Constructors

```
OENoOpLigandAligner()
```

Default constructor.

operator()

```
bool operator()(OEChem::OEMolBase& ligand) const
```

No-op implementation always returns true.

CreateCopy

```
base_type* CreateCopy() const
```

Deep copy constructor that returns a copy of the object. The memory for the returned *OENoOpLigandAligner* object is dynamically allocated and owned by the caller.

5.2 Preliminary OEGrapheme Functions

5.2.1 OEDrawIridiumData

> **Attention**
>
> This API is currently available in C++ and Python.

```
bool OEDrawIridiumData(OEDepict::OEImageBase& image,
                       const OEBio::OEDesignUnit& du)
bool OEDrawIridiumData(OEDepict::OEImageBase& image,
                       const OEBio::OEIridiumData& iridium)
```

Renders the Iridium data into an easily interpretable graph.

image
: The image on which the Iridium data graph is drawn.

du
: The OEDesignUnit object of which Iridium data is being rendered.

iridium
: The OEIridiumData object that is being rendered.

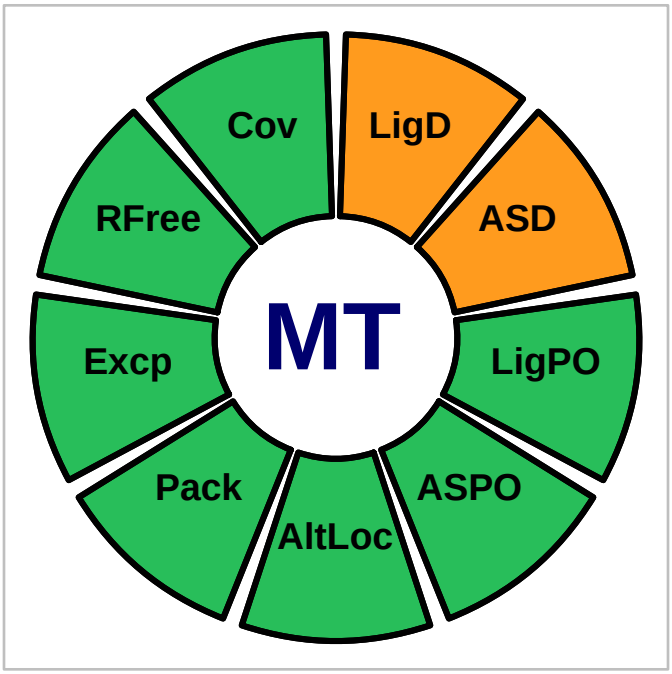


Fig. 3: **Example of using the OEDrawIridiumData function**

Note

When generating `.svg` image, the output will be interactive. With any other image file format it will be static. See also Generating Interactive SVG Images chapter in **OEDepict TK** manual.

See also

- Iridium chapter in **OESpruce TK** manual
- OEDesignUnit class
- OEIridiumData class

5.2.2 OEDrawMonomer

Attention

This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
bool OEDrawMonomer(OEDepict::OEImageBase& image,
                   const OEChem::OEMonomer& monomer)
```

Draws a monomer highlighting its attachment points.

image
The image on which the monomer is drawn (`OEImage`).

monomer
The monomer being draw (`OEMonomer`).

Example:

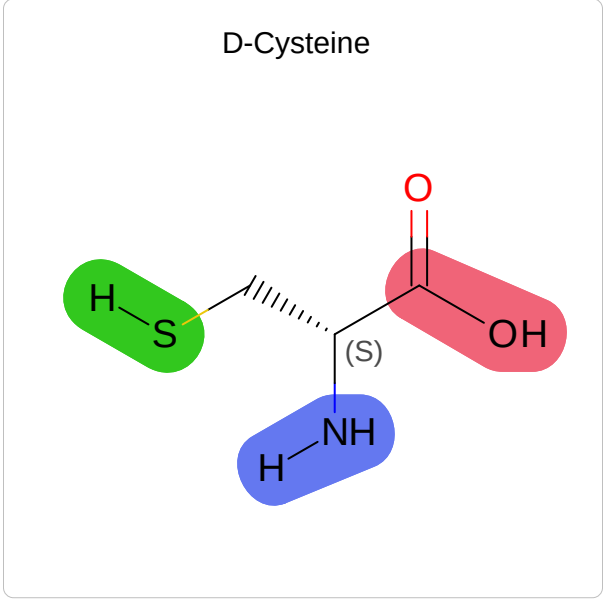


Fig. 4: **Example of using OEDrawMonomer function**

Code Example

- Depict Monomer **OpenEye Python Cookbook** recipe

5.2.3 OEDrawMonomerGraph

Attention

This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
bool OEDrawMonomerGraph(OEDepict::OEImageBase& image,
                        const OEChem::OEMolBase& mol)
bool OEDrawMonomerGraph(OEDepict::OEImageBase& image,
                        const OEChem::OEMolBase& mol,
                        const OEMonomerGraphDisplayOptions& opts)
```

The *OEDrawMonomerGraph* function is designed to depict the monomer graph of a molecule initialized from a HELM string.

image
The image on which the molecule is drawn.

mol
The molecule being depicted.

opts
The *OEMonomerGraphDisplayOptions* object that stores properties that determine the styles of the molecule depiction.

Example:

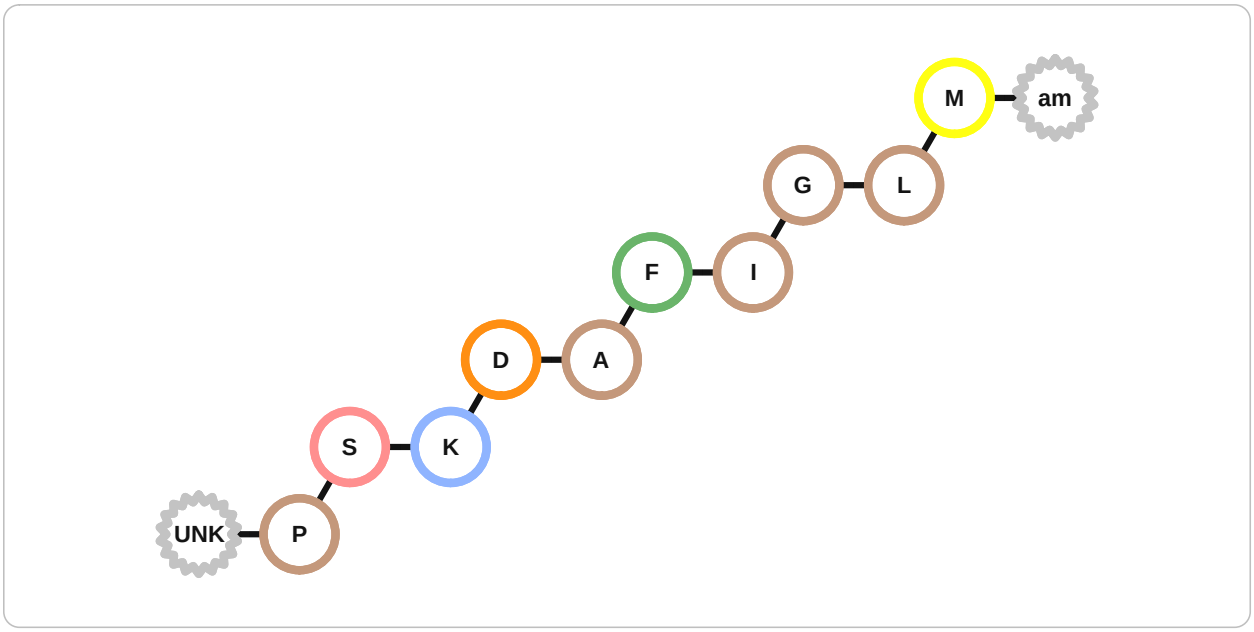


Fig. 5: **Example of using the OEDrawMonomerGraph function**

See also

- `OEHelmToMol` function
- *OEHighlightMonomers* function to depict the molecules initialized from HELM
- *OEDrawPeptide* function to depict a peptide with standard amino acids
- *OEDrawResidues* function to depict residues of peptides

Code Example

- Depict Peptide **OpenEye Python Cookbook** recipe

5.2.4 OEHighlightMonomers

Attention

This is a preliminary API and may be improved based on user feedback. It is currently available in C++ and Python.

```
bool OEHighlightMonomers(OEDepict::OEImageBase& image,
                         const OEChem::OEMolBase& mol)
bool OEHighlightMonomers(OEDepict::OEImageBase& image,
                         const OEChem::OEMolBase& mol,
                         const OEHighlightMonomerDisplayOptions& opts)
```

The *OEHighlightMonomers* function is designed to depict a molecule initialized from a HELM string and highlight its monomer components.

image
The image on which the molecule is drawn.

mol
The molecule being depicted.

opts
The *OEHighlightMonomerDisplayOptions* object that stores properties that determine the styles of the molecule depiction.

Example:

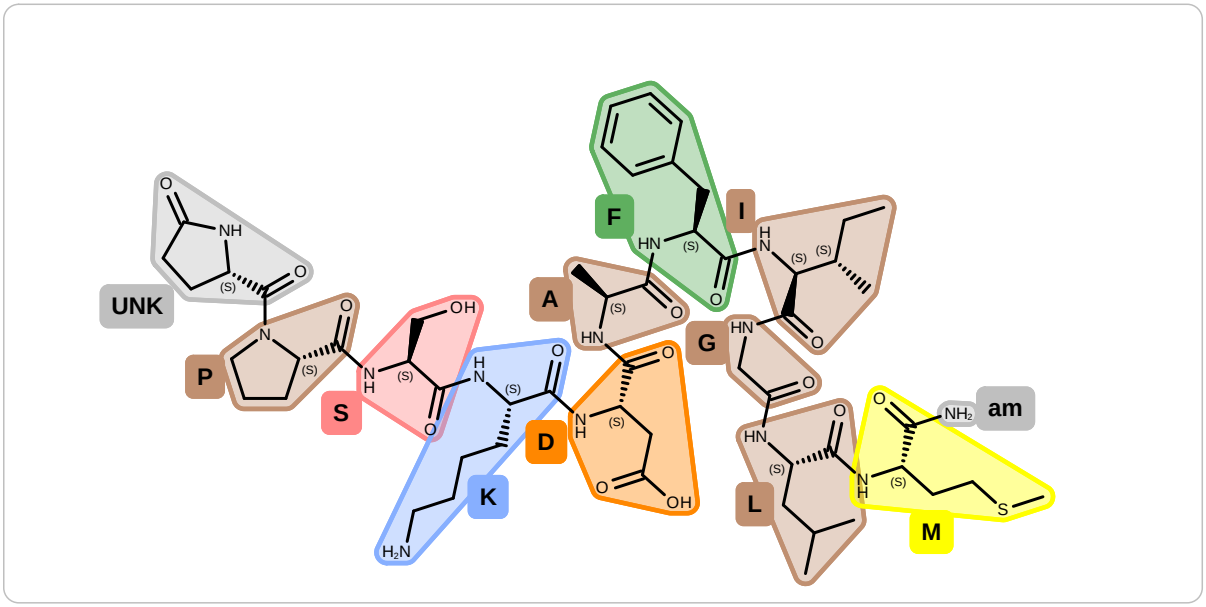


Fig. 6: **Example of using OEHighlightMonomers function**

```
bool OEHighlightMonomers(OEDepict::OE2DMolDisplay& disp,
                         const OEHighlightMonomerDisplayOptions& opts)
```

This overload of the *OEHighlightMonomers* function renders the molecule with the monomer highlighting to the display object.

disp
The display object for the molecule being depicted.

opts
The *OEHighlightMonomerDisplayOptions* object that stores properties that determine the styles of the molecule depiction.

See also

- *OEDrawMonomerGraph* function to depict the monomer graph of molecule initialized from HELM
- *OEDrawPeptide* function to depict a peptide with standard amino acids
- *OEDrawResidues* function to depict residues of a peptide

Code Example

- Depict Peptide **OpenEye Python Cookbook** recipe

CHAPTER

SIX

RELEASE HISTORY

6.1 Grapheme™ TK 1.5.5

6.1.1 New Features

- The following preliminary APIs have been added for monomer and peptide depiction support.

  Classes:

  - *OEPeptideDisplayOptions* — options for peptide depiction.
  - *OEHighlightMonomerDisplayOptions* — options for monomer highlight depiction.
  - *OEMonomerColorDefault* — default monomer color functor.
  - *OEMonomerColorMonochrome* — monochrome monomer color functor.
  - *OEMonomerGlyphStyleBase* — base class for monomer glyph style functor.
  - *OEMonomerGlyphStyleDefault* — default monomer glyph style functor.
  - *OEMonomerGraphDisplayOptions* — options for monomer graph depiction.

  Functions:

  - *OEDrawMonomer* — draws a monomer image.
  - *OEDrawMonomerGraph* — draws a monomer graph of a peptide.
  - *OEHighlightMonomers* — highlights monomers in a peptide depiction.
- The *OERenderActiveSite* function now displays a counter on the top of the T- and pi-stacking interaction lines if multiple interactions exist between the aromatic ring of the ligand and the aromatic ring of a specific residue.
- The following APIs have been added to allow customization of ligand alignment in protein–ligand interaction depictions:
  - *OELigandAlignerBase*
  - *OE2DActiveSiteDisplayOptions.SetLigandAlignerFunctor* method
  - *OE2DActiveSiteDisplayOptions.SetOptimizeLigandOrientation* method

6.1.2 Minor Bug Fixes

- The *OEDrawResidues* function now depicts molecules with up to 300 heavy atoms.

6.2 Grapheme™ TK 1.5.4

Minor internal improvements have been made.

6.3 Grapheme™ TK 1.5.3

Minor internal improvements have been made.

6.4 Grapheme™ TK 1.5.2

6.4.1 New features

- An option has been added to the *OEColorForceFieldDisplay* to preserve input query color atoms instead of applying them from the defined color force field. There are two new methods and one new constructor, taking that argument along with the color force field.
  - *SetPreserveQueryColors*
  - *GetPreserveQueryColors*
  - *Constructors*

6.4.2 Minor bug fixes

- An issue has been fixed which incorrectly used the ligand density value for the active site density value when depicting the Iridium classification.

6.5 Grapheme™ TK 1.5.1.3

- Following new methods have been added to *OEColorForceFieldDisplay* to enable preserving user defined query colors.
  - *SetPreserveQueryColors*
  - *GetPreserveQueryColors*
  - *Constructors*

6.6 Grapheme™ TK 1.5.1.0

Minor internal improvements have been made.

6.7 Grapheme™ TK 1.5.0

Minor internal improvements have been made.

6.8 Grapheme™ TK 1.4.7

Minor internal improvements have been made.

6.9 Grapheme™ TK 1.4.6

Minor internal improvements have been made.

6.10 Grapheme™ TK 1.4.5

- Minor internal improvements have been made.

6.11 Grapheme™ TK 1.4.4

Fall 2021

- Minor internal improvements have been made.

6.12 Grapheme™ TK 1.4.3

July 2021

6.12.1 New features

- Rendering of protein-ligand binding sites has been improved to handle new intramolecule types in the `OEPerceiveInteractionHints` function.

6.13 Grapheme™ TK 1.4.2

Fall 2020

6.13.1 New features

- A new function, *`OEDrawResidues`*, has been added that visualizes residue information for peptides.

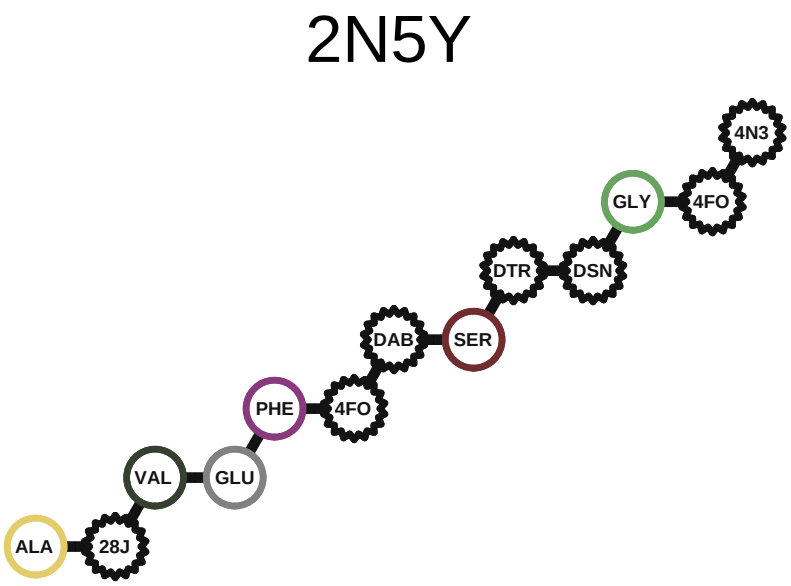


Fig. 1: **Example of using the OEDrawResidues function**

6.13.2 Minor bug fixes

- A bug related to drawing interaction hints spanning multiple residues has been fixed.

6.14 Grapheme ™ TK 1.4.1

- Minor internal improvements have been made.

6.15 Grapheme™ TK 1.4.0

6.15.1 New features

- New *OEDrawIridiumData* functions have been added to visualize OEIridiumData.

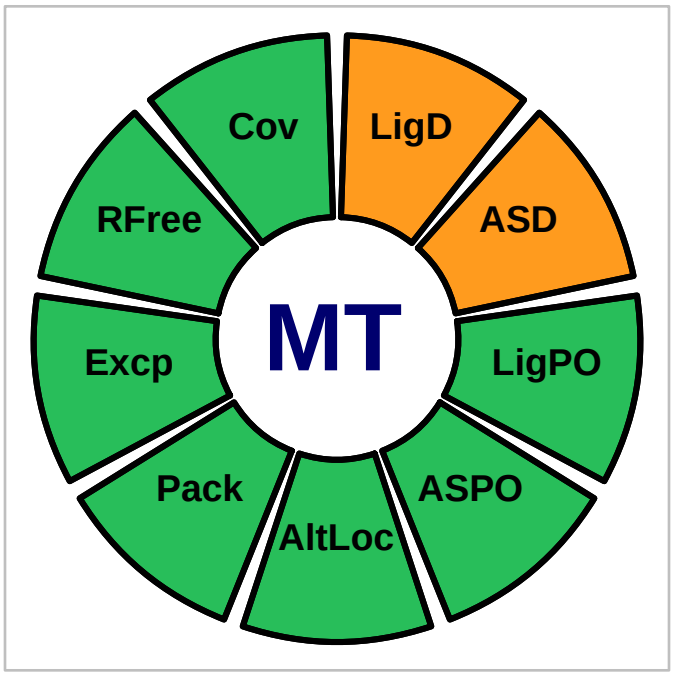


Fig. 2: **Example of visualizing Iridium data**

- A new function, *OERenderActiveSite*, has been added that depicts the new non-ideal hydrogen bonds. For more details about these interactions, see the OEIsNonIdealHBondInteractionHint functor.

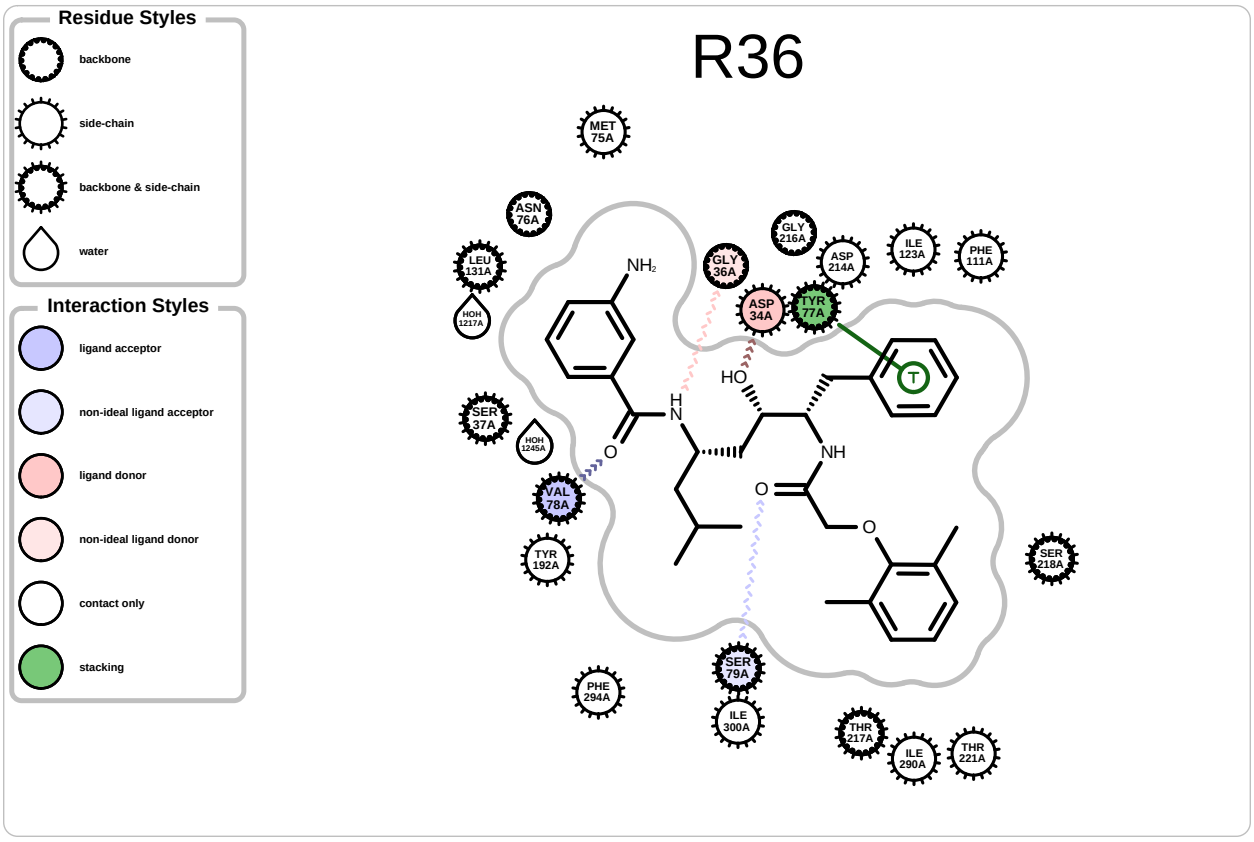


Fig. 3: **Example of visualizing non-ideal hydrogen bond (PDB:1LEE)**

- A new overload of *OEPrepareAlignedDepictionFrom3D* has been added that generates 2D coordinates of a molecule driven by the 3D overlay of the shape query display object.

6.16 Grapheme™ TK 1.3.7

6.16.1 New features

- The following APIs have been added to depict Ramachandran plots:
  - *OERamachandranPlot* class
  - *OEPlotMarker* class and *OEPlotMarkerStyle* namespace
  - *OERenderRamachandranPlot* function

6.16.2 Minor bug fixes

- *OEResidueSVGStandardMarkup.GetGroupId* now replaces spaces with an underscore character to generate valid SVG group IDs.

6.17 Grapheme™ TK 1.3.6

6.17.1 New features

- New *OERenderBFactorMap* and *OEDrawBFactorMapLegend* functions have been added to visualize the B-factor of a ligand and its protein environment.
- A new *OEDrawUnpairedInteractionMapLegend* function has been added to depict the legend of the interactions depicted in an unpaired map. (See also the *OERenderUnpairedInteractionMap* function).
- The *OE2DActiveSiteDisplay.GetDisplayedResidues* function now takes a parameter from the new *OEActiveSiteRenderStyle* namespace.
- Two new methods, *OEColorGradientDisplayOptions.GetColorStopVisibility* and *OEColorGradientDisplayOptions.SetColorStopVisibility*, have been added to customize color gradient depiction.

6.17.2 New features (Preliminary API)

- A new *OERenderActiveSiteMaps* function has been added that generates an interactive image of protein-ligand maps.

  There is no image available for the PDF version of this book.

6.17.3 Major bug fixes

- When rendering ligand-protein maps with **Grapheme TK**, molecule surfaces are generated to depict the overall shape of the pocket. In the case of a large protein, this process can be both computationally expensive and can require a large amount of memory. In order to avoid these issues, only protein atoms that are no more than 20Å away from any ligand atoms are now considered and the resolution of the surface is reduced.
- Rendering active site maps to an image with less than a `200.0` width or height parameter is now disabled to avoid generating very tiny, incomprehensible images.
- When rendering ligand-protein maps (e.g., active site, unpaired), interactions between the ligand and more than 1 residue is now ignored and a warning is thrown.
- Scaling of legend labels for large images has been fixed. All **Grapheme TK** legends of ligand-protein depictions now have a consistent style.

6.17.4 Minor bug fixes

- A problem that occurred when coloring hexagon glyphs representing DNA/RNA residues with more than 1 interaction type has been fixed.
- An issue with depicting an interaction type in a legend when that interaction type is missing from the corresponding active site map has been fixed.
- A small improvement has been made to help to generate residue layouts that are platform-independent.
- A small improvement has been made to generate better residue layout: the space around the ligand for potential residue positions is now better sampled to allow positioning residues close to edge of the canvas.
- The text associated with the color stops when depicting color gradients vertically using the *OEDrawColorGradient* function is now right-adjusted.

6.17.5 Documentation changes

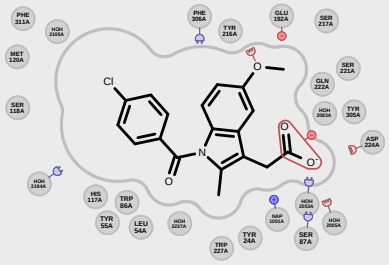
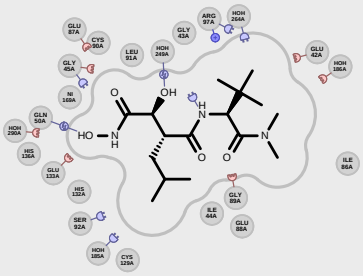
- The *shapeoverlap2pdf.py* example has been fixed.

6.18 Grapheme™ TK 1.3.5

6.18.1 New features

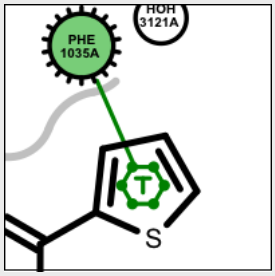
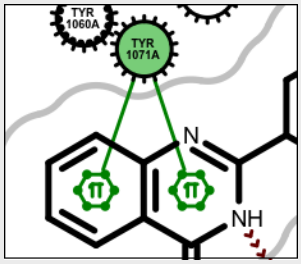
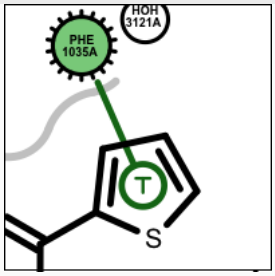
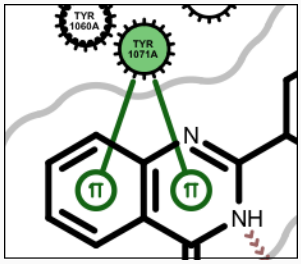
- A new function, `OERenderUnpairedInteractionMap`, has been added to visualize clash and unpaired interactions.

Table 1: **Examples of visualizing clash and unpaired interactions**

PDB: 1S2A	PDB: 3K6L
IMN	2BB
	

- `OERenderActiveSite` now depicts covalent and cation-pi interactions.
- The style for depicting stacking interactions has been simplified.

Table 2: **Examples of stacking interaction depiction**

	T-stacking	Pi-stacking
2017.Oct		
2018.Feb		

- The color scheme used to visualize interaction types when depicting protein-ligand interactions has been updated to accommodate the two new interaction types.

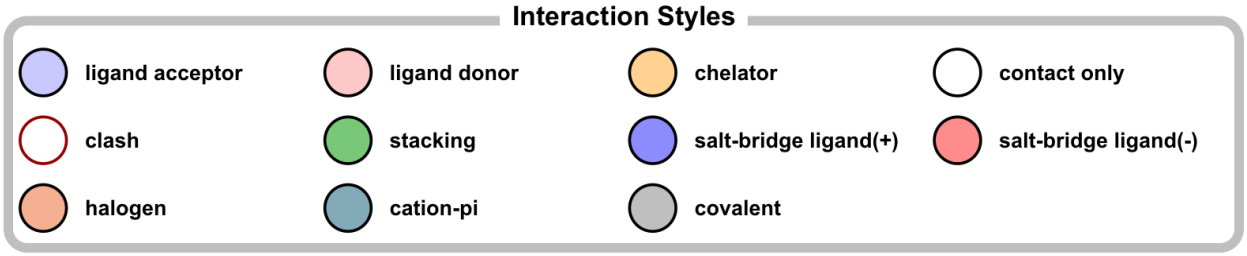


Fig. 4: **New interactions color scheme**

- A new method, *OE2DActiveSiteDisplay.IsValid*, has been added that is checked by all **Grapheme TK** API points.
- The *OEResidueSVGStandardMarkup* class now takes a separator option.
- The following API, introduced in the 2017.Jun release for marking up residue circles in SVG images generated by the *OERenderActiveSite* function, is no longer preliminary and is now also available in both Java and C#:
  - *OEResidueSVGMarkupBase* class
  - *OEResidueSVGNoMarkup* class
  - *OEResidueSVGStandardMarkup* class

6.18.2 Minor bug fixes

- The algorithm used by the *OE2DActiveSiteDisplay* function to scale and position residues around ligands has been improved.
- *OERenderActiveSite* is now able to generate images when some ligand atoms have identical 2D coordinates.
- *OERenderActiveSite* now throws a warning if an interaction cannot be rendered.

6.18.3 Java-specific changes

- The API for marking up residues in SVG images is now fully supported in Java (see the list above).

6.18.4 C#-specific changes

- The API for marking up residues in SVG images is now fully supported in C# (see the list above).

6.19 Grapheme™ TK 1.3.4

6.19.1 New features

- The following new API has been added to customize peptide depiction:
  - *OEPeptideDisplayOptions* class
  - *OEPeptideLabelStyle* namespace
  - *OEDrawPeptide* function that takes an option class

  The main feature is that *OEDrawPeptide* now can generate interactive SVG images where the amino acid components of the compact peptide representation are depicted on mouse-over.
- The following new API has been added to customize active site depiction:
  - *OE2DActiveSiteDisplayOptions.GetSVGMagnifyResidueInHover*

  - *OE2DActiveSiteDisplayOptions.SetSVGMagnifyResidueInHover*
  - *OE2DActiveSiteDisplayOptions.GetRenderInteractiveLegend*
  - *OE2DActiveSiteDisplayOptions.SetRenderInteractiveLegend*

  These new options allow users to generate interactive SVG images when calling the *OERenderActiveSite* function.

- The *OEBondGlyphZigZag* constructor now takes a parameter that allows drawing diagonal zig-zag lines across a bond.



Fig. 5: **Example of using the OEBondGlyphZigZag class**

- **OEGrapheme TK** now uses the new thread-safe **OEShape TK** API to generate shape and color overlaps for *OEShapeQueryDisplay* and *OEShapeOverlapDisplay* classes.

6.19.2 Minor bug fixes

- *OEResidueSVGStandardMarkup.GetGroupId* now returns `group id` for a residue that includes the insert code (see also `OEResidue.GetInsertCode`)

6.20 Grapheme™ TK 1.3.3

6.20.1 New features

- A new function, *OERenderContactMap*, has been added that generates interactive SVG images. Clicking on any residue circle will highlight those ligand atoms that interact with the given residue.

6.20.2 Preliminary API

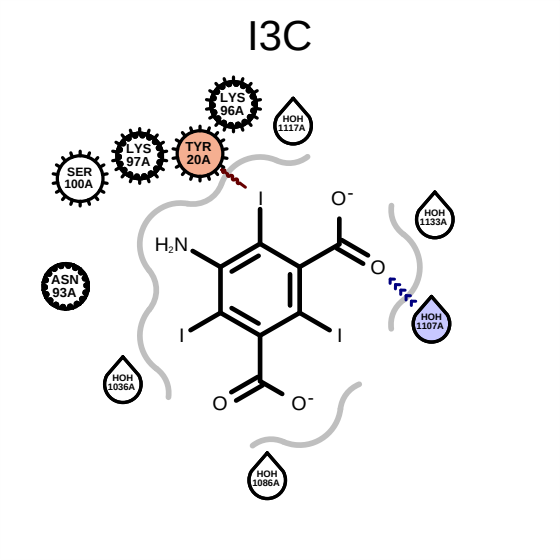
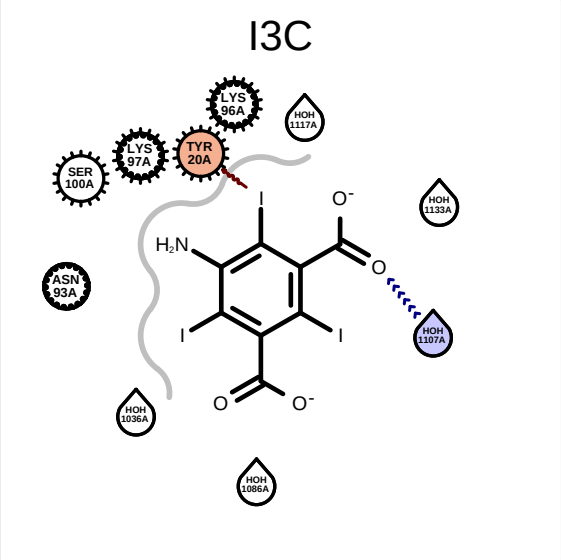
- A new Preliminary API has been added to mark up the residue circles in SVG images generated by the *OERenderActiveSite* function:
  - *OE2DActiveSiteDisplayOptions.SetResidueSVGMarkupFunctor* and *OE2DActiveSiteDisplayOptions.GetResidueSVGMarkupFunctor* methods
  - *OEResidueSVGMarkupBase* base class and *OEResidueSVGNoMarkup* and *OEResidueSVGStandardMarkup* concrete classes

6.20.3 Major bug fixes

- Several problems in the *OERenderActiveSite* function have been fixed:
  - When depicting protein-ligand interactions, grey lines represent the shape of the binding pocket. The absence of lines indicates solvent-accessible regions. In order visualize the solvent-accessible areas more realistically, waters are now ignored when calculating the molecule surfaces that determine the 2D representation (compare images below).

- The rare case where the ligand is completely surrounded by water is now handled correctly.
- The following heuristics are now used to improve the positioning of residue glyphs around the ligand:
  * Waters are more likely to be positioned in solvent-exposed regions.
  * Other residues are only positioned in solvent-exposed regions if it results in a more visually pleasing layout.

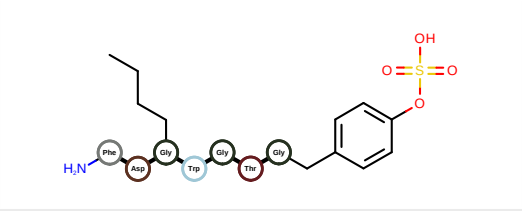
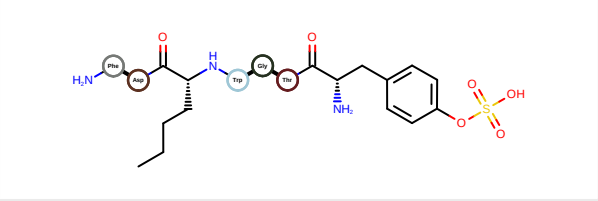
Table 3: **Example of visualizing solvent-accessible regions and water positioning (PDB: 3E3D)**

2017.Feb	2017.Jun
	

6.20.4 Minor bug fixes

- *OEDrawPeptide* has been fixed and now allows only amide and disulfide bonds when identifying standard amino acids. No alpha-carbon substitutions are allowed.

Table 4: **Example of visualizing peptides with OEDrawPeptide**

2017.Feb	2017.Jun
	

6.20.5 Documentation changes

- A new *Python Cookbook Examples* section has been added.

6.21 Grapheme™ TK 1.3.2

6.21.1 New features

- A new function, *OEDrawPeptide*, has been added that identifies standard amino acid components in molecules and substitutes them with circular glyphs and corresponding labels. The figure below illustrates this functionality for oxytocin.

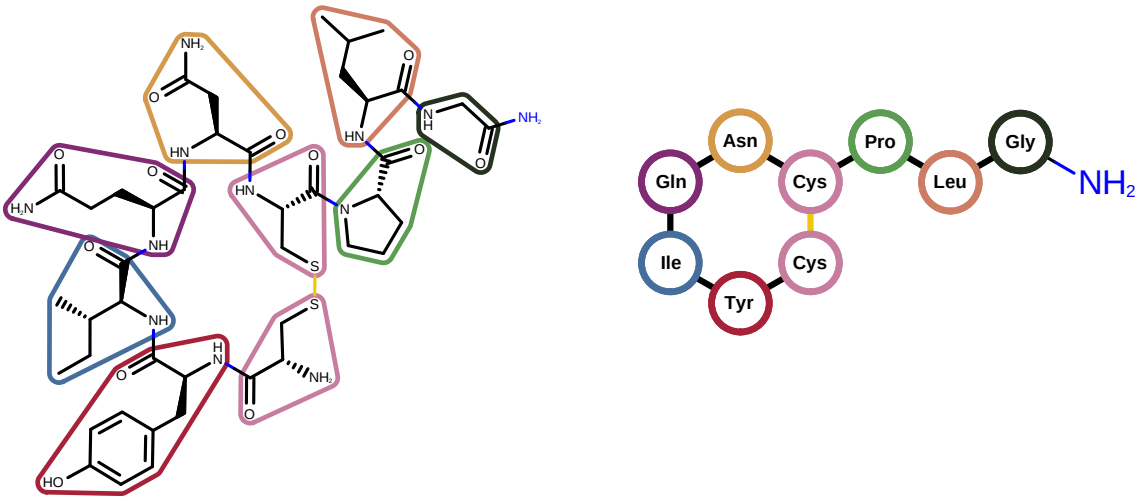


Fig. 6: **Example of visualizing oxytocin**
(left) depiction of oxytocin highlighting its amino acid fragments
(right) depiction of oxytocin using the OEDrawPeptide function

- *OERenderActiveSite* can now visualize halogen bond interactions (represented by a brown wavy arrow).

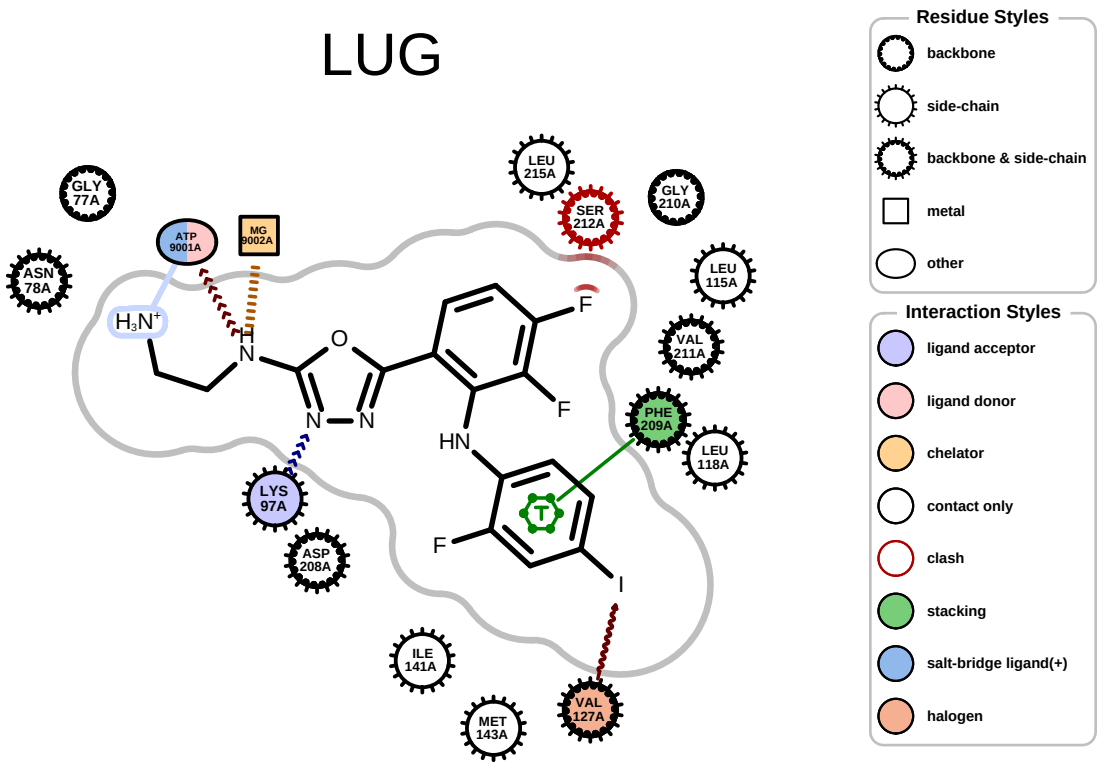


Fig. 7: Example of visualizing halogen bond interaction (`PDB 3EQB`)

- The legend of the protein-ligand depiction has been improved by separating and labeling residue styles and interaction styles (see image above).

6.21.2 Minor bug fixes

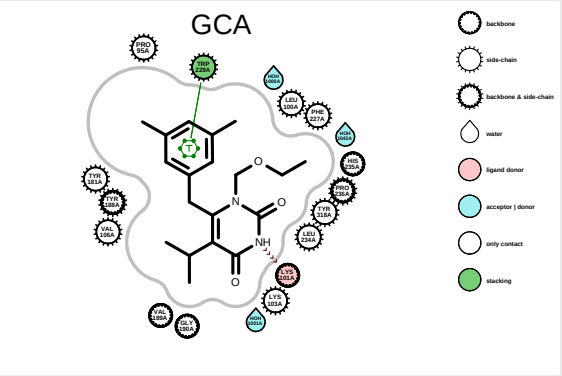
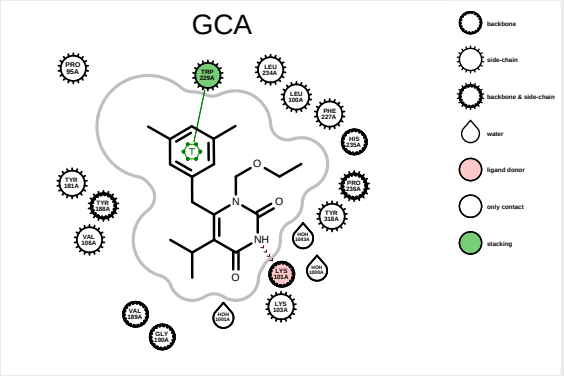
- The label “only contact” has been changed to “contact only” in the legend of the protein-label depiction (see image above).

6.22 Grapheme TK 1.3.1

6.22.1 New features

- The algorithm that determines the position of residues around a ligand has been improved. Residues are no longer placed the same distance from the ligand but are positioned closer to the ligand atoms they interact with and farther away if they have `only contact` interactions with the ligand.

Table 5: **Example of the placement of residues around a ligand**

2016.Jun	2016.Oct
	

- `Clash` interactions perceived by the `OEPerceiveInteractionHints` function are now visualized when rendering protein-ligand interactions.

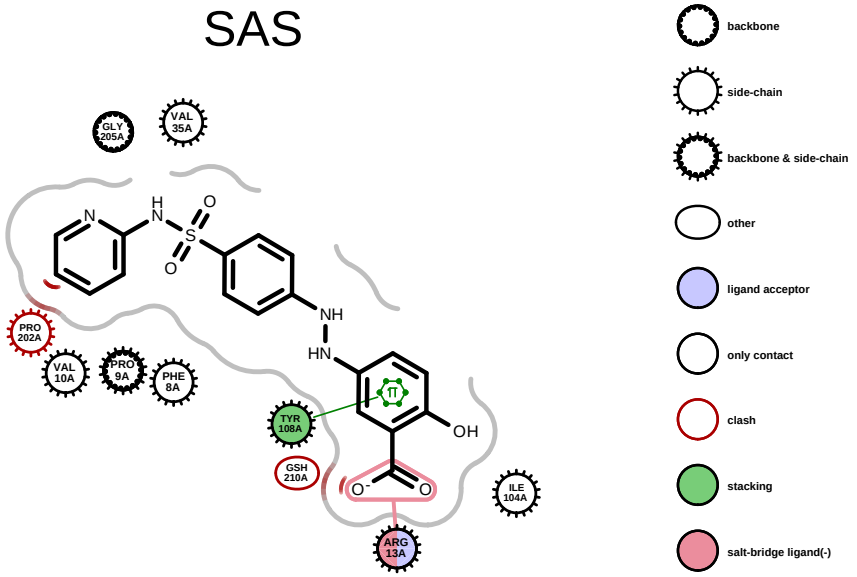
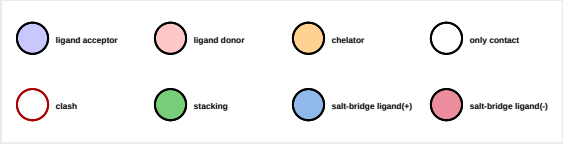


Fig. 8: **Example of visualizing Clash interactions**

- The color scheme used to visualize interaction types when depicting protein-ligand interactions has been updated.
  - The `acceptor | donor` color style no longer exists since `OEPerceiveInteractionHints` now perceives possible hydrogen bonding interactions between ligand and water. If no such interaction exists, the glyph representing a specific water residue will be left uncolored, indicating an `only contact` interaction.
  - `Clash` interactions are now colored dark red.

Table 6: **Color schemes**

2016.Jun color scheme	2016.Oct color scheme
	

6.22.2 Minor bug fixes

- A problem that caused the glyph that represents a residue in a receptor-ligand interaction plot to be incorrectly colored based on its interaction type has been fixed.

6.22.3 Documentation changes

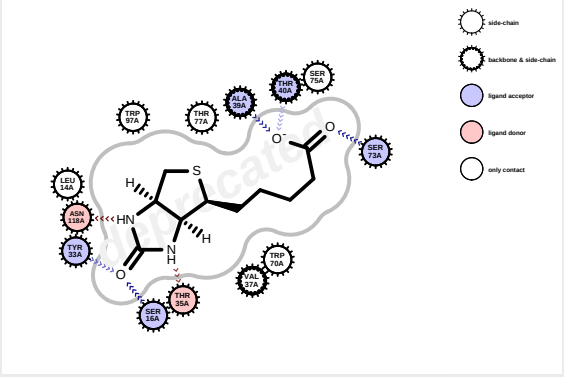
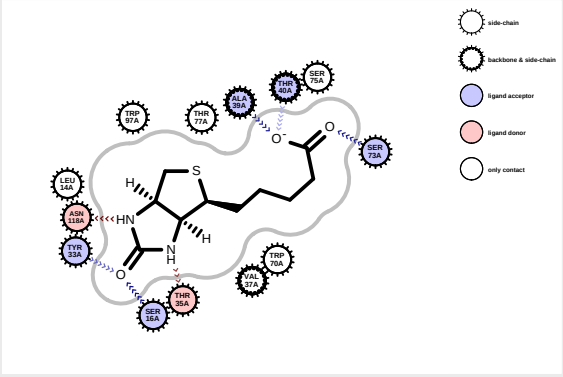
- All images in the **Grapheme TK** documentation have been regenerated to reflect the changes made since the previous release.

6.23 Grapheme TK 1.3.0

6.23.1 New features

- Due to the deprecation of *OEFragmentNetwork* and related classes in **OEBio**, **Grapheme TK** has introduced the following overloads:
  - `OEPrepareActiveSiteDepiction` function has been overloaded. The new version takes an OEInteractionHintContainer object.
  - *OE2DActiveSiteDisplay* class now should be constructed with an OEInteractionHintContainer object.

  All functions and methods using *OEFragmentNetwork* are deprecated in **Grapheme TK** and will throw a warning when called. Active site depictions generated with the deprecated API will have a "deprecated" watermark (see the example below).

using the deprecated OEFragmentNetwork class	using the new OEInteractionHintContainer class
	

In order to avoid the "deprecated" watermark from appearing in the image, use

```
OEInteractionHintContainer asite = new OEInteractionHintContainer(protein, ligand);
```

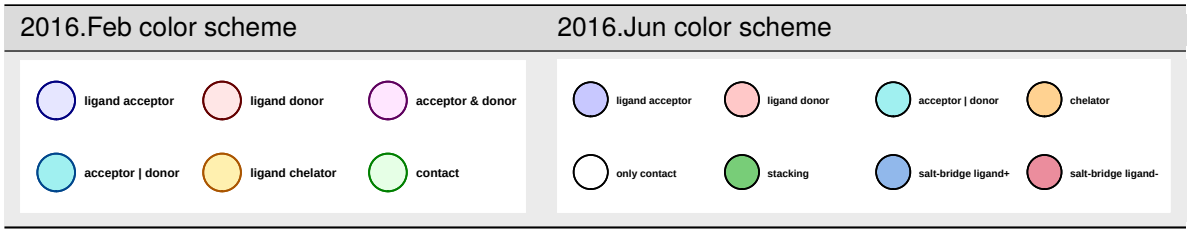
instead of

```
OEFragmentNetwork asite = new OEFragmentNetwork(protein, ligand);
```

See also

For more details about the deprecated classes and functions, see the Deprecated OEFragmentNetwork and related API section.

- *OE2DActiveSiteDisplay* now can visualize salt-bridge and aromatic ring stacking interactions.
- The color scheme used to visualize different interaction types when depicting protein-ligand interaction has been changed to accommodate the new types (Pi- and T-stacking and salt-bridge).



- *OE2DActiveSiteDisplay.GetDisplayedLigand* and *OE2DActiveSiteDisplay.GetDisplayedResidues* methods have been added.
- *OEAddLigandHighlighting* and *OEAddResidueHighlighting* functions have been added to allow highlighting in active site depiction. See the example below.

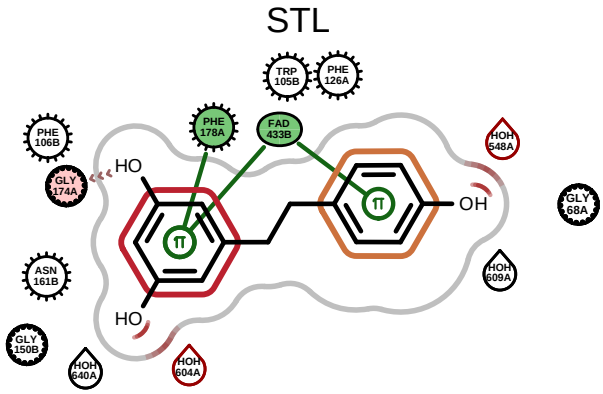


Fig. 9: **Example of ligand highlighting**

- *OEShapeOverlapDisplayStyle.PropertyCloud* style has been added to visualize shape overlays along with the following APIs:
  - *OEShapeOverlapDisplayStyle* namespace

  - *OEShapeOverlapDisplayOptions.SetOverlapDisplayStyle* and *OEShapeOverlapDisplayOptions.SetOverlapDisplayStyle* methods

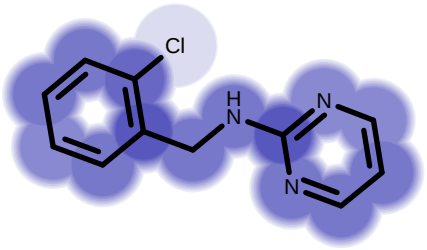


Fig. 10: **Example of visualizing shape overlaps with PropertyCloud style**

Using this new style can result in a 10-times performance increase when generating shape overlay images; the generated images will also be significantly smaller (PNG 50%, SVG 60-times, PDF 9-times).

- The following APIs have been added to identify ligand atoms and bonds, along with residues of an *OE2DActiveSiteDisplay* object that are close to a given 2D display coordinate:
  - *OEGetNearestAtom* and *OEGetNearbyAtom* functions
  - *OEGetNearestBond* and *OEGetNearbyBond* functions
  - *OEGetNearestResidue* and *OEGetNearbyResidue* functions along with the *OENearestResidue* class

6.23.2 Documentation changes

- All images in the **Grapheme TK** documentation were automatically regenerated to reflect the changes made since the previous release.
- The example in *Depicting Active Site Interactions* has been updated to use the new OEInteractionHintContainer class instead of the deprecated *OEFragmentNetwork*.

6.24 Grapheme TK 1.2.3

6.24.1 Minor bug fixes

- Minor internal improvements have been made.

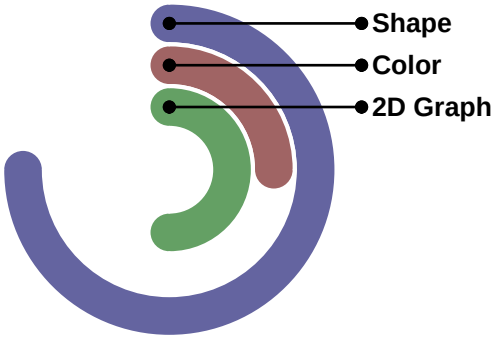
6.24.2 Documentation changes

- The *Depicting BFactor* has been updated to perceive residues when necessary.

6.25 Grapheme TK 1.2.2

6.25.1 New features

- *OEDrawROCSScores* function has been added to visualize scores.



- *OEAlignedDepictionFrom3DOptions* has been added to allow more customization of the *OEPrepareAlignedDepictionFrom3D* function.
- *OEShapeQueryDisplayOptions.GetDepictOrientation* and *OEShapeQueryDisplayOptions.SetDepictOrientation* methods have been added to allow the shape query to orient `horizontal`, `vertical`, or `square`.
- *OEShapeQueryDisplay.GetTitle* and *OEShapeOverlapDisplay.GetTitle* methods were added to access the title of the objects.
- *OEShapeQueryDisplay.IsValid* and *OEShapeOverlapDisplay.IsValid* methods were added to determine whether the objects have been initialized correctly.
- *OERenderShapeOverlap* function has been overloaded to enable rendering of shape overlap into two images: one image containing the fit molecule with transparent background and the other image showing the property map of the shape overlap. By superimposing the two images (molecule image on top), the result would be similar to visualizing the shape overlay in one image.
- *OE2DActiveSiteDisplay* class was rewritten to use the new implementation of the *OEBio::OEFragmentNetwork* class.

6.25.2 Major bug fixes

- *OEPrepareDepictionFrom3D* function has been improved to use a better scoring function in cases of atom clashes.

6.25.3 Minor bug fixes

- The default parameter for the maximum number of bond rotations of the *OEDepictionFrom3DOptions* class is now $2^{16} = 65536$. This means that by default only the first 16 single bonds of molecule will be rotated to find the "closest" 2D layout from the original 3D conformation when invoking the *OEPrepareDepictionFrom3D* function.
- *OEShapeOverlapDisplay* now limits the maximum number of bond rotations to $2^{14} = 16384$ when calling the *OEPrepareAlignedDepictionFrom3D* function to generate 2D coordinates aligned to the query molecule. This limit provides a reasonable trade-off between speed and the quality of the alignments.

- *OEPrepareDepictionFrom3D* and *OEPrepareAlignedDepictionFrom3D* have been optimized to be 10% faster on average. The algorithms were also revised to reduce the accumulation of floating point errors.
- Constructing *OEShapeQueryDisplay* and *OEShapeOverlapDisplay* objects is now significantly faster due to code optimization and constraining the 2D layout generations.
- *OERenderActiveSite* function was fixed to depict the `OEResidueIndex.I` residue with the DNA/RNA style.
- *OEShapeOverlapDisplay* and *OEShapeOverlapDisplay* now can handle queries without any color atoms.
- *OERenderShapeQuery* and *OERenderColorOverlap* now scale the line width of the pens used to render the circles representing color atom matches.
- *OEPrepareDepictionFrom3D* can now handle molecules containing atoms with an unknown atomic number (PDB examples: 1S9R, 3KXA, 1I1W).
- *OEGetMoleculeSurfaceScale* has been improved to capitalize on the molecule layout enhancements done in **OEDepict TK** in this release.
- *OERenderColorOverlap* was fixed to render the molecule surface of the query molecule as *OERenderShapeOverlap* does.

6.25.4 Documentation changes

- The *Depicting Shape and Color Atom Overlaps* example has been updated to visualize shape and color atom scores.
- The *Depicting Active Site Interactions* and *Depicting BFactor* examples have been updated to perceive residues in cases when it is not done by the molecule reader (for example, in the case of `mol2`).
- All images in this documentation have been automatically regenerated to reflect the changes made since the previous release.

6.26 Grapheme TK 1.2.1

6.26.1 Major bug fixes

- *OEPrepareDepictionFrom3D* has been optimized to be twice as fast on average.

6.26.2 Documentation changes

- The following examples have been updated to use `OESplitMolComplex` for separating the ligand and protein:
  - *bfactor2img* example
  - *complex2img* example
- All images in this documentation were automatically regenerated to reflect the changes made since the previous release.

6.27 Grapheme TK 1.2.0

6.27.1 New features

- The following classes and functions have been added for visualizing ligand-receptor interactions:
  - *OE2DActiveSiteDisplay* class
  - *OE2DActiveSiteDisplayOptions* class
  - *OE2DActiveSiteLegendDisplayOptions* class

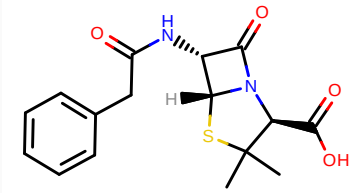
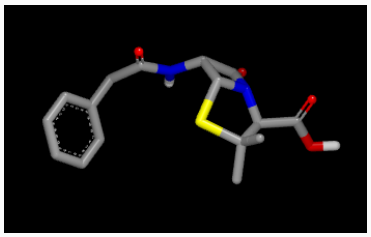
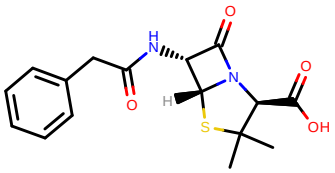
  - *OEPrepareActiveSiteDepiction* function
  - *OERenderActiveSite* function
  - *OEDrawActiveSiteLegend* function

- A new **Grapheme TK** example that visualizes receptor-ligand interactions has been added. For more details, see the *Depicting Active Site Interactions* section.

6.27.2 Major bug fixes

- When generating a 2D layout from a 3D conformation by using the *OEPrepareDepictionFrom3D* function, the configuration of the amide bonds are now always respected.

Table 7: **Example of generating a 2D layout from a 3D conformation**

Grapheme TK version 1.1.3		Grapheme TK version 1.2.0
		

6.27.3 Minor bug fixes

- The memory leak has been fixed and a check has been added to prevent division-by-zero in the *OEAddComplexSurfaceArcs* function.

6.27.4 Documentation changes

- All images in this documentation were automatically regenerated to reflect the changes made since the previous release.
- More C# examples have been added.

6.28 Grapheme TK 1.1.3

6.28.1 New features

- *OEGet2DSurfaceArcs* now has a radius scale parameter that allows the radii of the arcs to be easily scaled to different sizes.
- *OEGet2DSurfaceArcs* now allows the generation of molecule surfaces with a different radius for every atom. See *Listing 5* in *Drawing a Molecule Surface* chapter.
- The following classes do not require a valid **Shape TK** license any longer.
  - *OEColorForceFieldDisplay* class
  - *OEShapeQueryDisplay* class
  - *OEShapeOverlapDisplay* class

6.28.2 Major bug fixes

- When atom clashes are checked during the molecule alignment of the *OEPrepareAlignedDepictionFrom3D* function, every atom is now considered not just the heavy atoms.

6.28.3 Minor bug fixes

- *OEShapeOverlapDisplay*, *OEShapeQueryDisplay*, *OERenderColorOverlap*, *OERenderShapeOverlap* will no longer crash when dealing with improperly initialized objects.
- *OEPrepareDepictionFrom3D* performance has been improved.
- *OEPrepareDepictionFrom3D* will now just alter coordinates in-place, and not invalidate all OEAtomBase and OEBondBase pointers.
- Added parameter checks to the *OEAddComplexSurfaceArcs* function

6.28.4 Documentation changes

- All images in this documentation were automatically regenerated to reflect the changes made since the previous release.

6.29 Grapheme TK 1.1.2

6.29.1 New features

- Added *OEBondGlyphZigZag* and *OEBondGlyphStitch* bond annotation classes.
- Added the following methods to control the color of the property map generated by *OERenderShapeOverlap*:
  - *OEShapeOverlapDisplayOptions.GetOverlapColor*
  - *OEShapeOverlapDisplayOptions.SetOverlapColor*
- The maximum number of bond rotations performed by *OEPrepareDepictionFrom3D* when 2D depiction coordinates are generated based on 3D can be controlled by the following methods:
  - *OEDepictionFrom3DOptions.GetMaxBondRotations*
  - *OEDepictionFrom3DOptions.SetMaxBondRotations*

6.29.2 Major bug fixes

- The following classes and functions were made thread safe:
  - *OEShapeQueryDisplay*
  - *OERenderShapeQuery*
  - *OEShapeOverlapDisplay*
  - *OERenderShapeOverlap*
  - *OERenderColorOverlap*

  These functions were introduced in 2013.Oct and were not thread safe for the 2013.Oct and 2014.Feb releases.

6.29.3 Documentation changes

- All images were automatically regenerated to reflect the changes made since the previous release.
- Multiple examples migrated to the OpenEye Python Cookbook.

6.30 Grapheme TK 1.1.1

6.30.1 New features

- Added the following setter and getter methods that control whether a background color is cleared prior to rendering. This allows to generate an image with transparent background.
  - *OEShapeQueryDisplayOptions.GetClearBackground* and *OEShapeQueryDisplayOptions.SetClearBackground*
  - *OEShapeOverlapDisplayOptions.GetClearBackground* and *OEShapeOverlapDisplayOptions.SetClearBackground*
  - *OEColorOverlapDisplayOptions.GetClearBackground* and *OEColorOverlapDisplayOptions.SetClearBackground*

6.30.2 Minor bug fixes

- Added "BRIEF" line to interfaces of **Grapheme TK** examples.

6.30.3 Documentation changes

- Added example of drawing multiple 2D surfaces in the *Drawing a Molecule Surface* chapter.
- All images were automatically regenerated to reflect the changes made since the previous release.

6.31 Grapheme TK 1.1.0

6.31.1 New features

- Added the following to visualize color forcefields:
  - *OEColorForceFieldDisplay*
  - *OEColorForceFieldLegendDisplayOptions*
  - *OEDrawColorForceFieldLegend*
- Added the following to visualize the reference molecule of a color and shape overlap:
  - *OEShapeOverlapDisplay*
  - *OEShapeOverlapDisplayOptions*
  - *OERenderShapeQuery*
- Added the following to visualize shape and color overlaps:
  - *OEShapeOverlapDisplay*
  - *OEShapeOverlapDisplayOptions*
  - *OERenderShapeOverlap*
  - *OEColorOverlapDisplayOptions*
  - *OERenderColorOverlap*

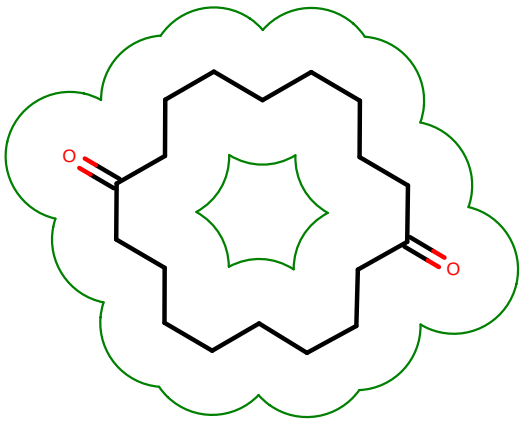
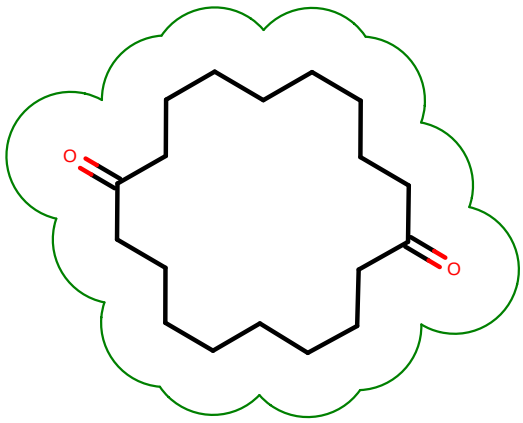
> **Note**
>
> A valid **shape** license is required to use any of these new shape or color APIs.

- Added the *OEDepictionFrom3DOptions* class that allows more advanced customization of the generation of 2D coordinates driven by the 3D conformation when calling the `OEPrepareDepictionFrom3D` function. By default, `OEPrepareDepictionFrom3D` is tuned for generating 2D coordinates for representing 3D overlays in 2D. This class allows the 2D coordinate generation to choose a set of 2D coordinates that are much closer to how the molecule would look in a 3D viewer.
- Added the following to render user-defined labels when depicting color gradients:
  - `OEColorGradientDisplayOptions.AddLabel`
  - `OEColorGradientDisplayOptions.ClearLabels`
  - `OEColorGradientDisplayOptions.GetLabels`
- Added `OEColorGradientDisplayOptions.ClearBoxRange` method to remove the box range previously set to a color gradient.
- Added `OEGet2DSurfaceArcs` function to return the arcs of the molecule surface of a specific atom with the given radius. Varying radii can be passed to generate multiple surfaces in the range specified by the `OESurfaceArcScale` namespace. `OEGetMoleculeSurfaceScale` can be used to determine the appropriate scale for the molecule with the specified radius.
- Implemented the *ShapeOverlap2PDF.cs* C# **Grapheme TK** example.

6.31.2 Major bug fixes

- Molecule surface arcs inside macrocycles are no longer returned. See the effect of this change in *Table: Example of depicting molecule surface*.

Table 8: **Example of depicting molecule surface**

Grapheme TK 1.0.6 (2013 Jun)	Grapheme TK 1.1.0 (2013 Oct)
	

6.31.3 Documentation changes

- All images were automatically regenerated to reflect the changes made since the previous release.

6.32 Grapheme TK 1.0.6

6.32.1 New features

- *OEPrepareAlignedDepictionFrom3D* performance improvement of 10% on average.
- The *OE2DPropMap* class now allows rendering bond properties along with atom ones. See examples in the *Table: Depicting atom or/and bond properties*.
- Implemented the *Bfactor2Img.cs* C# **Grapheme TK** example.

6.32.2 Major bug fixes

- The *OEDrawColorGradient* functions now takes an OEColorGradientBase object allowing to depict the following color gradients derived from the OEColorGradientBase base class:
  - OEExponentColorGradient
  - OEExponentialColorGradient
  - OELinearColorGradient
  - OELogarithmicColorGradient

6.32.3 Minor bug fixes

- Using a pen with miter line join in order to make the transition between grid cells smoother when rendering the property map by the *OE2DPropMap.Render* method.
- The *OE2DPropMap.SetResolution* lower limit has been reduced from *2* to *1*. Using resolution *1* will result in a very smooth property grid, however it significantly increase the size of vector images (such as `.pdf` and `.ps`).

6.32.4 Documentation changes

- Importing the following OESystem classes from the **OEChem TK** documentation.
  - OEColor
  - OEColorGradientBase
  - OEColorStop
  - OEExponentColorGradient
  - OEExponentialColorGradient
  - OELinearColorGradient
  - OELogarithmicColorGradient
- All images were automatically regenerated to reflect the changes made since the previous release.

6.33 Grapheme TK 1.0.5

6.33.1 New features

- Added the following methods to the *OEColorGradientDisplayOptions* class that allow the customization of the gradient visualization.

  - *OEColorGradientDisplayOptions.SetBoxRange* and *OEColorGradientDisplayOptions.SetBoxRangePen*
  - *OEColorGradientDisplayOptions.SetMarkedValuePen* and *OEColorGradientDisplayOptions.SetMarkedValuePrecision*
  - *OEColorGradientDisplayOptions.AddMarkedValues* and *OEColorGradientDisplayOptions.SetMarkedValues*
  - *OEColorGradientDisplayOptions.SetColorStopLabelFontScale*
- Added the *OEBondGlyphCross* and the *OEBondGlyphScissors* bond glyphs.

6.34 Grapheme TK 1.0.4

6.34.1 New features

- *OEDrawColorGradient* function and *OEColorGradientDisplayOptions* class added to visualize color gradients.

6.34.2 Major bug fixes

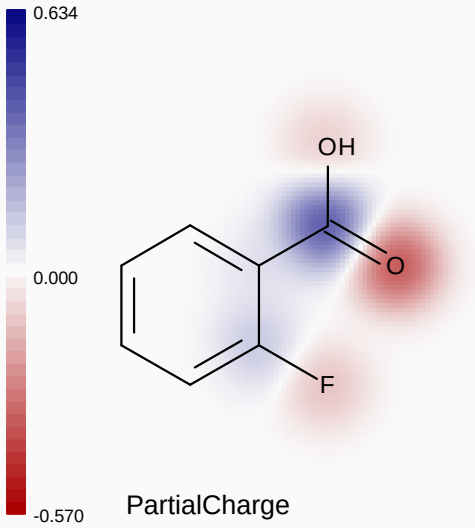
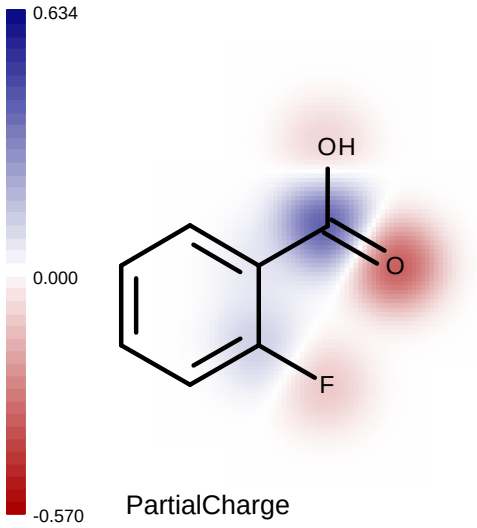
- *OEPrepareAlignedDepictionFrom3D* improved to produce better alignments in many cases.

6.34.3 Minor bug fixes

- *OEPrepareDepictionFrom3D* improved to produce 2D coordinates that align better with the input 3D coordinates.
- *OE2DPropMap* default background color changed from a very light grey to pure white.

  See the effect of this change in *Table: Example of depicting a property map*.

Table 9: **Example of depicting a property map**

Grapheme TK 1.0.3	Grapheme TK 1.0.4
	

6.34.4 Documentation changes

- All images were automatically regenerated to reflect the changes made since the previous release.

6.35 Grapheme TK 1.0.3

6.35.1 New features

- *OEBondGlyphCurvedArrow* added as a new bond annotation style.
- `OE2DPropMap.GetLegendFont` and `OE2DPropMap.GetLegendFontScale` methods added in order to customize the legend font of the property map.

6.35.2 Minor bug fixes

- *OE2DPropMap* legend layout improved.
- *OE2DPropMap* will now display at least one decimal number when rendering the value of the linear color gradient of the class.
- *OE2DPropMap* will now set the line width of the box around the linear color gradient to the default bond line width of the molecule.
- Increased the line width of the following complex arc styles: *OEDefaultSolventArcFxn*, *OEDefaultCavityArcFxn*, and *OEDefaultVoidArcFxn*.
- Atoms that are not visible are now not considered for the minimum and maximum range of the linear color gradient.
- Fixed cases where the property value of adjacent atoms had opposite signs and the colors were not blurred smoothly by increasing the size of the box used to blur the values.

6.35.3 Documentation changes

- All images were automatically regenerated to reflect the changes made since the previous release.

6.36 Grapheme TK 1.0.2

6.36.1 Minor bug fix

- Avoiding extraneous runtime warnings from `OEDrawPearlsCircle` and `OEDrawFlowerCircle` when the pen's fill is turned off.

6.36.2 Documentation changes

- Some examples and images were updated in order to consistently use red and blue color for representing negative and positive values respectively when using the OELinearColorGradient class.
- All images in this manual are now automatically generated as part of the release process and tested against benchmark images on all supported platforms.

6.37 Grapheme TK 1.0.1

6.37.1 Minor bug fixes

- Scaling the fonts of the legend of the *OE2DPropMap* class based upon the scale of the OE2DMolDisplay object.

6.38 Grapheme TK 1.0.0

6.38.1 New features

- Providing a mechanism to annotate atoms and bonds in the depicted molecule (see also *Annotating Atoms and Bonds* chapter)
- Implementing the *OEAtomGlyphCircle* atom annotation style and allowing user-defined styles that derive from the *OEAtomGlyphBase* abstract class
- Implementing the *OEBondGlyphArrow* and the *OEBondGlyphBase* bond annotation styles and allowing user-defined styles that derive from the: *OEBondGlyphBase* abstract class
- Adding 18 default circle drawing styles (`OECircleStyle`) that can be customized (see for example the `OEDrawCastleCircle` function)
- Adding 17 default arc drawing styles (`OESurfaceArcStyle`) that can be customized (see for example the `OEDrawCastleSurfaceArc` function)
- Adding the *OE2DPropMap* class that generates a 2D grid of user-defined atom properties (see *Depicting Atom Property Map* chapter).
- Adding functions that automatically generate standard interfaces for *Grapheme* applications (see `OEConfigure2DPropMap` and `OESetup2DPropMap` functions)
- Adding the `OEPrepareDepictionFrom3D` and the `OEPrepareAlignedDepictionFrom3D` functions in which the generation of the 2D coordinates of a molecule is driven by a given 3D reference molecule.
- Drawing molecule surfaces `OEDraw2DSurface` (see *Drawing Molecule Surface* chapter)
- Providing 18 built-in surface drawing styles and allowing to the implementation of user-defined ones.
- Drawing complex molecule surfaces where the distance between the ligand and the receptor is projected into the 2D molecule surface (`OEAddComplexSurfaceArcs`) Implementing the four default styles for drawing complex surfaces (*OEDefaultBuriedArcFxn*, *OEDefaultCavityArcFxn*, *OEDefaultSolventArcFxn* and *OEDefaultVoidArcFxn*) and handling user-defined ones (see *Drawing Ligand-Protein Complex Surface* chapter).

CHAPTER

SEVEN

COPYRIGHT AND TRADEMARKS

© 1997-2024 Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA. All rights reserved.

OpenEye, Lexichem, ROCS, Grapheme, and Orion are registered trademarks of Cadence Design Systems, Inc.

This material contains proprietary information of Cadence Design Systems, Inc. Use of copyright notice is precautionary only and does not imply publication or disclosure.

The information supplied in this document is believed to be true, but no liability is assumed for its use or the infringement of the rights of others resulting from its use. Information in this document is subject to change without notice and does not represent a commitment on the part of Cadence Design Systems, Inc.

This package is sold/licensed/distributed subject to the condition that it shall not, by way of trade or otherwise, be lent, re-sold, hired out, or otherwise circulated without Cadence Design Systems' prior consent, in any form of packaging or cover other than that in which it was produced. No part of this manual or accompanying documentation, may be reproduced, stored in a retrieval system on optical or magnetic disk, tape, CD, DVD or other medium, or transmitted in any form or by any means, electronic, mechanical, photocopying recording, or otherwise for any purpose other than for the purchaser's personal use without a legal agreement or other written permission granted by Cadence.

This product should not be used in the planning, construction, maintenance, operation or use of any nuclear facility nor the flight, navigation, or communication of aircraft or ground support equipment. Cadence Design Systems, Inc., shall not be liable, in whole or in part, for any claims arising from such use, including death, bankruptcy or outbreak of war.

Windows is a registered trademark of Microsoft Corporation. Apple, OS X, macOS, and Macintosh are registered trademarks of Apple Inc. UNIX is a registered trademark of the Open Group. Linux is a registered trademark of Linus Torvalds. Red Hat is a registered trademark of Red Hat, Inc. SUSE, SLED, and SLES are registered trademarks of Novell, Inc. Ubuntu is a registered trademark of Canonical Ltd.

SYBYL is a registered trademark of Certara, L.P. MDL, BIOVIA and ISIS are registered trademarks of Dassault Systemes and/or its affiliates. SMARTS and SMIRKS are registered trademarks of Daylight Chemical Information Systems, Inc. MacroModel is a registered trademark of Schrodinger, LLC.

Python is a registered trademark of the Python Software Foundation. Django is a registered trademark of the Django Software Foundation. Java is a registered trademark of Oracle and/or its affiliates.

Other products and software packages referenced in this document are trademarks and registered trademarks of their respective vendors or manufacturers.

CHAPTER

EIGHT

SAMPLE CODE

For all of the code examples and samples within documents published by OpenEye, Cadence Molecular Sciences, the following terms of use apply.

TERMS FOR USE OF SAMPLE CODE

The software below ("Sample Code") is provided to current licensees or subscribers of Cadence products or SaaS offerings (each a "Customer"). Customer is hereby permitted to use, copy, and modify the Sample Code, subject to these terms. Cadence claims no rights to Customer's modifications. Modification of Sample Code is at Customer's sole and exclusive risk. Sample Code may require Customer to have a then current license or subscription to the applicable Cadence offering.

THE SAMPLE CODE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. CADENCE DISCLAIMS ALL WARRANTIES, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. In no event shall Cadence be liable for any damages or liability in connection with the Sample Code or its use.

CHAPTER

NINE

CITATION

Please follow the guidelines in this document to cite use of our products in research publications and other documents.

For more information, please contact OpenEye Support.

9.1 Orion®

To cite use of Orion, please use the following:

```
Orion <version-number>. OpenEye, Cadence Molecular Sciences, Santa Fe, NM. http://www.
↪eyesopen.com.
```

For example:

```
Orion 2025.1. OpenEye, Cadence Molecular Sciences, Santa Fe, NM. http://www.eyesopen.com.
```

The version number for Orion appears in the product's release notes, for example, https://docs.eyesopen.com/orion-user-interface/releasenotes/index.html.

9.2 Orion Floes

To cite use of an Orion-based floe package, please use the following:

```
OpenEye <package-name> <version-number>. OpenEye, Cadence Molecular Sciences, Santa Fe,␣
↪NM. http://www.eyesopen.com.
```

For example:

```
OpenEye Large-Scale Floes 4.2.3. OpenEye, Cadence Molecular Sciences, Santa Fe, NM.␣
↪http://www.eyesopen.com.
```

The version number for a floe package is displayed on the first page of the package's release notes. For example: https://docs.eyesopen.com/floe/modules/large_scale_floes/docs/source/releasenotes_2025_1.html.

9.3 Toolkits and Applications

To cite OpenEye toolkits or applications used in your work, please use the following:

```
OpenEye Toolkits [or Applications] <version-number>. OpenEye, Cadence Molecular Sciences,
↪ Santa Fe, NM. http://www.eyesopen.com.
```

For example:

```
OpenEye Toolkits 2024.2.1. OpenEye, Cadence Molecular Sciences, Santa Fe, NM. http://www.
↪eyesopen.com.
```

The Applications and Toolkits are released as a package, multiple times per year. The first part of the version number designates the calendar year. The version number appears on the summary document for the package.

If you want to cite an individual application or toolkit, you can use the syntax:

```
<product name> <version-number>. OpenEye, Cadence Molecular Sciences, Santa Fe, NM.␣
↪http://www.eyesopen.com.
```

where <*product name*> is:

- Bioisostere TK
- OEChem TK
- OEDepict TK
- OEDocking TK
- OEFF TK
- Eon TK
- FastROCS TK
- Grapheme TK
- GraphSim TK
- Lexichem TK
- OEMedChem TK
- MolProp TK
- Omega TK
- Quacpac TK
- Shape TK
- Spicoli TK
- Spruce TK
- Szmap TK
- Szybki TK
- Zap TK
- AFITT Application
- BROOD Application
- OEDocking Application
- EON Application
- OMEGA Application
- PICTO Application
- pKa-Prospector Application
- QUACPAC Application

- ROCS Application
- SiteHopper Application
- SPRUCE Application
- SZMAP Application
- SZYBKI Application
- VIDA Application

You can use the version number of the individual Application or Toolkit. For example:

```
SZYBKI Application 2.8.0.1. OpenEye, Cadence Molecular Sciences, Santa Fe, NM. http://
↪www.eyesopen.com.

Szybki TK 2.8.0.1. OpenEye, Cadence Molecular Sciences, Santa Fe, NM. http://www.
↪eyesopen.com.
```

The first documentation pages give the product names and version numbers, as in https://docs.eyesopen.com/applications/szybki/index.html and https://docs.eyesopen.com/toolkits/python/szybkitk/index.html.

9.4 Publications for Bibliographies

Please also cite these publications if the corresponding product was used in your research.

9.4.1 Orion

Sørensen, J., Bannan, C.C., Calabrò, G., Jain, V., Ovanesyan, G., Smith, A., Zhang, S., Bayly, C.I., Darden, T.A., Geballe, M.T., LeBard, D.N., McGann, M., Moon, J.B., Muddana, H.S., Shewmaker, A., LaFon, J., Tolbert, R.W., Skillman, A.G., Nicholls, A. Orion® A Cloud-Native Molecular Design Platform. In *Computational Drug Discovery*. V. Poongavanam and V. Ramaswamy, Eds. Wiley-VCH, 2024; pp. 579–615. DOI: 10.1002/9783527840748.ch24

9.4.2 AFITT and FLYNN Applications

Wlodek, S.; Skillman, A.G.; Nicholls, A. Automated Ligand Placement and Refinement with a Combined Force Field and Shape Potential. *Acta Crystallogr., Sect. D: Struct. Biol.* **2006**, *62* (7), 741–749. DOI: 10.1107/s0907444906016076

9.4.3 OEDocking Application and Toolkit

Kelley, B.P.; Brown, S.P.; Warren, G.L.; Muchmore, S.W. POSIT: Flexible Shape-Guided Docking For Pose Prediction. *J. Chem. Inf. Model.* **2015**, *55* (8), 1771-1780. DOI: 10.1021/acs.jcim.5b00142

McGann, M. FRED Pose Prediction and Virtual Screening Accuracy. *J. Chem. Inf. Model.* **2011**, *51* (3), 578-596. DOI: 10.1021/ci100436p

McGann, M. FRED and HYBRID Docking Performance on Standardized Datasets. *J. Comput.-Aided Mol. Des.* **2012**, *26* (8), 897-906. DOI: 10.1007/s10822-012-9584-8

9.4.4 OMEGA Application and Toolkit

If you are using OMEGA in torsion driving mode, please cite the following two publications.

Hawkins, P.C.D.; Skillman, A.G.; Warren, G.L.; Ellingson, B.A.; Stahl, M.T. Conformer Generation with OMEGA: Algorithm and Validation Using High Quality Structures from the Protein Databank and the Cambridge Structural Database. *J. Chem. Inf. Model.* **2010**, *50* (4), 572-584. DOI: 10.1021/ci100031x

Hawkins, P.C.D. and Nicholls, A. Conformer Generation with OMEGA: Learning from the Data Set and the Analysis of Failures. *J. Chem. Inf. Model.* **2012**, *52* (11), 2919–2936. DOI: /10.1021/ci300314k

If you are using OMEGA in macrocyle mode, please cite the following.

Hawkins, P.C.D. and Wlodek, S. Decisions with Confidence: Application to the Conformation Sampling of Molecules in the Solid State. *J. Chem. Inf. Model.* **2020**, *60* (7), 3518–3533. DOI: /10.1021/acs.jcim.0c00358

9.4.5 ROCS Application

Hawkins, P.C.D.; Skillman, A.G.; Nicholls, A. Comparison of Shape-Matching and Docking as Virtual Screening Tools. *J. Med. Chem.* **2007**, *50* (1), 74-82. DOI: 10.1021/jm0603365

9.5 OpenEye MMDS Web Service

To cite use of the Macromolecular Data Service (MMDS) web service, please use the syntax:

```
Macromolecular Data Service <version-number>. OpenEye, Cadence Molecular Sciences, Santa␣
↪Fe, NM. http://www.eyesopen.com.
```

For example:

```
Macromolecular Data Service 1.1. OpenEye, Cadence Molecular Sciences, Santa Fe, NM.␣
↪http://www.eyesopen.com.
```

The MMDS version number appears on the web service's release notes.

CHAPTER

TEN

TECHNOLOGY LICENSING

OpenEye products use the following technology under license.

Some of the open source technologies we use are licensed under the GNU Public License (GPL) version 2 or 3. In all instances, these technologies are communicated with using the "at arms length" principle, using pipes and command line arguments.

While the software is in some instances assembled into a container, that assembly does not invoke the GPL copyleft scope. For each Floe package where these are used, the environment log clearly details the installed programs and their version numbers. With the links below for each technology, the source for each of those programs is available.

Table 1: Licensed Technology

Name of Project	Website	License URL
_libgcc_mutex	https://github.com/conda-forge/_libgcc_mutex-feedstock	https://github.com/conda-forge/_libgcc_mutex-feedstock/blob/master/LICENSE.txt
_openmp_mutex	https://github.com/conda-forge/_openmp_mutex-feedstock	https://github.com/conda-forge/_openmp_mutex-feedstock/blob/main/LICENSE.txt
ablang2	https://github.com/TobiasHeOl/AbLang2	https://github.com/TobiasHeOl/AbLang2/blob/main/LICENSE
abseil-cpp	https://github.com/abseil/abseil-cpp	https://github.com/abseil/abseil-cpp/blob/master/LICENSE
absl-py	https://github.com/abseil/abseil-py	https://github.com/abseil/abseil-py/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
aiohappyeyeballs	https://github.com/aio-libs/aiohappyeyeballs	https://github.com/aio-libs/aiohappyeyeballs/blob/main/LICENSE
aiohttp	https://docs.aiohttp.org/en/stable/	https://github.com/aio-libs/aiohttp/blob/master/LICENSE.txt
aiosignal	https://github.com/aio-libs/aiosignal	https://github.com/aio-libs/aiosignal/blob/master/LICENSE
Amazon Linux 2	https://aws.amazon.com/amazon-linux-2	N/A
AmberUtils	http://ambermd.org	N/A
ambit	https://github.com/khimaros/ambit	https://github.com/khimaros/ambit/blob/master/LICENSE
amqp	https://github.com/celery/py-amqp	https://github.com/celery/py-amqp/blob/master/LICENSE
anaconda-anon-usage	https://github.com/anaconda/anaconda-anon-usage	https://github.com/anaconda/anaconda-anon-usage/blob/main/LICENSE
angular	https://github.com/angular/angular.js	https://github.com/angular/angular.js/raw/master/LICENSE.md
angular-animate	https://github.com/angular/angular.js	https://github.com/angular/angular.js/raw/master/LICENSE.md
angular-cache	https://github.com/jmdobry/angular-cache	https://github.com/jmdobry/angular-cache/raw/master/LICENSE
angular-cookies	https://github.com/angular/angular.js	https://github.com/angular/angular.js/raw/master/LICENSE.md

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
angular-loggly-logger	https://github.com/ajbrown/angular-loggly-logger	https://github.com/ajbrown/angular-loggly-logger/raw/master/LICENSE.md
angular-mocks	https://github.com/angular/angular.js	https://raw.githubusercontent.com/angular/angular.js/master/LICENSE
angular-resource	https://github.com/angular/angular.js	https://raw.githubusercontent.com/angular/angular.js/master/LICENSE
angular-toggle-switch	https://github.com/cgarvis/angular-toggle-switch	https://github.com/cgarvis/angular-toggle-switch/blob/master/LICENSE
angular-ui-grid	https://github.com/angular-ui/ui-grid	https://github.com/angular-ui/ui-grid
angular-ui-router	https://github.com/angular-ui/ui-router	https://github.com/angular-ui/ui-router/raw/master/LICENSE
angular-ui-tree	https://github.com/angular-ui-tree/angular-ui-tree	https://github.com/angular-ui-tree/angular-ui-tree/raw/master/LICENSE
angular-vs-repeat	https://github.com/kamilkp/angular-vs-repeat	https://raw.githubusercontent.com/kamilkp/angular-vs-repeat/master/LICENSE
aniso8601	https://bitbucket.org/nielsenb/aniso8601	https://bitbucket.org/nielsenb/aniso8601/src/master/LICENSE
annotated-types	https://github.com/annotated-types/annotated-types	https://github.com/annotated-types/annotated-types/blob/main/LICENSE
antlr4-python3-runtime	http://www.antlr.org OR https://pypi.org/project/antlr4-python3-runtin	https://github.com/conda-forge/antlr4-python3-runtin/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
anyio	https://github.com/agronholm/anyio	https://github.com/agronholm/anyio/blob/master/LICENSE
appdirs	http://github.com/ActiveState/appdirs	http://github.com/ActiveState/appdirs/blob/master/LICENSE.txt
appengine	https://google.golang.org/appengine	https://github.com/golang/appengine/blob/master/LICENSE
arabic-reshaper	https://github.com/mpcabd/python-arabic-reshape	https://github.com/mpcabd/python-arabic-reshape/blob/master/LICENSE
archspec	https://github.com/archspec/archspec/blob/master/README.md	https://github.com/archspec/archspec/blob/master/COPYRIGHT
argon2-cffi	https://github.com/hynek/argon2-cffi	https://github.com/hynek/argon2-cffi/blob/master/LICENSE
argon2-cffi-bindings	https://github.com/hynek/argon2-cffi-bindings	https://github.com/hynek/argon2-cffi-bindings/blob/master/LICENSE
arpack	https://www.caam.rice.edu/software/ARPACK/	https://www.caam.rice.edu/software/ARPACK/
arrow	https://github.com/arrow-py/arrow	https://github.com/arrow-py/arrow?tab=Apache-2.0-1-ov-file#readme
ascii85	https://github.com/huandu/node-ascii85	https://github.com/huandu/node-ascii85/raw/master/LICENSE
ase	https://wiki.fysik.dtu.dk/ase/	https://wiki.fysik.dtu.dk/ase/development/licenseinfo.html#license-info
asgiref	https://github.com/django/asgiref/	https://github.com/django/asgiref/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
asn1crypto	https://github.com/wbond/asn1crypto	https://github.com/wbond/asn1crypto/blob/master/LICENSE
assertions go-render	https://github.com/smartystreets/assertions/internal/go-render/render	https://github.com/smartystreets/assertions/blob/master/internal/go-render/LICENSE
assertions oglmatchers	https://github.com/smartystreets/assertions/internal/oglematchers	https://github.com/smartystreets/assertions/blob/master/internal/oglematchers/LICENSE
assertions	https://github.com/smartystreets/assertions	https://github.com/smartystreets/assertions/blob/master/LICENSE.md
asttokens	https://github.com/gristlabs/asttokens	https://github.com/gristlabs/asttokens/blob/master/LICENSE
astunparse	https://github.com/simonpercivall/astunparse	https://github.com/simonpercivall/astunparse/blob/master/LICENSE
async-lru	https://github.com/aio-libs/async-lru	https://github.com/aio-libs/async-lru/blob/master/LICENSE
async-timeout	https://github.com/aio-libs/async-timeout	https://github.com/aio-libs/async-timeout/blob/master/LICENSE
atk-1.0	https://docs.gtk.org/atk/	https://gitlab.gnome.org/GNOME/atk/-/blob/master/COPYING
atomic	https://github.com/uber-go/atomic	https://github.com/uber-go/atomic/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
atomicwrites	https://github.com/untitaker/python-atomicwrites	https://github.com/untitaker/python-atomicwrites/blob/master/LICENSE
attrs	https://www.attrs.org/	https://github.com/python-attrs/attrs/blob/main/LICENSE
aws-sdk-go	https://github.com/aws/aws-sdk-go	https://github.com/aws/aws-sdk-go/blob/master/LICENSE.txt
babel	https://babel.pocoo.org/ OR https://github.com/python-babel/babel	https://github.com/python-babel/babel/blob/master/LICENSE OR https://babel.pocoo.org/en/latest/license.html#babel-license
Babel	https://github.com/python-babel/babel	https://github.com/python-babel/babel/blob/master/LICENSE
backcall	https://github.com/takluyver/backcall	https://github.com/takluyver/backcall/blob/master/LICENSE
backports	https://github.com/brandon-rhodes/backports	https://github.com/brandon-rhodes/backports/blob/master/LICENSE.TXT
backports.functools-lru-cache	https://github.com/jaraco/backports.functools_lru_cache	https://github.com/jaraco/backports.functools_lru_cache/blob/main/LICENSE
backports.zstd	https://github.com/rogdham/backports.zstd	https://github.com/rogdham/backports.zstd?tab=License-1-ov-file#readme
base62	https://github.com/kare/base62	https://github.com/kare/base62/blob/main/LICENSE
beautifulsoup4	https://www.crummy.com/software/BeautifulSoup/	https://www.crummy.com/software/BeautifulSoup/

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
billiard	https://github.com/celery/billiard	https://github.com/celery/billiard/blob/master/LICENSE.txt
biopython	https://biopython.org	https://github.com/biopython/biopython/blob/master/LICENSE.rst
biotite	https://github.com/biotite-dev/biotite/blob/master/README.rst	https://github.com/biotite-dev/biotite/blob/master/LICENSE.rst
bitarray	https://github.com/ilanschnell/bitarray	https://github.com/ilanschnell/bitarray/blob/master/LICENSE
bitset	https://github.com/willf/bitset	https://github.com/willf/bitset/blob/master/LICENSE
blas	https://www.netlib.org/blas/	https://www.netlib.org/blas/#_licensing
bleach	https://github.com/mozilla/bleach	https://github.com/mozilla/bleach/blob/main/LICENSE
blessings	https://github.com/erikrose/blessings	https://github.com/erikrose/blessings/blob/master/LICENSE
blinker	https://pythonhosted.org/blinker/	https://github.com/pallets-eco/blinker/blob/master/LICENSE
bokeh	https://github.com/bokeh/bokeh	https://github.com/bokeh/bokeh/blob/branch-4.0/LICENSE.txt
blosc	https://github.com/Blosc/python-blosc	https://github.com/Blosc/python-blosc/blob/master/LICENSE.txt
blosc2	https://github.com/Blosc/python-blosc2	https://github.com/Blosc/c-blosc2/blob/main/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
boltons	https://github.com/mahmoud/boltons	https://github.com/mahmoud/boltons/blob/master/LICENSE
Boltz	https://boltz.bio	https://github.com/jwohlwend/boltz/blob/main/LICENSE
boost	https://www.boost.org/doc/libs/1_71_0/libs/python/doc/html/index.html	https://www.boost.org/LICENSE_1_0.txt
boost-cpp	https://www.boost.org/doc/libs/1_71_0/libs/python/doc/html/index.html	https://www.boost.org/LICENSE_1_0.txt
bootstrap-vue	https://github.com/bootstrap-vue/bootstrap-vue	https://github.com/bootstrap-vue/bootstrap-vue/raw/master/LICENSE
boto3	https://github.com/boto/boto3	https://github.com/boto/boto3/blob/develop/LICENSE
botocore	https://github.com/boto/botocore	https://github.com/boto/botocore/blob/develop/LICENSE.txt
Bottleneck	https://bottleneck.readthedocs.io/en/latest/index.html	https://bottleneck.readthedocs.io/en/latest/license.html
Brotli	https://github.com/google/brotli	https://github.com/google/brotli/blob/master/LICENSE
brotli-bin	https://github.com/google/brotli	https://github.com/google/brotli/blob/master/LICENSE
brotli-python	http://python-hyper.org/projects/brotlipy/en/latest/	https://github.com/python-hyper/brotlicffi/blob/main/LICENSE
brotlicffi	https://github.com/python-hyper/brotlicffi	https://github.com/python-hyper/brotlicffi?tab=MIT-1-ov-file#readme
brotlipy	https://github.com/python-hyper/brotlicffi	https://github.com/python-hyper/brotlicffi/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
bson	http://github.com/py-bson/bson	https://github.com/py-bson/bson/blob/master/LICENSE_APACHE
bytefmt	https://code.cloudfoundry.org/bytefmt	https://github.com/cloudfoundry/bytefmt/blob/master/LICENSE
bzip2	https://www.bzip.org	https://spdx.org/licenses/bzip2-1.0.6.html
c-ares	https://github.com/c-ares/c-ares	https://github.com/c-ares/c-ares/blob/main/LICENSE.md
ca-certificates	https://github.com/conda-forge/ca-certificates-feedsto	https://wiki.mozilla.org/CA/Included_Certificates
cached-property	https://github.com/pydanny/cached-property	https://github.com/pydanny/cached-property/blob/master/LICENSE
cachetools	https://github.com/tkem/cachetools/	https://github.com/tkem/cachetools/blob/master/LICENSE
cairo	https://pycairo.readthedocs.io/en/latest/	https://github.com/pygobject/pycairo/blob/master/COPYING
canvas	https://github.com/Automattic/node-canvas	https://github.com/Automattic/node-canvas
cctbx	https://github.com/cctbx/cctbx_project	https://github.com/cctbx/cctbx_project/blob/master/cctbx/LICENSE_2_0.txt
celery	https://github.com/celery/celery	https://github.com/celery/celery/blob/master/LICENSE
Cerberus	https://docs.python-cerberus.org/en/stable/	https://docs.python-cerberus.org/en/stable/license.html
certifi	https://certifiio.readthedocs.io/en/latest/	https://wiki.mozilla.org/index.php?title=CA/IncludedCertificates&redirect=no

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
cffi	https://github.com/python-cffi	https://github.com/cffi/cffi/blob/master/COPYRIGHT
cftime	https://pypi.org/project/cftime/	https://github.com/Unidata/cftime/blob/master/LICENSE
chardet	https://github.com/chardet/chardet	https://github.com/chardet/chardet/blob/master/LICENSE
charset-normalizer	https://github.com/ousret/charset_normalizer	https://github.com/Ousret/charset_normalizer/blob/master/LICENSE
ChEMBL Structure Pipeline	https://github.com/chembl/ChEMBL_Structure_Pipeline/tree/master	https://github.com/chembl/ChEMBL_Structure_Pipeline/blob/master/LICENSE
choreographer	https://github.com/mazpie/choreographer	https://github.com/mazpie/choreographer/blob/main/LICENSE
chunkreader	https://github.com/jackc/chunkreader/v2	https://github.com/jackc/chunkreader/blob/master/LICENSE
click	https://palletsprojects.com/p/click/	https://github.com/pallets/click/blob/main/LICENSE.rst
click-completion	https://github.com/click-contrib/click-completion	https://github.com/click-contrib/click-completion/blob/master/LICENSE
click-didyoumean	https://github.com/click-contrib/click-didyoumean	https://github.com/click-contrib/click-didyoumean/blob/master/LICENSE
click-plugins	https://github.com/click-contrib/click-plugins	https://github.com/click-contrib/click-plugins/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
click-repl	https://github.com/untitaker/click-repl	https://github.com/click-contrib/click-repl/blob/master/LICENSE
cloudpickle	https://github.com/cloudpipe/cloudpickle	https://github.com/cloudpipe/cloudpickle/blob/master/LICENSE
cmake	https://cmake.org/	https://cmake.org/licensing/
colorama	https://github.com/tartley/colorama	https://github.com/tartley/colorama/blob/master/LICENSE.txt
comm	https://github.com/ipython/comm	https://github.com/ipython/comm/blob/master/LICENSE
compose	https://github.com/docker/compose	https://github.com/docker/compose/blob/v2/LICENSE
conda-content-trust	https://github.com/conda/conda-content-trust	https://github.com/conda/conda-content-trust/blob/main/LICENSE
conda-libmamba-solver	https://github.com/conda/conda-libmamba-solv	https://github.com/conda/conda-libmamba-solv blob/main/LICENSE
conda-pack	https://conda.github.io/conda-pack/	https://github.com/conda/conda-pack/blob/main/LICENSE.txt
conda-package-handling	https://github.com/conda/conda-package-handli	https://github.com/conda/conda-package-handli blob/main/LICENSE
conda-package-streaming	https://anaconda.org/conda-forge/conda-package-stream	https://github.com/conda/conda-package-stream blob/main/LICENSE
conda-token	https://anaconda.org/anaconda/conda-token	https://anaconda.org/anaconda/conda-token

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
conda_package_strear	https://github.com/conda/conda-package-streamtree/main/conda_package_streaming	https://github.com/conda/conda-package-streamtab=License-1-ov-file
confuse	https://github.com/beetbox/confuse	https://github.com/beetbox/confuse/blob/main/LICENSE
contourpy	https://github.com/contourpy/contourpy	https://github.com/contourpy/contourpy/blob/main/LICENSE
cpp-expected	https://github.com/TartanLlama/expected	https://github.com/TartanLlama/expected?tab=CC0-1.0-1-ov-file#readme
cpp-peglib	https://github.com/yhirose/cpp-peglib	https://github.com/yhirose/cpp-peglib/blob/master/LICENSE
cryptography	https://github.com/pyca/cryptography	https://github.com/pyca/cryptography/blob/main/LICENSE
cssselect2	https://github.com/Kozea/cssselect2/	https://github.com/Kozea/cssselect2/blob/master/LICENSE
cuda-bindings	https://nvidia.github.io/cuda-python/cuda-bindings/latest/install.html	https://github.com/NVIDIA/cuda-python/blob/main/cuda_bindings/LICENSE
cuda-pathfinder	https://nvidia.github.io/cuda-python/cuda-pathfinder/latest/	https://nvidia.github.io/cuda-python/cuda-pathfinder/latest/license.html
cudatoolkit	https://developer.nvidia.com/cuda-toolkit	https://docs.nvidia.com/cuda/eula/index.html
cuequivariance-ops-cu12	https://developer.nvidia.com/cuequivariance	https://github.com/NVIDIA/cuEquivariance/blob/main/LICENSES/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
cuequivariance-ops-torch-cu12	https://developer.nvidia.com/cuequivariance	https://github.com/NVIDIA/cuEquivariance/blob/main/LICENSES/LICENSE
cuequivariance-torch	https://docs.nvidia.com/cuda/cuequivariance/	https://github.com/NVIDIA/cuEquivariance/blob/main/LICENSES/LICENSE
cupy-cuda113	https://cupy.dev/	https://github.com/cupy/cupy/blob/main/LICENSE
cupy-cuda11x	https://cupy.dev/	https://github.com/cupy/cupy/?tab=MIT-1-ov-file#readme
curl	https://curl.se	https://curl.se/docs/copyright.html
cycler	https://github.com/matplotlib/cycler	https://github.com/matplotlib/cycler/blob/main/LICENSE
cyrus-sasl	https://github.com/cyrusimap/cyrus-sasl	https://github.com/cyrusimap/cyrus-sasl/blob/master/COPYING
Cython	https://cython.org	https://www.apache.org/licenses/LICENSE-2.0.html
d3	https://github.com/mbostock/d3	https://github.com/mbostock/d3/raw/master/LICENSE
dataclasses	https://github.com/ericvsmith/dataclasses	https://github.com/ericvsmith/dataclasses/blob/master/LICENSE.txt
ddsketch	http://github.com/datadog/sketches-py	https://github.com/DataDog/sketches-py/blob/master/LICENSE
debugpy	https://aka.ms/debugpy	https://github.com/microsoft/debugpy/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
decimal	https://github.com/shopspring/decimal	https://github.com/shopspring/decimal/blob/master/LICENSE
decorator	https://github.com/micheles/decorator	https://github.com/micheles/decorator/blob/master/LICENSE.txt
deepdiff	https://github.com/seperman/deepdiff	https://github.com/seperman/deepdiff/blob/master/LICENSE
deeptime	https://github.com/deeptime-ml/deeptime	https://github.com/deeptime-ml/deeptime/blob/main/LICENSE.txt
defusedxml	https://github.com/tiran/defusedxml	https://github.com/tiran/defusedxml/blob/main/LICENSE
dftd3	https://github.com/dftd3/tad-dftd3	https://github.com/dftd3/tad-dftd3/blob/main/LICENSE
dill	https://github.com/uqfoundation/dill	https://github.com/uqfoundation/dill/blob/master/LICENSE
distlib	https://distlib.readthedocs.io/en/latest/	https://github.com/pypa/distlib/blob/master/LICENSE.txt
distro	https://github.com/python-distro/distro	https://github.com/python-distro/distro/blob/master/LICENSE
Django	https://www.djangoproject.com/	https://github.com/django/django/blob/main/LICENSE
django-classy-tags	https://github.com/django-cms/django-classy-tags	https://github.com/django-cms/django-classy-tags/blob/master/LICENSE
django-cors-headers	https://github.com/adamchainz/django-cors-headers	https://github.com/adamchainz/django-cors-headers/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
django-csp	https://github.com/mozilla/django-csp	https://github.com/mozilla/django-csp/blob/main/LICENSE
django-extensions	https://github.com/django-extensions/django-extensions	https://github.com/django-extensions/django-extensions/blob/main/LICENSE
django-filter	https://github.com/carltongibson/django-filter/tree/master	https://github.com/carltongibson/django-filter/blob/main/LICENSE
django-redis	https://github.com/jazzband/django-redis	https://github.com/jazzband/django-redis/blob/master/LICENSE
django-taggit	https://github.com/jazzband/django-taggit	https://github.com/jazzband/django-taggit/blob/master/LICENSE
django-taggit-serializer	https://github.com/glemmaPaul/django-taggit-serializ	https://github.com/glemmaPaul/django-taggit-serializ blob/master/LICENSE
django-taggit-templatetags2	https://github.com/fizista/django-taggit-templat	https://github.com/fizista/django-taggit-templat blob/master/LICENSE.txt
djangorestframework	https://www.django-rest-framewor org/	https://github.com/encode/django-rest-framewor blob/master/LICENSE.md
dkh	https://psicode.org/psi4manual/master/dkh.html	https://github.com/psi4/dkh/blob/master/COPYING.LESSER
dm-tree	https://github.com/deepmind/tree	https://github.com/deepmind/tree/blob/master/LICENSE
docker-py	https://github.com/docker/docker-py/	https://github.com/docker/docker-py/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
docker-pycreds	https://github.com/shin-/dockerpy-creds	https://github.com/shin-/dockerpy-creds/blob/master/LICENSE
docopt	https://docopt.org	https://github.com/docopt/docopt/blob/master/LICENSE-MIT
docutils	https://docutils.sourceforge.io	https://docutils.sourceforge.io/COPYING.html
drf-dynamic-fields	https://github.com/dbrgn/drf-dynamic-fields	https://github.com/dbrgn/drf-dynamic-fields/blob/master/LICENSE
DuckDB	https://duckdb.org/	https://github.com/duckdb/duckdb?tab=MIT-1-ov-file#readme
edgembar	https://rutgerslbsr.gitlab.io/fe-toolkit/edgembar/index.html	https://gitlab.com/RutgersLBSR/fe-toolkit/-/blob/master/LICENSE
editdistance	https://github.com/roy-ht/editdistance	https://github.com/roy-ht/editdistance/blob/master/LICENSE
eigen	https://eigen.tuxfamily.org/index.php?title=Main_Page	https://eigen.tuxfamily.org/index.php?title=Main_Page#License
einops	https://github.com/arogozhnikov/einops	https://github.com/arogozhnikov/einops/blob/master/LICENSE
einx	https://github.com/fferflo/einx	https://github.com/fferflo/einx/blob/master/LICENSE
entrypoints	https://github.com/takluyver/entrypoints	https://github.com/takluyver/entrypoints/blob/master/LICENSE
errors	https://github.com/pkg/errors	https://github.com/pkg/errors/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
eslint-plugin	https://github.com/typescript-eslint/typescript-eslint	https://github.com/typescript-eslint/typescript-eslint/raw/master/LICENSE
et_xmlfile	https://foss.heptapod.net/openpyxl/et_xmlfile	https://foss.heptapod.net/openpyxl/et_xmlfile/-/blob/branch/default/LICENCE.rst
exceptiongroup	https://github.com/agronholm/exceptiongroup	https://github.com/agronholm/exceptiongroup/blob/main/LICENSE
execnet	https://execnet.readthedocs.io/en/latest/	https://github.com/pytest-dev/execnet/blob/master/LICENSE
executing	https://github.com/alexmojaki/executing	https://github.com/alexmojaki/executing/blob/master/LICENSE.txt
expat	https://libexpat.github.io	https://github.com/libexpat/libexpat/blob/master/expat/COPYING
fairscale	https://github.com/facebookresearch/fairscale	https://github.com/facebookresearch/fairscale/blob/main/LICENSE
fairseq	https://github.com/pytorch/fairseq	https://github.com/facebookresearch/fairseq/blob/main/LICENSE
fastjsonschema	https://github.com/horejsek/python-fastjsonschem	https://github.com/horejsek/python-fastjsonschem/blob/master/LICENSE
fastrlock	https://github.com/scoder/fastrlock	https://github.com/scoder/fastrlock/blob/master/LICENSE
fftw	https://www.fftw.org	commercial license

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
filebuffer	https://github.com/mattetti/filebuffer	https://github.com/mattetti/filebuffer/blob/master/LICENSE
filelock	https://py-filelock.readthedocs.io/en/latest/index.html	https://github.com/tox-dev/py-filelock/blob/main/LICENSE
finufft	https://github.com/flatironinstitute/finufft	https://github.com/flatironinstitute/finufft/blob/master/LICENSE
Flask	https://flask.palletsprojects.com/en/2.1.x/	https://flask.palletsprojects.com/en/2.1.x/license/
flatbuffers	https://google.github.io/flatbuffers/	https://github.com/google/flatbuffers/blob/master/LICENSE.txt
flexcache	https://github.com/hgrecco/flexcache	https://github.com/hgrecco/flexcache/blob/main/LICENSE
flexparser	https://github.com/hgrecco/flexparser	https://github.com/hgrecco/flexparser/blob/main/LICENSE
flit-core	https://github.com/pypa/flit	https://github.com/scientific-python/lazy_loader/blob/main/LICENSE.md
FLTK	https://www.fltk.org/index.php	https://www.fltk.org/COPYING.php
fmt	https://fmt.dev/latest/index.html	https://github.com/fmtlib/fmt/blob/master/LICENSE.rst
font-awesome	https://github.com/FortAwesome/Font-Awesome	https://github.com/FortAwesome/Font-Awesome
font-ttf-dejavu-sans-mono	https://dejavu-fonts.github.io	https://dejavu-fonts.github.io/License.html
font-ttf-inconsolata	https://fonts.google.com/specimen/Inconsolata	https://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
font-ttf-source-code-pro	https://fonts.google.com/specimen/Source+Code+Pro	https://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=OFL
font-ttf-ubuntu	https://fonts.google.com/specimen/Ubuntu	https://ubuntu.com/legal/font-licence
fontconfig	https://www.freedesktop.org/wiki/Software/fontconfig/	https://gitlab.freedesktop.org/fontconfig/fontconfig/-/blob/main/COPYING
fonts-conda-ecosystem	https://anaconda.org/conda-forge/fonts-conda-ecosyster	https://docs.conda.io/en/latest/license.html
fonts-conda-forge	https://anaconda.org/conda-forge/fonts-conda-forge/	https://creativecommons.org/licenses/by/4.0/
fonttools	https://github.com/fonttools/fonttools	https://github.com/fonttools/fonttools/blob/main/LICENSE
forcebalance	https://simtk.org/home/forcebalance OR https://github.com/leeping/forcebalance	https://github.com/leeping/forcebalance/blob/master/LICENSE
fqdn	https://github.com/ypcrts/fqdn	https://github.com/ypcrts/fqdn?tab=MPL-2.0-1-ov-file#readme
freetype	https://freetype.org	https://freetype.org/license.html
freetype-py	https://github.com/rougier/freetype-py	https://github.com/rougier/freetype-py?tab=License-1-ov-file#readme
fribidi	https://github.com/fribidi/fribidi	https://github.com/fribidi/fribidi/blob/master/COPYING
frozendict	https://github.com/Marco-Sulla/python-frozendict	https://github.com/Marco-Sulla/python-frozendict/blob/master/LICENSE.txt
frozenlist	https://github.com/aio-libs/frozenlist	https://github.com/aio-libs/frozenlist/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
fsmlite	https://github.com/tkem/fsmlite	https://github.com/tkem/fsmlite/blob/master/LICENSE
fsspec	https://github.com/fsspec/filesystem_spec	https://github.com/fsspec/filesystem_spec/blob/master/LICENSE
funcy	https://github.com/Suor/funcy	https://github.com/Suor/funcy/blob/master/LICENSE
gast	https://github.com/serge-sans-paille/gast/	https://github.com/serge-sans-paille/gast/blob/master/LICENSE
gau2grid	https://github.com/dgasmith/gau2grid	https://github.com/dgasmith/gau2grid/blob/master/LICENSE
gax-go	https://github.com/googleapis/gax-go/v2	https://github.com/googleapis/gax-go/blob/main/LICENSE
gdk-pixbuf	https://gitlab.gnome.org/GNOME/gdk-pixbuf	https://gitlab.gnome.org/GNOME/gdk-pixbuf/-/blob/master/COPYING
gemmi	https://gemmi.readthedocs.io/en/latest/	https://github.com/project-gemmi/gemmi/blob/master/LICENSE.txt
genproto	https://google.golang.org/genproto/googleapis	https://github.com/googleapis/go-genproto/blob/main/LICENSE
geometric	https://openbase.com/python/geometric	https://openbase.com/python/geometric
giflib	https://giflib.sourceforge.net	https://github.com/mirrorer/giflib/blob/master/COPYING
gitdb	https://github.com/gitpython-developers/gitdb	https://github.com/gitpython-developers/gitdb/blob/master/LICENSE
GitPython	https://github.com/gitpython-developers/GitPython	https://github.com/gitpython-developers/GitPython/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
glib	https://docs.gtk.org/glib/	https://gitlab.gnome.org/GNOME/glib/-/blob/main/LICENSES/LGPL-2.1-or-later.txt
GLM Library	https://github.com/g-truc/glm	https://github.com/g-truc/glm/blob/master/manual.md#section0
gls	https://github.com/jtolds/gls	https://github.com/jtolds/gls/blob/master/LICENSE
gmpy2	https://github.com/aleaxit/gmpy	https://github.com/gmpy2/gmpy2/blob/master/COPYING.LESSER
go-cleanhttp	https://github.com/hashicorp/go-cleanhttp	https://github.com/hashicorp/go-cleanhttp/blob/master/LICENSE
go-connections	https://github.com/docker/go-connections	https://github.com/docker/go-connections/blob/master/LICENSE
go-cpio	https://github.com/cavaliercoder/go-cpio	https://github.com/cavaliercoder/go-cpio/blob/master/LICENSE
go-getter	https://github.com/hashicorp/go-getter	https://github.com/hashicorp/go-getter/blob/master/LICENSE
go-homedir	https://github.com/mitchellh/go-homedir	https://github.com/mitchellh/go-homedir/blob/master/LICENSE
go-ini	https://github.com/go-ini/ini	https://github.com/go-ini/ini/blob/master/LICENSE
go-jmespath	https://github.com/jmespath/go-jmespath	https://github.com/jmespath/go-jmespath/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
go-junit-report	https://github.com/jstemmer/go-junit-report	https://github.com/jstemmer/go-junit-report/blob/master/LICENSE
go-netrc	https://github.com/bgentry/go-netrc/netrc	https://github.com/bgentry/go-netrc/blob/master/LICENSE
go-ole	https://github.com/go-ole/go-ole	https://github.com/go-ole/go-ole/blob/master/LICENSE
go-pg	https://github.com/go-pg/pg	https://github.com/go-pg/pg/blob/master/LICENSE
go-redis	https://github.com/go-redis/redis/v8	https://github.com/go-redis/redis/blob/master/LICENSE
go-rendezvous	https://github.com/dgryski/go-rendezvous	https://github.com/dgryski/go-rendezvous/blob/master/LICENSE
go-safetemp	https://github.com/hashicorp/go-safetemp	https://github.com/hashicorp/go-safetemp/blob/master/LICENSE
go-sysconf	https://github.com/tklauser/go-sysconf	https://github.com/tklauser/go-sysconf/blob/master/LICENSE
go-testing-interface	https://github.com/mitchellh/go-testing-interface	https://github.com/mitchellh/go-testing-interface/blob/master/LICENSE
go-units	https://github.com/docker/go-units	https://github.com/docker/go-units/blob/master/LICENSE
go-version	https://github.com/hashicorp/go-version	https://github.com/hashicorp/go-version/blob/master/LICENSE
go-zglob	https://github.com/mattn/go-zglob	https://github.com/mattn/go-zglob/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
go.opencensus	https://go.opencensus.io	https://github.com/census-instrumentatio opencensus-go/blob/master/LICENSE
gobrake.v2	https://gopkg.in/airbrake/gobrake.v2	https://github.com/airbrake/gobrake/blob/v2.0.9/LICENSE
goconvey	https://github.com/smartystreets/goconvey	https://github.com/smartystreets/goconvey/blob/master/LICENSE.md
golden-layout	https://github.com/deepstreamIO/golden-layout	https://github.com/deepstreamIO/golden-layout/raw/master/LICENSE
google-auth	https://google-auth.readthedocs.io/en/master/	https://github.com/GoogleCloudPlatform google-auth-library-p blob/master/LICENSE
google-auth-oauthlib	https://github.com/googleapis/google-auth-library-p	https://github.com/googleapis/google-auth-library-p blob/main/LICENSE
google-cloud-go	https://cloud.google.com/go	https://github.com/googleapis/google-cloud-go/blob/main/LICENSE
google-cloud-go/storage	https://cloud.google.com/go/storage	https://github.com/googleapis/google-cloud-go/blob/main/LICENSE
google-pasta	https://github.com/google/pasta	https://github.com/google/pasta/blob/master/LICENSE
google.golang.org/api	https://google.golang.org/api	https://github.com/googleapis/google-api-go-client/blob/main/LICENSE
gopsutil	https://github.com/shirou/gopsutil	https://github.com/shirou/gopsutil/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
gorilla websocket	https://github.com/gorilla/websocket	https://github.com/gorilla/websocket/blob/master/LICENSE
gpytorch	https://gpytorch.ai	https://github.com/cornellius-gp/gpytorch?tab=MIT-1-ov-file#readme
graphite2	https://github.com/silnrsi/graphite	https://github.com/silnrsi/graphite/blob/master/LICENSE
graphviz	https://graphviz.org/	https://graphviz.org/license/
greenlet	https://greenlet.readthedocs.io/en/latest/	https://github.com/python-greenlet/greenlet/blob/master/LICENSE
groupcache	https://github.com/golang/groupcache	https://github.com/golang/groupcache/blob/master/LICENSE
grpc	https://google.golang.org/grpc	https://github.com/grpc/grpc-go/blob/master/LICENSE
grpc-cpp	https://github.com/grpc/grpc	https://github.com/grpc/grpc/blob/master/LICENSE
grpcio	https://github.com/grpc/grpc.io/blob/main/LICENSE	https://github.com/grpc/grpc.io/blob/main/LICENSE
gtk2	https://gitlab.gnome.org/GNOME/gtk	https://gitlab.gnome.org/GNOME/gtk/-/blob/main/COPYING
gts	https://sourceforge.net/projects/gts/	https://www.gnu.org/licenses/lgpl-3.0.html
h11	https://github.com/python-hyper/h11	https://github.com/python-hyper/h11/blob/master/LICENSE.txt
h2	https://github.com/python-hyper/h2	https://github.com/python-hyper/h2?tab=MIT-1-ov-file#readme

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
h5py	https://www.h5py.org	https://docs.h5py.org/en/stable/licenses.html
harfbuzz	https://github.com/harfbuzz/harfbuzz	https://github.com/harfbuzz/harfbuzz/blob/main/COPYING
hdbscan	https://hdbscan.readthedocs.io/en/latest/	https://github.com/scikit-learn-contrib/hdbscan/blob/master/LICENSE
hdf4	https://www.hdfgroup.org/solutions/hdf4/	https://github.com/HDFGroup/hdf4/blob/master/COPYING
hdf5	https://www.hdfgroup.org/solutions/hdf5/	https://github.com/HDFGroup/hdf5/blob/master/COPYING
he	https://github.com/mathiasbynens/he	https://github.com/mathiasbynens/he/raw/master/LICENSE-MIT.txt
hpack	https://github.com/python-hyper/hpack	https://github.com/python-hyper/hpack?tab=MIT-1-ov-file#readme
html-loader	https://github.com/webpack-contrib/html-loader	https://github.com/webpack-contrib/html-loader/raw/master/LICENSE
html5lib	https://github.com/html5lib/html5lib-python	https://github.com/html5lib/html5lib-python/blob/master/LICENSE
htslib	https://www.htslib.org	https://github.com/samtools/htslib/blob/develop/LICENSE
httpcore	https://www.encode.io/httpcore/	https://github.com/encode/httpcore/blob/master/LICENSE.md
httpx	https://github.com/encode/httpx	https://github.com/encode/httpx/blob/master/LICENSE.md

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
humanize	https://github.com/jmoiron/humanize	https://github.com/jmoiron/humanize/blob/master/LICENCE
hydra-core	https://github.com/facebookresearch/hydra	https://github.com/facebookresearch/hydra/blob/main/LICENSE
hyperframe	https://github.com/python-hyper/hyperframe	https://github.com/python-hyper/hyperframe?tab=MIT-1-ov-file#readme
icu	https://github.com/unicode-org/icu	https://github.com/unicode-org/icu/blob/main/icu4c/LICENSE
idna	https://github.com/kjd/idna	https://github.com/kjd/idna/blob/master/LICENSE.md
ihm	https://github.com/ihmwg/python-ihm	https://github.com/ihmwg/python-ihm/blob/main/LICENSE
imageio	https://github.com/imageio/imageio	https://github.com/imageio/imageio/blob/master/LICENSE
imagesize	https://github.com/shibukawa/imagesize_py	https://github.com/shibukawa/imagesize_py/blob/master/LICENSE.rst
ImmuneBuilder	https://github.com/oxpig/ImmuneBuilder/tree/main	https://github.com/oxpig/ImmuneBuilder/blob/main/LICENSE
importlib-metadata	https://github.com/python/importlib_metadata	https://github.com/python/importlib_metadata/blob/main/LICENSE
importlib_metadata	https://github.com/python/importlib_metadata	https://github.com/python/importlib_metadata/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
importlib_resources	https://github.com/python/importlib_resources	https://github.com/python/importlib_resources/blob/main/LICENSE
InChI	https://iupac.org/who-we-are/divisions/division-details/inchi/	https://iupac.org/wp-content/uploads/2021/06/iupac-inchi-license_2020.pdf
inflection	https://github.com/jinzhu/inflection	https://github.com/jinzhu/inflection/blob/master/LICENSE
ini.v1	https://gopkg.in/ini.v1	https://github.com/go-ini/ini/blob/v1.66.2/LICENSE
iniconfig	https://github.com/pytest-dev/iniconfig	https://github.com/pytest-dev/iniconfig/blob/main/LICENSE
innersvg-polyfill	https://github.com/dnozay/innersvg-polyfill	https://github.com/dnozay/innersvg-polyfill/raw/master/LICENSE
intel-openmp	https://github.com/hermitcore/libomp_oss	https://github.com/hermitcore/libomp_oss/blob/hermit/LICENSE.txt
ipykernel	https://ipython.org	https://ipython.org
ipython	https://ipython.org	https://github.com/ipython/ipython/blob/master/LICENSE
ipython-genutils	http://ipython.org	https://github.com/ipython/ipython_genutils/blob/master/COPYING.md
ipython_pygments_le	https://github.com/ipython/ipython-pygments-lex	https://github.com/ipython/ipython-pygments-lex blob/main/LICENSE
ipywidgets	http://jupyter.org	https://github.com/jupyter-widgets/ipywidgets/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
isodate	https://github.com/gweis/isodate/	https://github.com/gweis/isodate/blob/master/LICENSE
isoduration	https://github.com/bolsote/isoduration	https://github.com/bolsote/isoduration?tab=ISC-1-ov-file#readme
itsdangerous	https://palletsprojects.com/p/itsdangerous/	https://itsdangerous.palletsprojects.com/en/2.1.x/license/
jax	https://github.com/google/jax	https://github.com/google/jax/blob/main/LICENSE
jaxlib	https://github.com/google/jax	https://github.com/google/jax/blob/main/LICENSE
jaxtyping	https://github.com/patrick-kidger/jaxtyping	https://github.com/patrick-kidger/jaxtyping?tab=License-1-ov-file#readme
jedi	https://jedi.readthedocs.io/en/latest/	https://github.com/davidhalter/jedi/blob/master/LICENSE.txt
Jinja2	https://palletsprojects.com/p/jinja/	https://github.com/pallets/jinja/blob/main/LICENSE.rst
jmespath	https://github.com/jmespath/jmespath.py	https://github.com/jmespath/jmespath.py/blob/develop/LICENSE.txt
joblib	https://joblib.readthedocs.io/en/latest/	https://github.com/joblib/joblib/blob/master/LICENSE.txt
jpeg	https://www.ijg.org	https://www.ijg.org/files/README
json5	https://github.com/dpranke/pyjson5	https://github.com/dpranke/pyjson5/blob/main/LICENSE
jsonfield	https://github.com/dmkoch/django-jsonfield/	https://github.com/rpkilby/jsonfield/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
jsonpatch	https://github.com/stefankoegl/python-json-patch	https://github.com/stefankoegl/python-json-patch/blob/master/LICENSE
jsonpickle	https://github.com/jsonpickle/jsonpickle	https://github.com/jsonpickle/jsonpickle/blob/main/LICENSE
jsonpointer	https://github.com/stefankoegl/python-json-pointer	https://github.com/stefankoegl/python-json-pointer/blob/master/LICENSE.txt
jsonschema	https://github.com/python-jsonschema/jsonschema	https://github.com/python-jsonschema/jsonschema/blob/master/COPYING
jsonschema-specifications	https://github.com/python-jsonschema/jsonschema-specificat	https://github.com/python-jsonschema/jsonschema-specificat blob/main/COPYING
jstat	https://github.com/jstat/jstat	https://www.opensource.org/licenses/mit-license.php
jupyter-client	https://jupyter.org	https://github.com/jupyter/jupyter_client/blob/main/COPYING.md
jupyter-core	https://jupyter.org	https://github.com/jupyter/jupyter_core/blob/master/COPYING.md
jupyter-events	https://github.com/jupyter/jupyter_events	https://github.com/jupyter/jupyter_events/blob/main/LICENSE
jupyter-lsp	https://github.com/jupyter-lsp/jupyterlab-lsp	https://github.com/jupyter-lsp/jupyterlab-lsp
jupyter-serverhttp://jupyter.or	https://github.com/jupyter-server/jupyter_server/blob/main/LICENSE	
jupyter_client	http://jupyter.org	https://github.com/jupyter/jupyter_client/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
jupyter_core	http://jupyter.org	https://github.com/jupyter/jupyter_core/blob/master/LICENSE
jupyter_server	https://github.com/jupyter-server/jupyter_server	https://github.com/jupyter-server/jupyter_server/blob/main/LICENSE
jupyter_server_termin	https://github.com/jupyter-server/jupyter_server_terminals	https://github.com/jupyter-server/jupyter_server_terminals/blob/main/LICENSE
jupyterlab	https://github.com/jupyterlab/jupyterlab	https://github.com/jupyterlab/jupyterlab/blob/main/LICENSE
jupyterlab-pygments	http://jupyter.org	https://github.com/jupyterlab/jupyterlab_pygments/blob/main/LICENSE
jupyterlab-widgets	http://jupyter.org	https://github.com/jupyter-widgets/ipywidgets/blob/master/LICENSE
jupyter-lab_pygments	https://github.com/jupyterlab/jupyterlab_pygments	https://github.com/jupyterlab/jupyterlab_pygments?tab=BSD-3-Clause-1-ov-file#readme
jupyterlab_server	https://github.com/jupyterlab/jupyterlab_server	https://github.com/jupyterlab/jupyterlab_server/blob/main/LICENSE
jupyterlab_widgets	https://github.com/jupyter-widgets/ipywidgets	https://github.com/jupyter-widgets/ipywidgets?tab=BSD-3-Clause-1-ov-file#readme
kaleido	https://github.com/plotly/Kaleido	https://github.com/plotly/Kaleido/blob/master/LICENSE.txt
keras	https://github.com/keras-team/keras	https://github.com/keras-team/keras/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
Keras-Preprocessing	https://github.com/keras-team/keras-preprocessing	https://github.com/keras-team/keras-preprocessing/blob/master/LICENSE
keras-tuner	https://github.com/keras-team/keras-tuner	https://github.com/keras-team/keras-tuner/blob/master/LICENSE
keyring	https://github.com/jaraco/keyring	https://github.com/jaraco/keyring/blob/main/LICENSE
keyutils	https://github.com/sassoftware/python-keyutils	https://github.com/sassoftware/python-keyutils/blob/master/LICENSE
kiwisolver	https://kiwisolver.readthedocs.io/en/latest/	https://github.com/nucleic/kiwi/blob/main/LICENSE
kombu	https://kombu.readthedocs.io	https://github.com/celery/kombu/blob/master/LICENSE
krb5	https://web.mit.edu/kerberos/	https://github.com/krb5/krb5/blob/master/src/lib/gssapi/LICENSE
kt-legacy	https://github.com/haifeng-jin/kt-legacy	https://github.com/haifeng-jin/kt-legacy/blob/main/LICENSE
lark	https://github.com/lark-parser/lark	https://github.com/lark-parser/lark?tab=MIT-1-ov-file#readme
lazy_loader	https://github.com/scientific-python/lazy_loader	https://github.com/scientific-python/lazy_loader/blob/main/LICENSE.md
lcms2	https://www.littlecms.com	https://github.com/sk1project/python-lcms2/blob/master/LICENSE
legacy-cgi	https://github.com/jackrosenthal/legacy-cgi	https://github.com/jackrosenthal/legacy-cgi?tab=License-1-ov-file#readme

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
lerc	https://github.com/Esri/lerc	https://github.com/Esri/lerc/blob/master/LICENSE
libarchive	http://www.libarchive.org	https://raw.githubusercontent.com/libarchive/libarchive/master/COPYING
libblas	https://www.netlib.org/blas/	https://www.netlib.org/blas/#_licensing
libbrotlicommon	https://github.com/google/brotli	https://github.com/google/brotli/blob/master/LICENSE
libbrotlidec	https://github.com/google/brotli	https://github.com/google/brotli/blob/master/LICENSE
libbrotlienc	https://github.com/google/brotli	https://github.com/google/brotli/blob/master/LICENSE
libcblas	https://anaconda.org/conda-forge/libcblas	N/A
libclang	https://github.com/sighingnow/libclang	https://github.com/sighingnow/libclang/blob/master/LICENSE.TXT
libcurl	https://curl.se/libcurl/	https://curl.se/docs/copyright.html
libcxx	https://github.com/llvm-mirror/libcxx	https://github.com/llvm-mirror/libcxx/blob/master/LICENSE.TXT
libdb	https://www.oracle.com/technology/software/products/berkeley-db/index.html	https://github.com/berkeleydb/libdb/blob/master/LEGAL
libdeflate	https://github.com/ebiggers/libdeflate	https://github.com/ebiggers/libdeflate/blob/master/COPYING
libedit	https://thrysoee.dk/editline/	http://www.netbsd.org/Goals/redistribution.html
libev	https://software.schmorp.de/pkg/libev.html	https://software.schmorp.de/pkg/libev.html

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
libexpat	https://libexpat.github.io/	https://github.com/libexpat/libexpat?tab=MIT-1-ov-file#readme
libffi	https://github.com/libffi/libffi	https://github.com/libffi/libffi/blob/master/LICENSE
libgcc	https://gcc.gnu.org/onlinedocs/gccint/Libgcc.html	https://gcc.gnu.org/onlinedocs/libstdc++/manual/license.html
libgcrypt	https://gnupg.org/software/index.html	https://git.gnupg.org/cgi-bin/gitweb.cgi?p=libgcrypt.git;a=blob;f=COPYING.LIB;hb=HEAD
libgd	https://libgd.github.io	https://libgd.github.io/manuals/2.2.1/files/license-txt.html
libglib	https://github.com/PolMine/libglib	https://github.com/PolMine/libglib/blob/master/include/glib-2.0/glib.h
libiconv	https://www.gnu.org/software/libiconv/	https://www.gnu.org/software/libiconv/#downloading
libint	https://tinyurl.com/yvw97wbw	https://tinyurl.com/yvw97wbw
liblapack	http://www.netlib.org/lapack/	https://www.netlib.org/lapack/LICENSE.txt
liblapacke	https://anaconda.org/conda-forge/liblapacke	https://www.netlib.org/lapack/LICENSE.txt
liblzma	https://tukaani.org/xz/	https://git.tukaani.org/?p=xz.git;a=blob;f=COPYING.0BSD
libmamba	https://medium.com/@QuantStack/open-software-packag	https://github.com/mamba-org/mamba/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
libmambapy	https://www.anaconda.com/blog/a-faster-conda-for-a-g	https://github.com/conda/conda-libmamba-solv blob/main/LICENSE
libnetcdf	https://www.unidata.ucar.edu/software/netcdf/	https://www.unidata.ucar.edu/software/netcdf/copyright.html
libnghttp2	https://github.com/nghttp2/nghttp2	https://github.com/nghttp2/nghttp2/blob/master/COPYING
libnsl	https://github.com/thkukuk/libnsl	https://gcc.gnu.org/onlinedocs/libstdc++/manual/license.html
libopenblas	https://www.openblas.net/	https://github.com/xianyi/OpenBLAS/blob/develop/LICENSE
libpng	https://www.libpng.org/pub/png/libpng.html	https://www.libpng.org/pub/png/src/libpng-LICENSE.txt
libpq	https://www.postgresql.org/	https://www.postgresql.org/about/licence/
librsvg	https://gitlab.gnome.org/GNOME/librsvg	https://gitlab.gnome.org/GNOME/librsvg/-/blob/main/COPYING.LIB
libsolv	https://github.com/openSUSE/libsolv	https://github.com/openSUSE/libsolv/blob/master/LICENSE.BSD
libsqlite	https://github.com/mackyle/sqlite	https://github.com/mackyle/sqlite?tab=License-1-ov-file#readme
libssh2	https://github.com/libssh2/libssh2	https://github.com/libssh2/libssh2/blob/master/COPYING
libstdcxx	https://gcc.gnu.org/	https://gcc.gnu.org/onlinedocs/libstdc++/manual/license.html

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
libtiff	https://www.libtiff.org/	https://www.libtiff.org/misc.html
libtrust	https://github.com/docker/libtrust	https://github.com/docker/libtrust/blob/master/LICENSE
libuuid	https://sourceforge.net/projects/libuuid/	https://sourceforge.net/p/libuuid/code/ci/master/tree/COPYING
libuv	https://github.com/libuv/libuv	https://github.com/libuv/libuv/blob/v1.x/LICENSE
libwebp	https://chromium.googlesource.com/?format=HTML	https://chromium.googlesource.com/chromium/src/third_party/+/refs/heads/main/libwebp/LICENSE
libwebp-base	https://chromium.googlesource.com/?format=HTML	https://chromium.googlesource.com/chromium/src/third_party/+/refs/heads/main/libwebp/LICENSE
libxc	https://www.tddft.org/programs/libxc/	https://gitlab.com/libxc/libxc/-/blob/master/COPYING
libxcb	https://xcb.freedesktop.org	https://github.com/freedesktop/xcb-libxcb/blob/master/COPYING
libxcrypt	https://github.com/besser82/libxcrypt	https://github.com/besser82/libxcrypt?tab=LGPL-2.1-1-ov-file#readme
libxml2	https://git.gnome.org/browse/libxml2/	https://gitlab.gnome.org/GNOME/libxml2/-/blob/master/Copyright
libxmlsec1	https://github.com/lsh123/xmlsec	https://github.com/lsh123/xmlsec/blob/master/README
libxslt	https://gitlab.gnome.org/GNOME/libxslt/-/wikis/home	https://gitlab.gnome.org/GNOME/libxslt/-/blob/master/Copyright

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
libzlib	https://zlib.net	https://zlib.net/zlib_license.html
lightning-utilities	https://github.com/Lightning-AI/utilities	https://github.com/Lightning-AI/utilities?tab=Apache-2.0-1-ov-file#readme
lime	https://github.com/marcotcr/lime	https://github.com/marcotcr/lime/blob/master/LICENSE
linear-operator	https://github.com/cornellius-gp/linear_operator	https://github.com/cornellius-gp/linear_operator?tab=MIT-1-ov-file#readme
lit	http://llvm.org	https://releases.llvm.org/14.0.0/LICENSE.TXT
llvm-openmp	https://github.com/llvm-mirror/openmp	https://github.com/llvm-mirror/openmp/blob/master/LICENSE.txt
llvmlite	http://llvmlite.readthedocs.io	https://github.com/numba/llvmlite/blob/main/LICENSE
loader-utils	https://github.com/webpack/loader-utils	https://github.com/webpack/loader-utils/raw/master/LICENSE
logistro	https://github.com/geopozo/logistro	https://github.com/geopozo/logistro?tab=MIT-1-ov-file#readme
logomaker	https://logomaker.readthedocs.io/en/latest/	https://github.com/jbkinney/logomaker/blob/master/LICENSE
logrus	https://github.com/sirupsen/logrus	https://github.com/sirupsen/logrus/blob/master/LICENSE
logrus-airbrake-hook.v2	https://gopkg.in/gemnasium/logrus-airbrake-hook.v2	https://github.com/gemnasium/logrus-airbrake-hook/blob/v2.1.2/LICENSE
lxml	https://lxml.de	https://lxml.de/index.html#license

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
lz4-c	https://www.lz4.org/	https://github.com/lz4/lz4/blob/dev/LICENSE
lzo	https://www.oberhumer.com/opensource/lzo/	https://www.oberhumer.com/opensource/gpl.html
mamba	https://github.com/mamba-org/mamba	https://github.com/mamba-org/mamba?tab=BSD-3-Clause-1-ov-f readme
markdown	https://github.com/evilstreak/markdown-js	https://www.opensource.org/licenses/mit-license.php
markdown-it-py	https://github.com/executablebooks/markdown-it-py	https://github.com/executablebooks/markdown-it-py/blob/master/LICENSE
markupsafe	https://github.com/pallets/markupsafe/tree/main/src/markupsafe	https://github.com/pallets/markupsafe/tree/main?tab=BSD-3-Clause-1-ov-f
MarkupSafe	https://palletsprojects.com/p/markupsafe/	https://github.com/pallets/markupsafe/blob/main/LICENSE.rst
mashumaro	https://github.com/Fatal1ty/mashumaro	https://github.com/Fatal1ty/mashumaro
matplotlib	https://matplotlib.org	https://matplotlib.org/stable/users/project/license.html
matplotlib-base	https://matplotlib.org	https://matplotlib.org/stable/users/project/license.html
matplotlib-inline	https://github.com/ipython/matplotlib-inline	https://github.com/ipython/matplotlib-inline/blob/master/LICENSE
mda-xdrlib	https://github.com/MDAnalysis/mda-xdrlib	https://github.com/MDAnalysis/mda-xdrlib/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
mda_xdrlib	https://github.com/MDAnalysis/mda-xdrlib	https://github.com/MDAnalysis/mda-xdrlib?tab=License-1-ov-file#readme
mdtraj	https://www.mdtraj.org/	https://www.mdtraj.org/1.9.8.dev0/index.html#license
mdurl	https://github.com/markdown-it/mdurl	https://github.com/markdown-it/mdurl/blob/master/LICENSE
menuinst	https://github.com/conda/menuinst	https://github.com/conda/menuinst/blob/main/LICENSE.txt
mergo	https://github.com/imdario/mergo	https://github.com/imdario/mergo/blob/master/LICENSE
mistune	https://github.com/lepture/mistune	https://github.com/lepture/mistune/blob/master/LICENSE
mkl	https://github.com/rust-math/intel-mkl-src	https://github.com/rust-math/intel-mkl-src/blob/master/License.txt
mkl-fft	https://github.com/IntelPython/mkl_fft	https://github.com/IntelPython/mkl_fft/blob/master/LICENSE.txt
mkl-random	https://github.com/IntelPython/mkl_random	https://github.com/IntelPython/mkl_random/blob/master/LICENSE.txt
mkl-service	https://github.com/IntelPython/mkl-service	https://github.com/IntelPython/mkl-service/blob/master/LICENSE.txt
mkl_fft	http://github.com/IntelPython/mkl_fft	https://github.com/IntelPython/mkl_fft?tab=BSD-3-Clause-1-ov-file#readme

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
mkl_random	http://github.com/IntelPython/mkl_random	https://github.com/IntelPython/mkl_random?tab=BSD-3-Clause-1-ov-f readme
ml-dtypes	https://github.com/jax-ml/ml_dtypes	https://github.com/jax-ml/ml_dtypes/blob/main/LICENSE
ml_dtypes	https://github.com/jax-ml/ml_dtypes	https://github.com/jax-ml/ml_dtypes/blob/main/LICENSE
modelcif	https://github.com/ihmwg/python-modelcif	https://github.com/ihmwg/python-modelcif/blob/main/LICENSE
molecupy	https://molecupy.readthedocs.io/en/latest/	https://github.com/samirelanduk/atomium/blob/master/LICENSE
moment	https://github.com/moment/moment	https://github.com/moment/moment/raw/master/LICENSE
moment-precise-range-plugin	https://github.com/codebox/moment-precise-range	https://github.com/codebox/moment-precise-range raw/master/LICENSE.md
more-itertools	https://github.com/more-itertools/more-itertools	https://github.com/more-itertools/more-itertools/blob/master/LICENSE
mpiplus	https://github.com/choderalab/mpiplus	https://github.com/choderalab/mpiplus/blob/main/LICENSE
mpmath	http://mpmath.org/	https://github.com/mpmath/mpmath/blob/master/LICENSE
mrcfile	https://github.com/ccpem/mrcfile	https://github.com/ccpem/mrcfile/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
msgpack	https://msgpack.org/	https://github.com/msgpack/msgpack-python/blob/main/COPYING
multidict	https://github.com/aio-libs/multidict	https://github.com/aio-libs/multidict/blob/master/LICENSE
multierr	https://go.uber.org/multierr	https://github.com/uber-go/multierr/blob/master/LICENSE.txt
multiprocess	https://github.com/uqfoundation/multiprocess	https://github.com/uqfoundation/multiprocess/blob/master/LICENSE
munkres	https://software.clapper.org/munkres/	https://github.com/bmc/munkres/blob/master/LICENSE.md
myesui uuid	https://github.com/myesui/uuid	https://pkg.go.dev/github.com/myesui/uuid?tab=licenses
namex	https://github.com/bcgov/namex	https://github.com/bcgov/namex/blob/main/LICENSE
narwhals	https://github.com/narwhals-dev/narwhals	https://github.com/narwhals-dev/narwhals?tab=MIT-1-ov-file#readme
nbclassic	http://jupyter.org	https://github.com/jupyter/nbclassic/blob/main/LICENSE
nbclient	https://jupyter.org	https://github.com/jupyter/nbclient/blob/main/LICENSE
nbconvert	https://jupyter.org	https://github.com/jupyter/nbconvert/blob/main/LICENSE
nbformat	http://jupyter.org	https://github.com/jupyter/nbformat/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
ncurses	https://invisible-island.net/ncurses/	https://invisible-island.net/ncurses/ncurses-license.html
ndfes	https://rutgerslbsr.gitlab.io/fe-toolkit/	https://gitlab.com/RutgersLBSR/fe-toolkit/-/blob/master/LICENSE
ndindex	https://quansight-labs.github.io/ndindex/	https://github.com/Quansight-Labs/ndindex?tab=MIT-1-ov-file#readme
nest-asyncio	https://github.com/erdewit/nest_asyncio	https://github.com/erdewit/nest_asyncio/blob/master/LICENSE
nest_asyncio	https://github.com/erdewit/nest_asyncio	https://github.com/erdewit/nest_asyncio?tab=BSD-2-Clause-1-ov-f readme
netcdf-fortran	https://www.unidata.ucar.edu/software/netcdf/	https://www.unidata.ucar.edu/software/netcdf/copyright.html
netCDF4	http://github.com/Unidata/netcdf4-python	https://github.com/Unidata/netcdf4-python/blob/master/LICENSE
nettle	https://git.lysator.liu.se/nettle/nettle	https://git.lysator.liu.se/nettle/nettle/-/blob/master/AUTHORS?ref_type=heads
networkx	https://networkx.org	https://github.com/networkx/networkx/blob/main/LICENSE.txt
nfpm	https://github.com/goreleaser/nfpm	https://github.com/goreleaser/nfpm/blob/master/LICENSE.md
ng-tags-input	https://github.com/mbenford/ngTagsInput	https://github.com/mbenford/ngTagsInput
ng-toast	https://github.com/tameraydin/ngToast	https://github.com/tameraydin/ngToast

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
ngdraggable	https://github.com/fatlinesofcode/ngDraggable	https://github.com/fatlinesofcode/ngDraggable/raw/master/LICENSE.txt
ngVue	https://github.com/ngVue/ngVue	https://github.com/ngVue/ngVue/raw/master/LICENSE
nh3	https://nh3.readthedocs.io/en/latest/	https://github.com/messense/nh3/blob/main/LICENSE
nlohmann_json	https://github.com/nlohmann/json	https://github.com/nlohmann/json?tab=MIT-1-ov-file#readme
nlopt	https://nlopt.readthedocs.io/en/latest/NLopt_Python_Reference/	https://nlopt.readthedocs.io/en/latest/NLopt_License_and_Copyright/
nodejs	https://nodejs.org/en/	https://raw.githubusercontent.com/nodejs/node/master/LICENSE
nomkl	https://github.com/conda-forge/nomkl-feedstock	https://github.com/conda-forge/nomkl-feedstock/blob/main/LICENSE.txt
notebook	http://jupyter.org	https://github.com/jupyter/notebook/blob/main/LICENSE
notebook-shim	https://github.com/jupyter/notebook_shim	https://github.com/jupyter/notebook_shim
notebook_shim	http://jupyter.org	https://github.com/jupyter/notebook_shim/blob/main/LICENSE
numba	https://numba.pydata.org	https://github.com/numba/numba/blob/main/LICENSE
numcpus	https://github.com/tklauser/numcpus	https://github.com/tklauser/numcpus/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
numexpr	https://github.com/pydata/numexpr	https://github.com/pydata/numexpr/blob/master/LICENSE.txt
numpy	https://numpy.org	https://github.com/numpy/numpy/blob/main/LICENSE.txt
numpy-base	https://numpy.org	https://github.com/numpy/numpy/blob/main/LICENSE.txt
numpydoc	https://numpydoc.readthedocs.io/en/latest/index.html	https://github.com/numpy/numpydoc/blob/main/LICENSE.txt
nvidia-cublas-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cublas-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-cupti-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-cupti-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-nvrtc-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-nvrtc-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-runtime-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cuda-runtime-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cudnn-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/deeplearning/cudnn/sla/index.html
nvidia-cudnn-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/deeplearning/cudnn/sla/index.html

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
nvidia-cuEquivariance	https://github.com/NVIDIA/cuEquivariance	https://github.com/NVIDIA/cuEquivariance/blob/main/LICENSES/LICENSE
nvidia-cufft-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cufft-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cufile-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html#
nvidia-curand-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-curand-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cusolver-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cusolver-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cusparse-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cusparse-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-cusparselt-cu12	https://docs.nvidia.com/cuda/cusparselt/index.html	https://docs.nvidia.com/cuda/cusparselt/license.html
nvidia-ml-py	https://pypi.org/project/nvidia-ml-py/	https://pypi.org/project/nvidia-ml-py/
nvidia-nccl-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/hpc-sdk/eula/index.html
nvidia-nccl-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/hpc-sdk/eula/index.html
nvidia-nvjitlink-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
nvidia-nvshmem-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-nvtx-cu11	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
nvidia-nvtx-cu12	https://developer.nvidia.com/cuda-zone	https://docs.nvidia.com/cuda/eula/index.html
Oat++	https://oatpp.io/	https://github.com/oatpp/oatpp/blob/master/LICENSE
oauthlib	https://github.com/oauthlib/oauthlib	https://github.com/oauthlib/oauthlib/blob/master/LICENSE
ocl-icd	https://github.com/OCL-dev/ocl-icd	https://github.com/OCL-dev/ocl-icd/blob/master/COPYING
ocl-icd-system	https://github.com/conda-forge/ocl-icd-system-feedst	https://github.com/conda-forge/ocl-icd-system-feedst blob/main/LICENSE.txt
olefile	https://www.decalage.info/python/olefileio	https://olefile.readthedocs.io/en/latest/License.html
omegaconf	https://github.com/omry/omegaconf	https://github.com/omry/omegaconf/blob/master/LICENSE
OmegaFold	https://github.com/HeliXonProtein/OmegaFold/tree/main	https://github.com/HeliXonProtein/OmegaFold/blob/main/LICENSE
omnicanvas	https://omnicanvas.readthedocs.io/en/latest/	https://github.com/samirelanduk/omnicanvas/blob/master/LICENSE
OpenFF	https://openforcefield.org/	https://github.com/openforcefield/openff-forcefields
openff-amber-ff-ports	https://github.com/openforcefield/openff-amber-ff-ports	https://github.com/openforcefield/openff-amber-ff-ports blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
openff-forcefields	https://openforcefield.org	https://github.com/openforcefield/openff-forcefields/blob/master/LICENSE
openff-fragmenter	https://github.com/openforcefield/openff-fragmenter	https://github.com/openforcefield/openff-fragmenter/blob/main/LICENSE
openff-interchange	https://github.com/openforcefield/openff-interchange	https://github.com/openforcefield/openff-interchange/blob/main/LICENSE
openff-models	https://github.com/openforcefield/openff-models	https://github.com/openforcefield/openff-models/blob/main/LICENSE
openff-nagl	https://docs.openforcefield.org/projects/nagl/en/latest/?badge=latest	https://github.com/openforcefield/openff-nagl/blob/main/LICENSE
openff-nagl-models	https://github.com/openforcefield/openff-nagl-models	https://github.com/openforcefield/openff-nagl-models/blob/main/LICENSE
openff-toolkit	https://openforcefield.org	https://github.com/openforcefield/openff-toolkit/blob/master/LICENSE
openff-toolkit-base	https://openforcefield.org	https://github.com/openforcefield/openff-toolkit/blob/master/LICENSE
openff-units	https://github.com/openforcefield/openff-units	https://github.com/openforcefield/openff-units/blob/main/LICENSE
openff-utilities	https://github.com/openforcefield/openff-utilities	https://github.com/openforcefield/openff-utilities/blob/main/LICENSE
openjpeg	https://github.com/uclouvain/openjpeg	https://github.com/uclouvain/openjpeg/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
openldap	https://www.openldap.org/software/repo.html	https://git.openldap.org/openldap/openldap/-/blob/master/LICENSE
OpenMM	https://openmm.org	https://docs.openmm.org/7.1.0/userguide/library.html#license
openmmtools	https://github.com/choderalab/openmmtools	https://github.com/choderalab/openmmtools/blob/main/LICENSE
openmoltools	https://github.com/choderalab/openmoltools	https://github.com/choderalab/openmoltools/blob/master/LICENSE
openpyxl	https://openpyxl.readthedocs.io/en/stable/	https://foss.heptapod.net/openpyxl/openpyxl/-/blob/branch/3.0/LICENCE.rst
openssl	https://www.openssl.org	https://github.com/openssl/openssl/blob/master/LICENSE.txt
opt-einsum	https://github.com/dgasmith/opt_einsum	https://github.com/dgasmith/opt_einsum/blob/master/LICENSE
opt_einsum	https://github.com/dgasmith/opt_einsum	https://github.com/dgasmith/opt_einsum/blob/main/LICENSE
OptKing	https://github.com/psi-rking/optking	https://github.com/psi-rking/optking/blob/master/LICENSE
optree	https://optree.readthedocs.io/en/latest/	https://github.com/metaopt/optree/blob/main/LICENSE
oscrypto	https://github.com/wbond/oscrypto	https://github.com/wbond/oscrypto/blob/master/LICENSE
overrides	https://github.com/mkorpela/overrides	https://github.com/mkorpela/overrides/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
packaging	https://github.com/pypa/packaging	https://github.com/pypa/packaging/blob/main/LICENSE
packmol	https://leandro.iqm.unicamp.br/m3g/packmol/home.shtml	https://github.com/m3g/packmol/blob/master/LICENSE
packmol_memgen	http://cpclab.uni-duesseldorf.de/	https://gitlab.com/cbjh/memgen/py-memgen/-/blob/main/LICENSE
pandas	https://pandas.pydata.org	https://github.com/pandas-dev/pandas/blob/main/LICENSE
pandocfilters	http://github.com/jgm/pandocfilters	https://github.com/jgm/pandocfilters/blob/master/LICENSE
panedr	https://github.com/MDAnalysis/panedr	https://github.com/MDAnalysis/panedr/blob/master/LICENSE.txt
pango	https://pango.gnome.org	https://gitlab.gnome.org/GNOME/pango/-/blob/main/COPYING
ParmEd	https://parmed.github.io/ParmEd/html/index.html	https://github.com/ParmEd/ParmEd#license
parser	https://github.com/typescript-eslint/typescript-eslint	https://github.com/typescript-eslint/typescript-eslint/raw/master/LICENSE
parso	https://parso.readthedocs.io/en/latest/	https://github.com/davidhalter/parso/blob/master/LICENSE.txt
pathos	https://github.com/uqfoundation/pathos	https://github.com/uqfoundation/pathos/blob/master/LICENSE
patsy	https://patsy.readthedocs.io/en/latest/	https://github.com/pydata/patsy/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
pbkdf2	https://golang.org/x/crypto/pbkdf2	https://cs.opensource.google/go/x/crypto/+/master:LICENSE
pbr	https://docs.openstack.org/pbr/latest/	https://opendev.org/openstack/pbr/src/branch/master/LICENSE
pcmsolver	https://pcmsolver.readthedocs.io/en/stable/	https://github.com/PCMSolver/pcmsolver/blob/master/LICENSE
pcre	https://www.pcre.org	https://www.pcre.org/licence.txt
pcre2	https://www.pcre.org	https://github.com/PCRE2Project/pcre2/blob/master/LICENCE
pdb4amber	https://github.com/Amber-MD/pdb4amber	https://github.com/Amber-MD/pdb4amber/blob/master/LICENSE
pdbfixer	https://github.com/openmm/pdbfixer	https://github.com/openmm/pdbfixer/blob/master/LICENSE
pexpect	https://pexpect.readthedocs.io/	https://github.com/pexpect/pexpect/blob/master/LICENSE
pgconn	https://github.com/jackc/pgconn	https://github.com/jackc/pgconn/blob/master/LICENSE
pgio	https://github.com/jackc/pgio	https://github.com/jackc/pgio/blob/master/LICENSE
pgpassfile	https://github.com/jackc/pgpassfile	https://github.com/jackc/pgpassfile/blob/master/LICENSE
pgproto3	https://github.com/jackc/pgproto3/v2	https://github.com/jackc/pgproto3/blob/master/LICENSE
pgtype	https://github.com/jackc/pgtype	https://github.com/jackc/pgtype/blob/master/LICENSE
pgx	https://github.com/jackc/pgx/v4	https://github.com/jackc/pgx/blob/v4.0.1/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
phonopy	https://github.com/phonopy/phono3py	https://github.com/phonopy/phono3py/blob/develop/LICENSE
pickleshare	https://github.com/pickleshare/pickleshare	https://github.com/pickleshare/pickleshare/blob/master/LICENSE
Pillow	https://python-pillow.org	https://github.com/python-pillow/Pillow/blob/main/LICENSE
Pint	https://pint.readthedocs.io/en/stable/	https://github.com/hgrecco/pint/blob/master/LICENSE
pip	https://pip.pypa.io/	https://github.com/pypa/pip/blob/main/LICENSE.txt
pip-licenses	https://github.com/raimon49/pip-licenses	https://github.com/raimon49/pip-licenses/blob/master/LICENSE
pixman	https://pixman.org	https://cgit.freedesktop.org/pixman/tree/COPYING
pkginfo	https://launchpad.net/pkginfo	https://bazaar.launchpad.net/~tseaver/pkginfo/trunk/view/head:/LICENSE.txt
pkgutil-resolve-name	https://github.com/graingert/pkgutil-resolve-name	https://github.com/graingert/pkgutil-resolve-name?tab=License-1-ov-file#readme
pkgutil_resolve_name	https://github.com/graingert/pkgutil-resolve-name	https://github.com/graingert/pkgutil-resolve-name?tab=License-1-ov-file#readme
platformdirs	https://github.com/platformdirs/platformdirs	https://github.com/platformdirs/platformdirs/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
plotly	https://plotly.com/python/	https://github.com/plotly/plotly.py/blob/master/LICENSE.txt
plotly-orca	https://github.com/plotly/orca	https://github.com/plotly/orca/blob/master/LICENSE
plotly.js	https://github.com/plotly/plotly.js	https://github.com/plotly/plotly.js/raw/master/LICENSE
pluggy	https://pluggy.readthedocs.io/en/stable/index.html	https://github.com/pytest-dev/pluggy/blob/main/LICENSE
pooch	https://github.com/fatiando/pooch	https://github.com/fatiando/pooch/blob/main/LICENSE.txt
portalocker	https://github.com/WoLpH/portalocker	https://github.com/wolph/portalocker/blob/develop/LICENSE
pox	https://github.com/uqfoundation/pox	https://github.com/uqfoundation/pox/blob/master/LICENSE
ppft	https://github.com/uqfoundation/ppft	https://github.com/uqfoundation/ppft/blob/master/LICENSE
pq	https://github.com/lib/pq	https://github.com/lib/pq/blob/master/LICENSE.md
ProDy	http://www.csb.pitt.edu/ProDy	https://github.com/prody/ProDy/blob/master/LICENSE.rst
prometheus-client	https://github.com/prometheus/client_python	https://github.com/prometheus/client_python/blob/master/LICENSE
prometheus_client	https://github.com/prometheus/client_python	https://github.com/prometheus/client_python/blob/master/LICENSE
prompt-toolkit	https://python-prompt-toolkit.readthedocs.io/en/stable/	https://github.com/prompt-toolkit/python-prompt-toolkit/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
prompt_toolkit	https://github.com/prompt-toolkit/python-prompt-toolki	https://github.com/prompt-toolkit/python-prompt-toolki tab=BSD-3-Clause-1-ov-f readme
propcache	https://github.com/aio-libs/propcache	https://github.com/aio-libs/propcache?tab=Apache-2.0-1-ov-file#readme
protobuf	https://google.golang.org/protobuf	https://github.com/protocolbuffers/protobuf-go/blob/master/LICENSE
psi4	https://psicode.org	https://github.com/psi4/psi4/blob/master/COPYING.LESSER
psutil	https://psutil.readthedocs.io/en/latest/	https://github.com/giampaolo/psutil/blob/master/LICENSE
psycopg2	https://psycopg.org/	https://github.com/psycopg/psycopg2/blob/master/LICENSE
PTable	https://github.com/kxxoling/PTable	https://github.com/kxxoling/PTable/blob/master/COPYING
pthread-stubs	https://xcb.freedesktop.org	https://github.com/freedesktop/xcb-pthread-stubs/blob/master/COPYING
ptyprocess	https://ptyprocess.readthedocs.io/en/latest/	https://github.com/pexpect/ptyprocess/blob/master/LICENSE
pure-eval	http://github.com/alexmojaki/pure_eval	http://github.com/alexmojaki/pure_eval/blob/master/LICENSE.txt
pure_eval	http://github.com/alexmojaki/pure_eval	https://github.com/alexmojaki/pure_eval?tab=MIT-1-ov-file#readme

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
py	https://py.readthedocs.io/en/latest/	https://github.com/pytest-dev/py/blob/master/LICENSE
py-cpuinfo	https://github.com/workhorsy/py-cpuinfo	https://github.com/workhorsy/py-cpuinfo/blob/master/LICENSE
pyasn1	https://github.com/etingof/pyasn1	https://github.com/etingof/pyasn1/blob/master/LICENSE.rst
pyasn1-modules	https://github.com/etingof/pyasn1-modules	https://github.com/etingof/pyasn1-modules/blob/master/LICENSE.txt
pyasn1_modules	https://pyasn1.readthedocs.io/en/latest/contents.html	https://github.com/pyasn1/pyasn1-modules?tab=BSD-2-Clause-1-ov-file#readme
pybind11	http://pybind11.readthedocs.io/en/stable/	https://github.com/pybind/pybind11/blob/master/LICENSE
pybind11-abi	https://github.com/pybind/pybind11	https://github.com/pybind/pybind11/blob/master/LICENSE
pybind11_global	http://pybind11.readthedocs.io/en/stable/	https://github.com/pybind/pybind11/blob/master/LICENSE
pycairo	https://pycairo.readthedocs.io/en/latest/index.html	https://github.com/pygobject/pycairo/blob/master/COPYING
pycosat	https://github.com/conda/pycosat	https://github.com/conda/pycosat/blob/main/LICENSE
pycparser	https://github.com/eliben/pycparser	https://github.com/eliben/pycparser/blob/master/LICENSE
pydantic	https://pydantic-docs.helpmanual.io	https://github.com/samuelcolvin/pydantic/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
pydantic-core	https://github.com/pydantic/pydantic-core	https://github.com/pydantic/pydantic-core
pyedr	https://github.com/MDAnalysis/panedr	https://github.com/MDAnalysis/panedr/blob/master/LICENSE.txt
pyEMMA	http://github.com/markovmodel/PyEMMA	https://github.com/markovmodel/PyEMMA/blob/devel/LICENSE.txt
Pygments	https://pygments.org	https://github.com/pygments/pygments/blob/master/LICENSE
pygments	https://pygments.org/	https://github.com/pygments/pygments?tab=BSD-2-Clause-1-ov-file#readme
pygraphviz	https://pygraphviz.github.io	https://github.com/pygraphviz/pygraphviz/blob/main/LICENSE
pygtop	https://pygtop.readthedocs.io/en/latest/	https://github.com/samirelanduk/pygtop/blob/master/LICENSE
pyHanko	https://github.com/MatthiasValvekens/pyHanko	https://github.com/MatthiasValvekens/pyHanko/blob/master/LICENSE
pyhanko-certvalidator	https://github.com/MatthiasValvekens/certvalidator	https://github.com/MatthiasValvekens/certvalidator/blob/master/LICENSE
PyJWT	https://github.com/jpadilla/pyjwt	https://github.com/jpadilla/pyjwt/blob/master/LICENSE
pymbar	https://pymbar.org	https://github.com/choderalab/pymbar/blob/master/LICENSE
pynvml	https://github.com/gpuopenanalytics/pynvml	https://github.com/gpuopenanalytics/pynvml/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
pyOpenSSL	https://pyopenssl.org/	https://github.com/pyca/pyopenssl/blob/main/LICENSE
pyparsing	https://pyparsing-docs.readthedocs.io/en/latest/	https://github.com/pyparsing/pyparsing/blob/master/LICENSE
PyPDF3	https://github.com/sfneal/PyPDF3	https://github.com/sfneal/PyPDF3/blob/master/LICENSE
pyrsistent	http://github.com/tobgu/pyrsistent/	https://github.com/tobgu/pyrsistent/blob/master/LICENSE.mit
pysam	https://github.com/pysam-developers/pysam	https://github.com/pysam-developers/pysam/blob/master/COPYING
PySocks	https://github.com/Anorov/PySocks	https://github.com/Anorov/PySocks/blob/master/LICENSE
pytables	https://www.pytables.org	https://github.com/PyTables/PyTables/blob/master/LICENSE.txt
python	https://www.python.org/	https://docs.python.org/3/license.html
python-bidi	https://github.com/MeirKriheli/python-bidi	https://github.com/MeirKriheli/python-bidi/blob/master/COPYING
python-constraint	https://github.com/python-constraint/python-constraint	https://github.com/python-constraint/python-constraint/blob/master/LICENSE
python-dateutil	https://dateutil.readthedocs.io	https://github.com/dateutil/dateutil/blob/master/LICENSE
python-json-logger	http://github.com/madzak/python-json-logger	https://github.com/madzak/python-json-logger/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
python-pptx	https://github.com/scanny/python-pptx	https://github.com/scanny/python-pptx/blob/master/LICENSE
python-ldap	https://www.python-ldap.org/	https://github.com/python-ldap/python-ldap/blob/python-ldap-3.4.0/LICENCE
python3-saml	https://github.com/onelogin/python3-saml	https://github.com/onelogin/python3-saml/blob/master/LICENSE
python_abi	https://github.com/conda-forge/python_abi-feedstock	https://github.com/conda-forge/python_abi-feedstock/blob/main/LICENSE.txt
pytorch-lightning	https://github.com/Lightning-AI/pytorch-lightning	https://github.com/Lightning-AI/pytorch-lightning/blob/master/LICENSE
pytz	https://pythonhosted.org/pytz	https://git.launchpad.net/pytz/tree/LICENSE.txt
pytz-deprecation-shim	https://github.com/pganssle/pytz-deprecation-shim	https://github.com/pganssle/pytz-deprecation-shim/blob/master/LICENSE
PyWavelets	https://github.com/PyWavelets/pywt	https://github.com/PyWavelets/pywt/blob/master/LICENSE
PyYAML	https://pyyaml.org/	https://github.com/yaml/pyyaml/blob/master/LICENSE
pyyml	No longer available	No longer available
pyzmq	https://pyzmq.readthedocs.io/en/latest/	https://github.com/zeromq/pyzmq/blob/main/COPYING.BSD
qcelemental	https://github.com/MolSSI/QCElemental	https://github.com/MolSSI/QCElemental/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
qcengine	https://github.com/MolSSI/QCEngine	https://github.com/MolSSI/QCEngine/blob/master/LICENSE
qrcode	https://github.com/lincolnloop/python-qrcode	https://github.com/lincolnloop/python-qrcode/blob/main/LICENSE
ramda	https://github.com/ramda/ramda	https://github.com/ramda/ramda/raw/master/LICENSE.txt
rapidjson	https://rapidjson.org/	https://github.com/Tencent/rapidjson/blob/master/license.txt
rdkit	https://www.rdkit.org	https://github.com/rdkit/rdkit/blob/master/license.txt
re2	https://github.com/google/re2	https://github.com/google/re2/blob/main/LICENSE
readme-renderer	https://github.com/pypa/readme_renderer	https://github.com/pypa/readme_renderer/blob/main/LICENSE
redis-py	https://github.com/andymccurdy/redis-py	https://github.com/redis/redis-py/blob/master/LICENSE
referencing	https://github.com/python-jsonschema/referencing	https://github.com/python-jsonschema/referencing
regex	https://github.com/mrabarnett/mrab-regex	https://github.com/mrabarnett/mrab-regex/blob/hg/LICENSE.txt
reportlab	https://www.reportlab.com	https://www.reportlab.com/dev/docs/faq/#1.3
reproc	https://github.com/DaanDeMeyer/reproc	https://github.com/DaanDeMeyer/reproc/blob/main/LICENSE
reproc-cpp	https://github.com/DaanDeMeyer/reproc	https://github.com/DaanDeMeyer/reproc/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
requests	https://requests.readthedocs.io	https://github.com/psf/requests/blob/main/LICENSE
requests-oauthlib	https://github.com/requests/requests-oauthlib	https://github.com/requests/requests-oauthlib/blob/master/LICENSE
requests-toolbelt	https://toolbelt.readthedocs.org	https://github.com/requests/toolbelt/blob/master/LICENSE
resumable	https://github.com/stevvooe/resumable	https://github.com/stevvooe/resumable/blob/master/LICENSE
retrying	https://github.com/rholder/retrying	https://github.com/rholder/retrying/blob/master/LICENSE
rfc3339-validator	https://github.com/naimetti/rfc3339-validator	https://github.com/naimetti/rfc3339-validator/blob/master/LICENSE
rfc3339_validator	https://github.com/naimetti/rfc3339-validator	https://github.com/naimetti/rfc3339-validator?tab=MIT-1-ov-file#readme
rfc3986	https://rfc3986.readthedocs.io/en/latest/	https://github.com/python-hyper/rfc3986/blob/main/LICENSE
rfc3986-validator	https://github.com/naimetti/rfc3986-validator	https://github.com/naimetti/rfc3986-validator/blob/master/LICENSE
rfc3987-syntax	https://github.com/willynilly/rfc3987-syntax	https://github.com/willynilly/rfc3987-syntax?tab=MIT-1-ov-file#readme
rich	https://github.com/Textualize/rich	https://github.com/Textualize/rich/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
rich-argparse	https://github.com/hamdanal/rich-argparse	https://github.com/hamdanal/rich-argparse/blob/main/LICENSE
rlPyCairo	https://pypi.org/project/rlPyCairo/	https://hg.reportlab.com/hg-public/rlPyCairo/file/tip/rlPyCairo/LICENSE.txt
RocksDict	https://github.com/rocksdict/RocksDict	https://github.com/rocksdict/RocksDict?tab=MIT-1-ov-file#readme
roman-numerals	https://github.com/AA-Turner/roman-numerals	https://github.com/AA-Turner/roman-numerals?tab=License-1-ov-file#readme
roman-numerals-py	https://github.com/AA-Turner/roman-numerals	https://github.com/AA-Turner/roman-numerals?tab=License-1-ov-file#readme
rotary-embedding-torch	https://github.com/lucidrains/rotary-embedding-tor	https://github.com/lucidrains/rotary-embedding-tor blob/main/LICENSE
rpds-py	https://github.com/crate-py/rpds	https://github.com/crate-py/rpds
rpmpack	https://github.com/google/rpmpack	https://github.com/google/rpmpack/blob/master/LICENSE
rsa	https://stuvel.eu/rsa	https://github.com/sybrenstuvel/python-rsa/blob/main/LICENSE
ruamel-yaml	https://sourceforge.net/p/ruamel-yaml/code/ci/default/tree/	https://sourceforge.net/p/ruamel-yaml/code/ci/default/tree/LICENSE
ruamel.yaml.clib	https://sourceforge.net/p/ruamel-yaml-clib/code/ci/default/tree/	https://sourceforge.net/p/ruamel-yaml-clib/code/ci/default/tree/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
s3transfer	https://github.com/boto/s3transfer	https://github.com/boto/s3transfer/blob/develop/LICENSE.txt
sacrebleu	https://github.com/mjpost/sacrebleu	https://github.com/mjpost/sacrebleu/blob/master/LICENSE.txt
sapiens	https://github.com/Merck/Sapiens	https://github.com/Merck/Sapiens/blob/main/LICENSE
sasl	https://mellium.im/sasl	https://github.com/mellium/sasl/blob/main/LICENSE
scikit-gstat	https://mmaelicke.github.io/scikit-gstat/	https://github.com/mmaelicke/scikit-gstat/blob/main/LICENSE
scikit-image	https://scikit-image.org	https://scikit-image.org/docs/dev/license.html
scikit-learn	https://scikit-learn.org/stable/	https://github.com/scikit-learn/scikit-learn/blob/main/COPYING
scikit-learn-extra	https://github.com/scikit-learn-contrib/scikit-learn-extra	https://github.com/scikit-learn-contrib/scikit-learn-extra/blob/main/LICENSE
scikit-network	https://github.com/sknetwork-team/scikit-network	https://github.com/sknetwork-team/scikit-network?tab=License-1-ov-file#readme
scipy	https://scipy.org	https://github.com/scipy/scipy/blob/main/LICENSE.txt
seaborn	https://seaborn.pydata.org	https://github.com/mwaskom/seaborn/blob/master/LICENSE
seaborn-base	https://seaborn.pydata.org	https://github.com/mwaskom/seaborn/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
semver	https://github.com/Masterminds/semver/v3	https://github.com/Masterminds/semver/blob/v3.0.3/LICENSE.txt
Send2Trash	https://github.com/arsenetar/send2trash	https://github.com/arsenetar/send2trash/blob/master/LICENSE
sentry-sdk	https://github.com/getsentry/sentry-python	https://github.com/getsentry/sentry-python/blob/master/LICENSE
setproctitle	https://github.com/dvarrazzo/py-setproctitle	https://github.com/dvarrazzo/py-setproctitle/blob/master/LICENSE
setuptools	https://github.com/pypa/setuptools	https://github.com/pypa/setuptools/blob/main/LICENSE
setuptools-scm	https://github.com/pypa/setuptools_scm/	https://github.com/pypa/setuptools_scm/blob/main/LICENSE
sh	https://github.com/amoffat/sh	https://github.com/amoffat/sh/blob/develop/LICENSE.txt
shellingham	https://github.com/sarugaku/shellingham	https://github.com/sarugaku/shellingham/blob/master/LICENSE
simdjson	https://github.com/simdjson/simdjson/tree/master	https://github.com/simdjson/simdjson/blob/master/LICENSE
simint	https://www.bennyp.org/research/simint/	https://github.com/simint-chem/simint-generator/blob/master/LICENSE
simplejson	https://github.com/simplejson/simplejson	https://github.com/simplejson/simplejson/blob/master/LICENSE.txt
six	https://github.com/benjaminp/six	https://github.com/benjaminp/six/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
smirnoff99frosst	https://github.com/openforcefield/smirnoff99frosst	https://github.com/openforcefield/smirnoff99Frosst/blob/master/LICENSE
smmap	https://github.com/gitpython-developers/smmap	https://github.com/gitpython-developers/smmap/blob/master/LICENSE
snappy	https://github.com/google/snappy	https://github.com/google/snappy/blob/main/COPYING
sniffio	https://github.com/python-trio/sniffio	https://github.com/python-trio/sniffio/blob/master/LICENSE.MIT
snowballstemmer	https://github.com/snowballstem/snowball	https://github.com/snowballstem/snowball/blob/master/COPYING
soupsieve	https://github.com/facelessuser/soupsieve	https://github.com/facelessuser/soupsieve/blob/main/LICENSE.md
spdlog	https://github.com/gabime/spdlog	https://github.com/gabime/spdlog?tab=License-1-ov-file#readme
spglib	https://spglib.readthedocs.io/	https://github.com/spglib/spglib/blob/develop/COPYING
sphinx	https://github.com/sphinx-doc/sphinx	https://github.com/sphinx-doc/sphinx/blob/5.x/LICENSE
sphinxcontrib-applehelp	https://github.com/sphinx-doc/sphinxcontrib-applehe	https://github.com/sphinx-doc/sphinxcontrib-applehe blob/master/LICENSE
sphinxcontrib-devhelp	https://github.com/sphinx-doc/sphinxcontrib-devhelp	https://github.com/sphinx-doc/sphinxcontrib-devhelp blob/master/LICENSE
sphinxcontrib-htmlhelp	https://github.com/sphinx-doc/sphinxcontrib-htmlhel	https://github.com/sphinx-doc/sphinxcontrib-htmlhel blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
sphinxcontrib-jsmath	https://github.com/sphinx-doc/sphinxcontrib-jsmath	https://github.com/sphinx-doc/sphinxcontrib-jsmath/blob/master/LICENSE
sphinxcontrib-qthelp	https://github.com/sphinx-doc/sphinxcontrib-qthelp	https://github.com/sphinx-doc/sphinxcontrib-qthelp/blob/master/LICENSE
sphinxcontrib-serializinghtml	https://github.com/sphinx-doc/sphinxcontrib-serializ	https://github.com/sphinx-doc/sphinxcontrib-serializ blob/master/LICENSE
SQLAlchemy	https://www.sqlalchemy.org	https://docs.sqlalchemy.org/en/14/copyright.html
SQLAlchemy	https://www.sqlalchemy.org/	https://github.com/sqlalchemy/sqlalchemy?tab=MIT-1-ov-file#readme
sqlite	https://sqlite.org/index.html	https://sqlite.org/copyright.html
sqlparse	https://github.com/andialbrecht/sqlparse	https://github.com/andialbrecht/sqlparse/blob/master/LICENSE
stack-data	http://github.com/alexmojaki/stack_data	https://github.com/alexmojaki/stack_data/blob/master/LICENSE.txt
stack_data	http://github.com/alexmojaki/stack_data	https://github.com/alexmojaki/stack_data?tab=MIT-1-ov-file#readme
starfile	https://github.com/alisterburt/starfile	https://github.com/teamtomo/starfile/blob/main/LICENSE
statsmodels	https://github.com/statsmodels/statsmodels	https://github.com/statsmodels/statsmodels/blob/main/LICENSE.txt
structlog	https://www.structlog.org/	https://github.com/hynek/structlog/blob/main/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
streamlit	https://github.com/streamlit/streamlit	https://github.com/streamlit/streamlit/blob/develop/LICENSE
svglib	https://github.com/deeplook/svglib	https://github.com/deeplook/svglib/blob/master/LICENSE.txt
sympy	https://sympy.org	https://github.com/sympy/sympy/blob/master/LICENSE
tables	https://www.pytables.org/	https://github.com/PyTables/PyTables/blob/master/LICENSE.txt
tabulate	https://github.com/astanin/python-tabulate	https://github.com/astanin/python-tabulate/blob/master/LICENSE
tbb	https://github.com/oneapi-src/oneTBB	https://github.com/oneapi-src/oneTBB/blob/master/LICENSE.txt
tenacity	https://github.com/jd/tenacity	https://github.com/jd/tenacity/blob/master/LICENSE
tensorboard	https://github.com/tensorflow/tensorboard	https://github.com/tensorflow/tensorboard/blob/master/LICENSE
tensorboard-data-server	https://github.com/tensorflow/tensorboard	https://github.com/tensorflow/tensorboard/blob/master/LICENSE
tensorboard-plugin-wit	https://github.com/pair-code/what-if-tool	https://github.com/PAIR-code/what-if-tool/blob/master/LICENSE
tensorboardX	https://github.com/lanpa/tensorboardX	https://github.com/lanpa/tensorboardX/blob/master/LICENSE
tensorflow	https://github.com/tensorflow/tensorflow	https://github.com/tensorflow/tensorflow/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
tensorflow-estimator	https://github.com/tensorflow/estimator	https://github.com/tensorflow/estimator/blob/master/LICENSE
tensorflow-io-gcs-filesystem	https://github.com/tensorflow/io	https://github.com/tensorflow/io/blob/master/LICENSE
tensorflow-probability	https://github.com/tensorflow/probability	https://github.com/tensorflow/probability/blob/main/LICENSE
termcolor	https://github.com/hugovk/termcolor	https://github.com/hugovk/termcolor/blob/main/COPYING.txt
terminado	https://github.com/jupyter/terminado	https://github.com/jupyter/terminado/blob/main/LICENSE
testpath	https://github.com/jupyter/testpath	https://github.com/jupyter/testpath/blob/master/LICENSE
textangular	https://github.com/fraywing/textAngular	https://github.com/fraywing/textAngular
texttable	https://github.com/foutaise/texttable/	https://github.com/foutaise/texttable/?tab=MIT-1-ov-file#readme
tf_keras	https://github.com/keras-team/keras-io	https://github.com/keras-team/keras-io/blob/master/LICENSE
tf_keras	https://github.com/keras-team/keras-io	https://github.com/keras-team/keras-io/blob/master/LICENSE
threadpoolctl	https://github.com/joblib/threadpoolctl	https://github.com/joblib/threadpoolctl/blob/master/LICENSE
three	https://github.com/mrdoob/three.js	https://github.com/mrdoob/three.js/raw/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
tiffile	https://github.com/cgohlke/tiffile/	https://github.com/cgohlke/tiffile/blob/master/LICENSE
tinycss2	https://github.com/Kozea/tinycss2/	https://github.com/Kozea/tinycss2/blob/master/LICENSE
tinyxml2	https://github.com/leethomason/tinyxml2	https://github.com/leethomason/tinyxml2/blob/master/LICENSE.txt
tk	https://www.tcl.tk/	https://www.tcl.tk/software/tcltk/license.html
toml	https://github.com/toml-lang/toml	https://github.com/toml-lang/toml/blob/main/LICENSE
tomli	https://github.com/hukkin/tomli	https://github.com/hukkin/tomli/blob/master/LICENSE
toolz	https://github.com/pytoolz/toolz	https://github.com/pytoolz/toolz/blob/master/LICENSE.txt
torch	https://pytorch.org/	https://pypi.org/project/torch/#license
torch-geometric	https://pyg.org/	https://github.com/pyg-team/pytorch_geometric?tab=MIT-1-ov-file#readme
torcheval	https://github.com/pytorch/torcheval	https://github.com/pytorch/torcheval?tab=License-1-ov-file#readme
torchmetrics	https://github.com/Lightning-AI/torchmetrics	https://github.com/Lightning-AI/torchmetrics?tab=Apache-2.0-1-ov-file#readme
tornado	https://www.tornadoweb.org	https://github.com/tornadoweb/tornado/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL		
tqdm	https://github.com/tqdm/tqdm	https://github.com/tqdm/tqdm/blob/master/LICENCE		
tracy	https://github.com/gear-genomics/tracy	https://github.com/gear-genomics/tracy/blob/main/LICENSE		
traitlets	https://github.com/ipython/traitlets	https://github.com/ipython/traitlets/blob/main/COPYING.md		
triton	https://github.com/openai/triton/	https://github.com/openai/triton/blob/main/LICENSE		
truststore	https://github.com/sethmlarson/trus	https://github.com/sethmlarson/truststore/blob/main/LICENSE		
truststore	https://github.com/sethmlarson/truststore	https://github.com/sethmlarson/truststore/blob/main/LICENSEtruststore	https://github.com/sethmlarson/truststore	https://github.com/sethmlarson/truststore/blob/main/LICENSE
ts-jest	https://github.com/kulshekhar/ts-jest	https://github.com/kulshekhar/ts-jest/raw/master/LICENSE.md		
ts-loader	https://github.com/TypeStrong/ts-loader	https://github.com/TypeStrong/ts-loader/raw/master/LICENSE		
twine	https://github.com/pypa/twine	https://github.com/pypa/twine/blob/main/LICENSE		
twinj uuid	https://github.com/twinj/uuid	https://github.com/twinj/uuid/blob/master/LICENSE		
types	https://github.com/babel/babel	https://github.com/babel/babel/raw/master/LICENSE		
types-python-dateutil	https://github.com/python/typeshed	https://github.com/python/typeshed?tab=License-1-ov-file#readme		
types-requests	https://pypi.org/project/types-requests/	https://github.com/python/typeshed/blob/main/LICENSE		

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
typescript	https://github.com/Microsoft/TypeScript	https://github.com/Microsoft/TypeScript/raw/master/LICENSE.txt
typing-inspection	https://github.com/pydantic/typing-inspection	https://github.com/pydantic/typing-inspection?tab=MIT-1-ov-file#readme
typing-utils	https://github.com/bojiang/typing_utils	https://github.com/bojiang/typing_utils?tab=Apache-2.0-1-ov-file#readme
typing_extensions	https://github.com/python/typing	https://github.com/python/typing/blob/master/LICENSE
typing_utils	https://github.com/bojiang/typing_utils	https://github.com/bojiang/typing_utils?tab=Apache-2.0-1-ov-file#readme
tzdata	https://github.com/python/tzdata	https://github.com/python/tzdata/blob/master/LICENSE
tzlocal	https://github.com/regebro/tzlocal	https://github.com/regebro/tzlocal/blob/master/LICENSE.txt
umf	https://github.com/oneapi-src/unified-memory-fram	https://github.com/oneapi-src/unified-memory-fram tab=License-1-ov-file#readme
umi-tools	https://github.com/CGATOxford/UMI-tools	https://github.com/CGATOxford/UMI-tools/blob/master/LICENSE
unicodedata2	https://github.com/fonttools/unicodedata2	https://github.com/fonttools/unicodedata2/blob/master/LICENSE
uri-template	https://gitlab.linss.com/open-source/python/uri-template	https://gitlab.linss.com/open-source/python/uri-template/-/blob/main/LICENSE
uritools	https://github.com/tkem/uritools/	https://github.com/tkem/uritools/blob/master/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL		
urllib3	https://urllib3.readthedocs.io/	https://github.com/urllib3/urllib3/blob/main/LICENSE.txt		
vine	https://github.com/celery/vine	https://github.com/celery/vine/blob/master/LICENSE		
vue	https://github.com/vuejs/vue	https://github.com/vuejs/vue/raw/master/LICENSE		
wadler_lindig	https://github.com/patrick-kidger/wadler_lindig	https://github.com/patrick-kidger/wadler_lindig?tab=Apache-2.0-1-ov-file#readme		
wadler_lindig	https://github.com/patrick-kidger/wadler_lindig	https://github.com/patrick-kidger/wadler_lindig?tab=Apache-2.0-1-ov-file#readmewcwidth	https://github.com/jquast/wcwidth	https://github.com/jquast/wcwidth/blob/master/LICENSE
wandb	https://github.com/wandb/wandb	https://github.com/wandb/wandb/blob/main/LICENSE		
webcolors	https://github.com/ubernostrum/webcolors	https://github.com/ubernostrum/webcolors?tab=BSD-3-Clause-1-ov-file#readme		
webencodings	https://github.com/gsnedders/python-webencodings	https://github.com/gsnedders/python-webencodings/blob/master/LICENSE		
websocket-client	https://github.com/websocket-client/websocket-client.git	https://github.com/websocket-client/websocket-client/blob/master/LICENSE		
Werkzeug	https://palletsprojects.com/p/werkzeug/	https://werkzeug.palletsprojects.com/en/2.1.x/license/		
westpa	http://github.com/westpa/westpa	https://github.com/westpa/westpa/blob/westpa2/LICENSE		
wheel	https://github.com/pypa/wheel	https://github.com/pypa/wheel/blob/main/LICENSE.txt		

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
widgetsnbextension	https://github.com/jupyter-widgets/ipywidgets#readme	https://github.com/jupyter-widgets/ipywidgets/blob/main/LICENSE
wrapt	https://github.com/GrahamDumpleton/wrapt	https://github.com/GrahamDumpleton/wrapt/blob/develop/LICENSE
wsutil	https://github.com/yhat/wsutil	https://github.com/yhat/wsutil/blob/master/LICENSE
x/lint	https://golang.org/x/lint	https://github.com/golang/lint/blob/master/LICENSE
x/mod	https://golang.org/x/mod/semver	https://cs.opensource.google/go/x/mod/+/master:LICENSE
x/net	https://golang.org/x/net	https://cs.opensource.google/go/x/net/+/master:LICENSE
x/oauth2	https://golang.org/x/oauth2	https://cs.opensource.google/go/x/oauth2/+/master:LICENSE
x/sys	https://golang.org/x/sys	https://cs.opensource.google/go/x/sys/+/master:LICENSE
x/text	https://golang.org/x/text	https://cs.opensource.google/go/x/text/+/master:LICENSE
x/tools	https://golang.org/x/tools	https://cs.opensource.google/go/x/tools/+/master:LICENSE
x/xerrors	https://golang.org/x/xerrors	https://cs.opensource.google/go/x/xerrors/+/master:LICENSE
xgboost	https://github.com/dmlc/xgboost	https://github.com/dmlc/xgboost/blob/master/LICENSE
xhtml2pdf	http://github.com/xhtml2pdf/xhtml2pdf	https://github.com/xhtml2pdf/xhtml2pdf/blob/master/LICENSE.txt

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
xlrd	https://github.com/python-excel/xlrd	https://github.com/python-excel/xlrd/blob/master/LICENSE
xlsxwriter	https://github.com/jmcnamara/XlsxWriter	https://github.com/jmcnamara/XlsxWriter/blob/main/LICENSE.txt
xmlsec	https://github.com/mehcode/python-xmlsec	https://github.com/mehcode/python-xmlsec/blob/master/LICENSE
xmltodict	https://github.com/martinblech/xmltodict	https://github.com/martinblech/xmltodict/blob/master/LICENSE
xorg-kbproto	https://gitlab.freedesktop.org/xorg/proto/kbproto	https://gitlab.freedesktop.org/xorg/proto/kbproto/-/blob/master/COPYING
xorg-libice	https://gitlab.freedesktop.org/xorg/lib/libice	https://gitlab.freedesktop.org/xorg/lib/libice/-/blob/master/COPYING
xorg-libsm	https://gitlab.freedesktop.org/xorg/lib/libsm	https://gitlab.freedesktop.org/xorg/lib/libsm/-/blob/master/COPYING
xorg-libx11	https://gitlab.freedesktop.org/xorg/lib/libx11	https://gitlab.freedesktop.org/xorg/lib/libx11/-/blob/master/COPYING
xorg-libxau	https://gitlab.freedesktop.org/xorg/lib/libxau	https://gitlab.freedesktop.org/xorg/lib/libxau/-/blob/master/COPYING
xorg-libxdmcp	https://gitlab.freedesktop.org/xorg/lib/libxdmcp	https://gitlab.freedesktop.org/xorg/lib/libxdmcp/-/blob/master/COPYING

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
xorg-libxext	https://gitlab.freedesktop.org/xorg/lib/libxext	https://gitlab.freedesktop.org/xorg/lib/libxext/-/blob/master/COPYING
xorg-libxrender	https://gitlab.freedesktop.org/xorg/lib/libxrender	https://gitlab.freedesktop.org/xorg/lib/libxrender/-/blob/master/COPYING
xorg-libxt	https://gitlab.freedesktop.org/xorg/lib/libxt	https://gitlab.freedesktop.org/xorg/lib/libxt/-/blob/master/COPYING
xorg-renderproto	https://gitlab.freedesktop.org/xorg/proto/renderproto	https://gitlab.freedesktop.org/xorg/proto/renderproto/-/blob/master/COPYING
xorg-xextproto	https://gitlab.freedesktop.org/xorg/proto/xextproto	https://gitlab.freedesktop.org/xorg/proto/xextproto/-/blob/master/COPYING
xorg-xproto	https://gitlab.freedesktop.org/xorg/proto/xproto	https://gitlab.freedesktop.org/xorg/proto/xproto/-/blob/master/COPYING
xxhash	https://github.com/cespare/xxhash/v2	https://github.com/cespare/xxhash/blob/main/LICENSE.txt
xz	https://github.com/ulikunitz/xz	https://github.com/ulikunitz/xz/blob/master/LICENSE
yaml	https://pyyaml.org/	https://github.com/yaml/pyyaml/blob/master/LICENSE
yaml-cpp	https://github.com/jbeder/yaml-cpp	https://github.com/jbeder/yaml-cpp/blob/master/LICENSE
yaml.v2	https://gopkg.in/yaml.v2	https://github.com/go-yaml/yaml/blob/v2.4.0/LICENSE
yaml.v3	https://gopkg.in/yaml.v3	https://github.com/go-yaml/yaml/blob/v3/LICENSE

continues on next page

Table 1 – continued from previous page

Name of Project	Website	License URL
yarl	https://github.com/aio-libs/yarl/	https://github.com/aio-libs/yarl/blob/master/LICENSE
yaspin	https://github.com/pavdmyt/yaspin	https://github.com/pavdmyt/yaspin/blob/master/LICENSE
yfiles	https://www.yworks.com/products/yfiles	commercial license
yml	https://pypi.org/project/yml/	N/A
zap	https://go.uber.org/zap	https://github.com/uber-go/zap/blob/master/LICENSE.txt
zipp	https://github.com/jaraco/zipp	https://github.com/jaraco/zipp/blob/main/LICENSE
zlib	https://zlib.net/	https://zlib.net/zlib_license.html
zstandard	https://github.com/indygreg/python-zstandard	https://github.com/indygreg/python-zstandard/blob/main/LICENSE
zstd	https://facebook.github.io/zstd/	https://github.com/facebook/zstd/blob/dev/LICENSE
click-option-group	https://github.com/click-contrib/click-option-group	https://github.com/click-contrib/click-option-group/blob/master/LICENSE

10.1 GCC

On the Linux platform, The OpenEye toolkits are linked with the GCC libraries using an eligible compilation process under the terms of the GCC RUNTIME LIBRARY EXCEPTION (see below).

The OpenEye toolkits are NOT open source software and are NOT governed by GPL or other open source licenses. Your right to use the toolkits are governed by the OpenEye license which is binding to You.

For certain Linux distributions, GCC library components are included as object files. Your rights to use GCC library components (libstdc++, libgcc, and libgomp) included with these distributions are governed by the GPLv3 license as follows:

The components libstdc++, libgcc, and libgomp are part of GCC.

- GCC is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
- GCC is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the

implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with GCC (see below). If not, see Licenses GNU Project .

Per GPLv3, section 6d, the source for GCC is available for download from the following location: GNU. The GCC library components included here are from version gcc-4.9.3 and are built using the standard instructions at Installing GCC.

10.1.1 GCC RUNTIME LIBRARY EXCEPTION

```
Version 3.1, 31 March 2009

Copyright © 2009 Free Software Foundation, Inc. http://fsf.org/

Everyone is permitted to copy and distribute verbatim copies of this license document,␣
↪but changing it is not allowed.

This GCC Runtime Library Exception ("Exception") is an additional permission under␣
↪section 7 of the GNU General Public License,
version 3 ("GPLv3"). It applies to a given file (the "Runtime Library") that bears a␣
↪notice placed by the copyright holder
of the file stating that the file is governed by GPLv3 along with this Exception.

When you use GCC to compile a program, GCC may combine portions of certain GCC header␣
↪files and runtime libraries with
the compiled program. The purpose of this Exception is to allow compilation of non-GPL␣
↪(including proprietary) programs
to use, in this way, the header files and runtime libraries covered by this Exception.

0. Definitions.

A file is an "Independent Module" if it either requires the Runtime Library for␣
↪execution after a Compilation Process,
or makes use of an interface provided by the Runtime Library, but is not otherwise based␣
↪on the Runtime Library.

"GCC" means a version of the GNU Compiler Collection, with or without modifications,␣
↪governed by version 3 (or a specified
later version) of the GNU General Public License (GPL) with the option of using any␣
↪subsequent versions published by the FSF.

"GPL-compatible Software" is software whose conditions of propagation, modification and␣
↪use would permit combination with
GCC in accord with the license of GCC.

"Target Code" refers to output from any compiler for a real or virtual target processor␣
↪architecture, in executable
form or suitable for input to an assembler, loader, linker and/or execution phase.␣
↪Notwithstanding that, Target Code
does not include data in any format that is used as a compiler intermediate␣
↪representation, or used for producing a
compiler intermediate representation.
```

(continues on next page)

(continued from previous page)

```
The "Compilation Process" transforms code entirely represented in non-intermediate␣
↪languages designed for human-written
code, and/or in Java Virtual Machine byte code, into Target Code. Thus, for example, use␣
↪of source code generators and
preprocessors need not be considered part of the Compilation Process, since the␣
↪Compilation Process can be understood
as starting with the output of the generators or preprocessors.

A Compilation Process is "Eligible" if it is done using GCC, alone or with other GPL-
↪compatible software, or if it is
done without using any work based on GCC. For example, using non-GPL-compatible Software␣
↪to optimize any GCC intermediate
representations would not qualify as an Eligible Compilation Process.

1. Grant of Additional Permission.

You have permission to propagate a work of Target Code formed by combining the Runtime␣
↪Library with Independent Modules,
even if such propagation would otherwise violate the terms of GPLv3, provided that all␣
↪Target Code was generated by
Eligible Compilation Processes. You may then convey such a combination under terms of␣
↪your choice, consistent with the
licensing of the Independent Modules.

2. No Weakening of GCC Copyleft.

The availability of this Exception does not imply any general presumption that third-
↪party software is unaffected by
the copyleft requirements of the license of GCC.
```

See also

- http://www.gnu.org/licenses/gcc-exception.html

10.1.2 GNU GENERAL PUBLIC LICENSE

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

 The GNU General Public License is a free, copyleft license for
 software and other kinds of works.

   The licenses for most software and other practical works are designed
```

(continues on next page)

(continued from previous page)

```
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
```

(continues on next page)

(continued from previous page)

```
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                    TERMS AND CONDITIONS

  0. Definitions.

  "This License" refers to version 3 of the GNU General Public License.

  "Copyright" also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.

  "The Program" refers to any copyrightable work licensed under this
License.  Each licensee is addressed as "you".  "Licensees" and
"recipients" may be individuals or organizations.

  To "modify" a work means to copy from or adapt all or part of the work
in a fashion requiring copyright permission, other than the making of an
exact copy.  The resulting work is called a "modified version" of the
earlier work or a work "based on" the earlier work.

  A "covered work" means either the unmodified Program or a work based
on the Program.

  To "propagate" a work means to do anything with it that, without
permission, would make you directly or secondarily liable for
infringement under applicable copyright law, except executing it on a
computer or modifying a private copy.  Propagation includes copying,
distribution (with or without modification), making available to the
public, and in some countries other activities as well.

  To "convey" a work means any kind of propagation that enables other
parties to make or receive copies.  Mere interaction with a user through
a computer network, with no transfer of a copy, is not conveying.

  An interactive user interface displays "Appropriate Legal Notices"
to the extent that it includes a convenient and prominently visible
feature that (1) displays an appropriate copyright notice, and (2)
tells the user that there is no warranty for the work (except to the
extent that warranties are provided), that licensees may convey the
work under this License, and how to view a copy of this License.  If
the interface presents a list of user commands or options, such as a
menu, a prominent item in the list meets this criterion.

  1. Source Code.

  The "source code" for a work means the preferred form of the work
for making modifications to it.  "Object code" means any non-source
form of a work.
```

(continues on next page)

(continued from previous page)

```
  A "Standard Interface" means an interface that either is an official
standard defined by a recognized standards body, or, in the case of
interfaces specified for a particular programming language, one that
is widely used among developers working in that language.

  The "System Libraries" of an executable work include anything, other
than the work as a whole, that (a) is included in the normal form of
packaging a Major Component, but which is not part of that Major
Component, and (b) serves only to enable use of the work with that
Major Component, or to implement a Standard Interface for which an
implementation is available to the public in source code form.  A
"Major Component", in this context, means a major essential component
(kernel, window system, and so on) of the specific operating system
(if any) on which the executable work runs, or a compiler used to
produce the work, or an object code interpreter used to run it.

  The "Corresponding Source" for a work in object code form means all
the source code needed to generate, install, and (for an executable
work) run the object code and to modify the work, including scripts to
control those activities.  However, it does not include the work's
System Libraries, or general-purpose tools or generally available free
programs which are used unmodified in performing those activities but
which are not part of the work.  For example, Corresponding Source
includes interface definition files associated with source files for
the work, and the source code for shared libraries and dynamically
linked subprograms that the work is specifically designed to require,
such as by intimate data communication or control flow between those
subprograms and other parts of the work.

  The Corresponding Source need not include anything that users
can regenerate automatically from other parts of the Corresponding
Source.

  The Corresponding Source for a work in source code form is that
same work.

  2. Basic Permissions.

  All rights granted under this License are granted for the term of
copyright on the Program, and are irrevocable provided the stated
conditions are met.  This License explicitly affirms your unlimited
permission to run the unmodified Program.  The output from running a
covered work is covered by this License only if the output, given its
content, constitutes a covered work.  This License acknowledges your
rights of fair use or other equivalent, as provided by copyright law.

  You may make, run and propagate covered works that you do not
convey, without conditions so long as your license otherwise remains
in force.  You may convey covered works to others for the sole purpose
of having them make modifications exclusively for you, or provide you
with facilities for running those works, provided that you comply with
the terms of this License in conveying all material for which you do
```

(continues on next page)

(continued from previous page)

```
not control copyright.  Those thus making or running the covered works
for you must do so exclusively on your behalf, under your direction
and control, on terms that prohibit them from making any copies of
your copyrighted material outside their relationship with you.

  Conveying under any other circumstances is permitted solely under
the conditions stated below.  Sublicensing is not allowed; section 10
makes it unnecessary.

  3. Protecting Users' Legal Rights From Anti-Circumvention Law.

  No covered work shall be deemed part of an effective technological
measure under any applicable law fulfilling obligations under article
11 of the WIPO copyright treaty adopted on 20 December 1996, or
similar laws prohibiting or restricting circumvention of such
measures.

  When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention
is effected by exercising rights under this License with respect to
the covered work, and you disclaim any intention to limit operation or
modification of the work as a means of enforcing, against the work's
users, your or third parties' legal rights to forbid circumvention of
technological measures.

  4. Conveying Verbatim Copies.

  You may convey verbatim copies of the Program's source code as you
receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice;
keep intact all notices stating that this License and any
non-permissive terms added in accord with section 7 apply to the code;
keep intact all notices of the absence of any warranty; and give all
recipients a copy of this License along with the Program.


  You may charge any price or no price for each copy that you convey,
and you may offer support or warranty protection for a fee.

  5. Conveying Modified Source Versions.

  You may convey a work based on the Program, or the modifications to
produce it from the Program, in the form of source code under the
terms of section 4, provided that you also meet all of these conditions:

   a) The work must carry prominent notices stating that you modified
   it, and giving a relevant date.

   b) The work must carry prominent notices stating that it is
   released under this License and any conditions added under section
   7.  This requirement modifies the requirement in section 4 to
   "keep intact all notices".
```

(continues on next page)

(continued from previous page)

```
    c) You must license the entire work, as a whole, under this
    License to anyone who comes into possession of a copy.  This
    License will therefore apply, along with any applicable section 7
    additional terms, to the whole of the work, and all its parts,
    regardless of how they are packaged.  This License gives no
    permission to license the work in any other way, but it does not
    invalidate such permission if you have separately received it.

    d) If the work has interactive user interfaces, each must display
    Appropriate Legal Notices; however, if the Program has interactive
    interfaces that do not display Appropriate Legal Notices, your
    work need not make them do so.

  A compilation of a covered work with other separate and independent
works, which are not by their nature extensions of the covered work,
and which are not combined with it such as to form a larger program,
in or on a volume of a storage or distribution medium, is called an
"aggregate" if the compilation and its resulting copyright are not
used to limit the access or legal rights of the compilation's users
beyond what the individual works permit.  Inclusion of a covered work
in an aggregate does not cause this License to apply to the other
parts of the aggregate.

  6. Conveying Non-Source Forms.

  You may convey a covered work in object code form under the terms
of sections 4 and 5, provided that you also convey the
machine-readable Corresponding Source under the terms of this License,
in one of these ways:

  a) Convey the object code in, or embodied in, a physical product
  (including a physical distribution medium), accompanied by the
  Corresponding Source fixed on a durable physical medium
  customarily used for software interchange.

  b) Convey the object code in, or embodied in, a physical product
  (including a physical distribution medium), accompanied by a
  written offer, valid for at least three years and valid for as
  long as you offer spare parts or customer support for that product
  model, to give anyone who possesses the object code either (1) a
  copy of the Corresponding Source for all the software in the
  product that is covered by this License, on a durable physical
  medium customarily used for software interchange, for a price no
  more than your reasonable cost of physically performing this
  conveying of source, or (2) access to copy the
  Corresponding Source from a network server at no charge.

  c) Convey individual copies of the object code with a copy of the
  written offer to provide the Corresponding Source.  This
  alternative is allowed only occasionally and noncommercially, and
  only if you received the object code with such an offer, in accord
```

(continues on next page)

(continued from previous page)

```
   with subsection 6b.

   d) Convey the object code by offering access from a designated
   place (gratis or for a charge), and offer equivalent access to the
   Corresponding Source in the same way through the same place at no
   further charge.  You need not require recipients to copy the
   Corresponding Source along with the object code.  If the place to
   copy the object code is a network server, the Corresponding Source
   may be on a different server (operated by you or a third party)
   that supports equivalent copying facilities, provided you maintain
   clear directions next to the object code saying where to find the
   Corresponding Source.  Regardless of what server hosts the
   Corresponding Source, you remain obligated to ensure that it is
   available for as long as needed to satisfy these requirements.

   e) Convey the object code using peer-to-peer transmission, provided
   you inform other peers where the object code and Corresponding
   Source of the work are being offered to the general public at no
   charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  "Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.

  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
```

(continues on next page)

(continued from previous page)

```
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  "Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
    reasonable ways as different from the original version; or

    d) Limiting the use for publicity purposes of names of licensors or
```

(continues on next page)

(continued from previous page)

```
   authors of the material; or

   e) Declining to grant rights under trademark law for use of some
   trade names, trademarks, or service marks; or

   f) Requiring indemnification of licensors and authors of that
   material by anyone who conveys the material (or modified versions of
   it) with contractual assumptions of liability to the recipient, for
   any liability that these contractual assumptions directly impose on
   those licensors and authors.

  All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10.  If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further
restriction, you may remove that term.  If a license document contains
a further restriction but permits relicensing or conveying under this
License, you may add to a covered work material governed by the terms
of that license document, provided that the further restriction does
not survive such relicensing or conveying.

  If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the
additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

  Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions;
the above requirements apply either way.

  8. Termination.

  You may not propagate or modify a covered work except as expressly
provided under this License.  Any attempt otherwise to propagate or
modify it is void, and will automatically terminate your rights under
this License (including any patent licenses granted under the third
paragraph of section 11).

  However, if you cease all violation of this License, then your
license from a particular copyright holder is reinstated (a)
provisionally, unless and until the copyright holder explicitly and
finally terminates your license, and (b) permanently, if the copyright
holder fails to notify you of the violation by some reasonable means
prior to 60 days after the cessation.

  Moreover, your license from a particular copyright holder is
reinstated permanently if the copyright holder notifies you of the
violation by some reasonable means, this is the first time you have
received notice of violation of this License (for any work) from that
copyright holder, and you cure the violation prior to 30 days after
your receipt of the notice.
```

(continues on next page)

(continued from previous page)

```
  Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under
this License.  If your rights have been terminated and not permanently
reinstated, you do not qualify to receive new licenses for the same
material under section 10.

  9. Acceptance Not Required for Having Copies.

  You are not required to accept this License in order to receive or
run a copy of the Program.  Ancillary propagation of a covered work
occurring solely as a consequence of using peer-to-peer transmission
to receive a copy likewise does not require acceptance.  However,
nothing other than this License grants you permission to propagate or
modify any covered work.  These actions infringe copyright if you do
not accept this License.  Therefore, by modifying or propagating a
covered work, you indicate your acceptance of this License to do so.

  10. Automatic Licensing of Downstream Recipients.

  Each time you convey a covered work, the recipient automatically
receives a license from the original licensors, to run, modify and
propagate that work, subject to this License.  You are not responsible
for enforcing compliance by third parties with this License.

  An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered
work results from an entity transaction, each party to that
transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could
give under the previous paragraph, plus a right to possession of the
Corresponding Source of the work from the predecessor in interest, if
the predecessor has it or can get it with reasonable efforts.

  You may not impose any further restrictions on the exercise of the
rights granted or affirmed under this License.  For example, you may
not impose a license fee, royalty, or other charge for exercise of
rights granted under this License, and you may not initiate litigation
(including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for
sale, or importing the Program or any portion of it.

  11. Patents.

  A "contributor" is a copyright holder who authorizes use under this
License of the Program or a work on which the Program is based.  The
work thus licensed is called the contributor's "contributor version".

  A contributor's "essential patent claims" are all patent claims
owned or controlled by the contributor, whether already acquired or
hereafter acquired, that would be infringed by some manner, permitted
by this License, of making, using, or selling its contributor version,
```

(continues on next page)

```
but do not include claims that would be infringed only as a
consequence of further modification of the contributor version.  For
purposes of this definition, "control" includes the right to grant
patent sublicenses in a manner consistent with the requirements of
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
patent license under the contributor's essential patent claims, to
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
```

(continues on next page)

(continued from previous page)

```
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.

  12. No Surrender of Others' Freedom.

  If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
covered work so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
to collect a royalty for further conveying from those to whom you convey
the Program, the only way you could satisfy both those terms and this
License would be to refrain entirely from conveying the Program.

  13. Use with the GNU Affero General Public License.

  Notwithstanding any other provision of this License, you have
permission to link or combine any covered work with a work licensed
under version 3 of the GNU Affero General Public License into a single
combined work, and to convey the resulting work.  The terms of this
License will continue to apply to the part which is the covered work,
but the special requirements of the GNU Affero General Public License,
section 13, concerning interaction through a network will apply to the
combination as such.

  14. Revised Versions of this License.

  The Free Software Foundation may publish revised and/or new versions of
the GNU General Public License from time to time.  Such new versions will
be similar in spirit to the present version, but may differ in detail to
address new problems or concerns.

  Each version is given a distinguishing version number.  If the
Program specifies that a certain numbered version of the GNU General
Public License "or any later version" applies to it, you have the
option of following the terms and conditions either of that numbered
version or of any later version published by the Free Software
Foundation.  If the Program does not specify a version number of the
GNU General Public License, you may choose any version ever published
by the Free Software Foundation.

  If the Program specifies that a proxy can decide which future
versions of the GNU General Public License can be used, that proxy's
public statement of acceptance of a version permanently authorizes you
to choose that version for the Program.

  Later license versions may give you additional or different
permissions.  However, no additional obligations are imposed on any
```

(continues on next page)

(continued from previous page)

```
author or copyright holder as a result of your choosing to follow a
later version.

  15. Disclaimer of Warranty.

  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT
HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY
OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM
IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF
ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  16. Limitation of Liability.

  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS
THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
```

(continues on next page)

(continued from previous page)

```
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```

See also

- http://www.gnu.org/licenses/gpl.txt

INDEX

A

AddLabel
    OEGrapheme::OEColorGradientDisplayOptions, 96
AddMarkedValue
    OEGrapheme::OEColorGradientDisplayOptions, 97
AddMarkedValues
    OEGrapheme::OEColorGradientDisplayOptions, 98
AddMolecule
    OEGrapheme::OERamachandranPlot, 125
AddRamachandran
    OEGrapheme::OERamachandranPlot, 125

B

bfactor2img.cs
    Example Code, 34

C

ClearBoxRange
    OEGrapheme::OEColorGradientDisplayOptions, 98
ClearLabels
    OEGrapheme::OEColorGradientDisplayOptions, 98
ClearMarkedValues
    OEGrapheme::OEColorGradientDisplayOptions, 99
ClearQuerySurfaceArcFxn
    OEGrapheme::OEColorOverlapDisplayOptions, 115
    OEGrapheme::OEShapeOverlapDisplayOptions, 132
ClearSurfaceArcFxn
    OEGrapheme::OEShapeQueryDisplayOptions, 138
complex2img.cs
    Example Code, 44
Constructors
    OEGrapheme::OE2DActiveSiteDisplay, 59
    OEGrapheme::OE2DActiveSiteDisplayOptions, 61
    OEGrapheme::OE2DActiveSiteLegendDisplayOptions, 66
    OEGrapheme::OE2DPropMap, 67
    OEGrapheme::OEAlignedDepictionFrom3DOptions, 79
    OEGrapheme::OEAlphaRainbowArcFxn, 144
    OEGrapheme::OEAtomGlyphBase, 82
    OEGrapheme::OEAtomGlyphCircle, 82
    OEGrapheme::OEBondGlyphArrow, 84
    OEGrapheme::OEBondGlyphBase, 85
    OEGrapheme::OEBondGlyphCircle, 86
    OEGrapheme::OEBondGlyphCross, 88
    OEGrapheme::OEBondGlyphCurvedArrow, 89
    OEGrapheme::OEBondGlyphScissors, 91
    OEGrapheme::OEBondGlyphStitch, 93
    OEGrapheme::OEBondGlyphZigZag, 94
    OEGrapheme::OEBrickRoadArcFxn, 145
    OEGrapheme::OECastleArcFxn, 147
    OEGrapheme::OECogArcFxn, 148
    OEGrapheme::OEColorForceFieldDisplay, 112
    OEGrapheme::OEColorForceFieldLegendDisplayOptions, 113
    OEGrapheme::OEColorGradientDisplayOptions, 96
    OEGrapheme::OEColorGradientLabel, 110
    OEGrapheme::OEColorOverlapDisplayOptions, 115
    OEGrapheme::OEComplexSurfaceArcFxnBase, 150
    OEGrapheme::OEDefaultArcFxn, 151
    OEGrapheme::OEDefaultBuriedArcFxn, 152
    OEGrapheme::OEDefaultCavityArcFxn, 154
    OEGrapheme::OEDefaultSolventArcFxn, 155
    OEGrapheme::OEDefaultVoidArcFxn, 156
    OEGrapheme::OEDepictionFrom3DOptions, 116
    OEGrapheme::OEEyelashArcFxn, 158
    OEGrapheme::OEFlowerArcFxn, 159
    OEGrapheme::OEHighlightMonomerDisplayOptions, 292
    OEGrapheme::OEMonomerColorDefault, 294

OEGrapheme::OEMonomerColorMonochrome, 295
OEGrapheme::OEMonomerGlyphStyleDefault, 296
OEGrapheme::OEMonomerGraphDisplayOptions, 298
OEGrapheme::OENearestResidue, 119
OEGrapheme::OENecklaceArcFxn, 161
OEGrapheme::OENoOpLigandAligner, 299
OEGrapheme::OEOliveBranchArcFxn, 163
OEGrapheme::OEPearlsArcFxn, 164
OEGrapheme::OEPeptideDisplayOptions, 120
OEGrapheme::OEPlotMarker, 123
OEGrapheme::OERaceTrackArcFxn, 166
OEGrapheme::OERailroadTrackArcFxn, 167
OEGrapheme::OERamachandranPlot, 125
OEGrapheme::OEResidueSVGNoMarkup, 127
OEGrapheme::OEResidueSVGStandardMarkup, 128
OEGrapheme::OESawArcFxn, 169
OEGrapheme::OEShapeOverlapDisplay, 130
OEGrapheme::OEShapeOverlapDisplayOptions, 132
OEGrapheme::OEShapeQueryDisplay, 135
OEGrapheme::OEShapeQueryDisplayOptions, 137
OEGrapheme::OESimpsonArcFxn, 170
OEGrapheme::OEStitchArcFxn, 172
OEGrapheme::OESunArcFxn, 173
OEGrapheme::OESurfaceArc, 141
OEGrapheme::OEWreathArcFxn, 176
CreateComplexCopy
OEGrapheme::OEComplexSurfaceArcFxnBase, 150
CreateCopy
OEGrapheme::OEAlphaRainbowArcFxn, 145
OEGrapheme::OEAtomGlyphBase, 82
OEGrapheme::OEAtomGlyphCircle, 83
OEGrapheme::OEBondGlyphArrow, 85
OEGrapheme::OEBondGlyphBase, 86
OEGrapheme::OEBondGlyphCircle, 87
OEGrapheme::OEBondGlyphCross, 89
OEGrapheme::OEBondGlyphCurvedArrow, 90
OEGrapheme::OEBondGlyphScissors, 92
OEGrapheme::OEBondGlyphStitch, 94
OEGrapheme::OEBondGlyphZigZag, 95
OEGrapheme::OEBrickRoadArcFxn, 146
OEGrapheme::OECastleArcFxn, 148
OEGrapheme::OECogArcFxn, 149
OEGrapheme::OEComplexSurfaceArcFxnBase, 150
OEGrapheme::OEDefaultArcFxn, 152
OEGrapheme::OEDefaultBuriedArcFxn, 153
OEGrapheme::OEDefaultCavityArcFxn, 154
OEGrapheme::OEDefaultSolventArcFxn, 156
OEGrapheme::OEDefaultVoidArcFxn, 157
OEGrapheme::OEEyelashArcFxn, 159
OEGrapheme::OEFlowerArcFxn, 160
OEGrapheme::OELigandAlignerBase, 293
OEGrapheme::OEMonomerColorBase, 294
OEGrapheme::OEMonomerColorDefault, 295
OEGrapheme::OEMonomerColorMonochrome, 295
OEGrapheme::OEMonomerGlyphStyleBase, 296
OEGrapheme::OEMonomerGlyphStyleDefault, 297
OEGrapheme::OENecklaceArcFxn, 162
OEGrapheme::OENoOpLigandAligner, 300
OEGrapheme::OENullArcFxn, 162
OEGrapheme::OEOliveBranchArcFxn, 164
OEGrapheme::OEPearlsArcFxn, 165
OEGrapheme::OERaceTrackArcFxn, 167
OEGrapheme::OERailroadTrackArcFxn, 168
OEGrapheme::OEResidueSVGMarkupBase, 126
OEGrapheme::OEResidueSVGNoMarkup, 127
OEGrapheme::OEResidueSVGStandardMarkup, 128
OEGrapheme::OESawArcFxn, 170
OEGrapheme::OESimpsonArcFxn, 171
OEGrapheme::OEStitchArcFxn, 173
OEGrapheme::OESunArcFxn, 174
OEGrapheme::OESurfaceArcFxnBase, 175
OEGrapheme::OEWreathArcFxn, 177

E

Example Code
bfactor2img.cs, 34
complex2img.cs, 44
shapeoverlap2pdf.cs, 51

G

GetAminoAcidScale
OEGrapheme::OEPeptideDisplayOptions, 121
GetAtomDisplay
OEGrapheme::OESurfaceArc, 142
GetBgnAngle
OEGrapheme::OESurfaceArc, 142
GetBoxRangeMax
OEGrapheme::OEColorGradientDisplayOptions, 99
GetBoxRangeMin
OEGrapheme::OEColorGradientDisplayOptions, 99
GetBoxRangePen
OEGrapheme::OEColorGradientDisplayOptions, 99
GetCenter
OEGrapheme::OESurfaceArc, 142
GetClassName
OEGrapheme::OEResidueSVGMarkupBase, 126

OEGrapheme::OEResidueSVGNoMarkup, 127
OEGrapheme::OEResidueSVGStandardMarkup, 129
GetClearBackground
OEGrapheme::OEColorOverlapDisplayOptions, 115
OEGrapheme::OEShapeOverlapDisplayOptions, 132
OEGrapheme::OEShapeQueryDisplayOptions, 138
GetClearCoords
OEGrapheme::OEAlignedDepictionFrom3DOptions, 80
GetColor
OEGrapheme::OEColorForceFieldDisplay, 112
GetColorAtomStyle
OEGrapheme::OEShapeQueryDisplayOptions, 138
GetColorStopLabelFont
OEGrapheme::OEColorGradientDisplayOptions, 100
GetColorStopLabelFontScale
OEGrapheme::OEColorGradientDisplayOptions, 100
GetColorStopPrecision
OEGrapheme::OEColorGradientDisplayOptions, 100
GetColorStopVisibility
OEGrapheme::OEColorGradientDisplayOptions, 101
GetColorTypes
OEGrapheme::OEColorForceFieldDisplay, 112
GetCoordsToInertialFrame
OEGrapheme::OEDepictionFrom3DOptions, 117
GetDepictOrientation
OEGrapheme::OEShapeQueryDisplayOptions, 138
GetDepth
OEGrapheme::OEComplexSurfaceArcFxnBase, 150
GetDisplayedLigand
OEGrapheme::OE2DActiveSiteDisplay, 60
GetDisplayedResidues
OEGrapheme::OE2DActiveSiteDisplay, 60
GetDisplayPosition
OEGrapheme::OENearestResidue, 119
GetDist
OEGrapheme::OENearestResidue, 119
GetEndAngle
OEGrapheme::OESurfaceArc, 142
GetGroupId
OEGrapheme::OEResidueSVGMarkupBase, 127
OEGrapheme::OEResidueSVGNoMarkup, 127
OEGrapheme::OEResidueSVGStandardMarkup, 129
GetHeight
OEGrapheme::OE2DActiveSiteDisplay, 60
GetHighlightUnspecifiedStereo
OEGrapheme::OEHighlightMonomerDisplayOptions, 293
GetInteractive
OEGrapheme::OEPeptideDisplayOptions, 121
GetInteractiveEffect
OEGrapheme::OEMonomerGraphDisplayOptions, 298
GetLabel
OEGrapheme::OEColorGradientLabel, 111
GetLabels
OEGrapheme::OEColorGradientDisplayOptions, 101
GetLabelStyle
OEGrapheme::OEPeptideDisplayOptions, 121
GetLegendFont
OEGrapheme::OE2DPropMap, 68
GetLegendFontScale
OEGrapheme::OE2DPropMap, 68
GetLegendLocation
OEGrapheme::OE2DPropMap, 68
GetLigandAlignerFunctor
OEGrapheme::OE2DActiveSiteDisplayOptions, 61
GetMarkedValuePen
OEGrapheme::OEColorGradientDisplayOptions, 101
GetMarkedValuePrecision
OEGrapheme::OEColorGradientDisplayOptions, 101
GetMarkedValues
OEGrapheme::OEColorGradientDisplayOptions, 102
GetMaxBondRotations
OEGrapheme::OEAlignedDepictionFrom3DOptions, 80
OEGrapheme::OEDepictionFrom3DOptions, 117
GetMaxValue
OEGrapheme::OE2DPropMap, 68
GetMinValue
OEGrapheme::OE2DPropMap, 69
GetMonomerColorFunctor
OEGrapheme::OEMonomerGraphDisplayOptions, 298
GetMonomerGlyphStyleFunctor
OEGrapheme::OEMonomerGraphDisplayOptions, 298
GetMonomerHighlightColorFunctor
OEGrapheme::OEHighlightMonomerDisplayOptions, 292

GetMonomerLabelColorFunctor
OEGrapheme::OEHighlightMonomerDisplayOptions, 292
OEGrapheme::OEMonomerGraphDisplayOptions, 298
GetMonomerScale
OEGrapheme::OEMonomerGraphDisplayOptions, 298
GetName
OEGrapheme::OEColorForceFieldDisplay, 112
GetNegativeColor
OEGrapheme::OE2DPropMap, 69
GetOptimizeLigandOrientation
OEGrapheme::OE2DActiveSiteDisplayOptions, 62
GetOptions
OEGrapheme::OE2DActiveSiteDisplay, 60
GetOverlapColor
OEGrapheme::OEShapeOverlapDisplayOptions, 133
GetOverlapDisplayStyle
OEGrapheme::OEShapeOverlapDisplayOptions, 133
GetPen
OEGrapheme::OEPlotMarker, 124
GetPositiveColor
OEGrapheme::OE2DPropMap, 69
GetQuerySurfaceArcFxn
OEGrapheme::OEColorOverlapDisplayOptions, 115
OEGrapheme::OEShapeOverlapDisplayOptions, 133
GetRadius
OEGrapheme::OESurfaceArc, 142
GetRadiusRatio
OEGrapheme::OE2DPropMap, 69
GetRenderInteractiveLegend
OEGrapheme::OE2DActiveSiteDisplayOptions, 62
GetResidue
OEGrapheme::OENearestResidue, 119
GetResidueSVGMarkupFunctor
OEGrapheme::OE2DActiveSiteDisplayOptions, 62
GetResolution
OEGrapheme::OE2DPropMap, 70
GetRotateTo2DReference
OEGrapheme::OEDepictionFrom3DOptions, 117
GetScale
OEGrapheme::OEPlotMarker, 124
GetStyle
OEGrapheme::OEPlotMarker, 124
GetSuppressHydrogens
OEGrapheme::OEAlignedDepictionFrom3DOptions, 80
OEGrapheme::OEDepictionFrom3DOptions, 117
GetSurfaceArcFxn
OEGrapheme::OEShapeQueryDisplayOptions, 139
GetSVGMagnifyResidueInHover
OEGrapheme::OE2DActiveSiteDisplayOptions, 62
GetTitle
OEGrapheme::OEShapeOverlapDisplay, 131
OEGrapheme::OEShapeQueryDisplay, 136
GetValue
OEGrapheme::OEColorGradientLabel, 111
GetWidth
OEGrapheme::OE2DActiveSiteDisplay, 60

I

IsValid
OEGrapheme::OE2DActiveSiteDisplay, 60
OEGrapheme::OENearestResidue, 119
OEGrapheme::OEShapeOverlapDisplay, 131
OEGrapheme::OEShapeQueryDisplay, 136

N

NumColorTypes
OEGrapheme::OEColorForceFieldDisplay, 113
NumCols
OEGrapheme::OE2DActiveSiteLegendDisplayOptions, 66
OEGrapheme::OEColorForceFieldLegendDisplayOptions, 113
NumDataPoints
OEGrapheme::OERamachandranPlot, 126
NumRows
OEGrapheme::OE2DActiveSiteLegendDisplayOptions, 66
OEGrapheme::OEColorForceFieldLegendDisplayOptions, 114

O

OEGrapheme::OE2DActiveSiteDisplay, 59
Constructors, 59
GetDisplayedLigand, 60
GetDisplayedResidues, 60
GetHeight, 60
GetOptions, 60
GetWidth, 60
IsValid, 60
OEGrapheme::OE2DActiveSiteDisplayOptions, 60
Constructors, 61
GetLigandAlignerFunctor, 61
GetOptimizeLigandOrientation, 62
GetRenderInteractiveLegend, 62

GetResidueSVGMarkupFunctor, 62
GetSVGMagnifyResidueInHover, 62
SetBackgroundColor, 63
SetDimensions, 63
SetHeight, 63
SetLigandAlignerFunctor, 63
SetOptimizeLigandOrientation, 63
SetRenderInteractiveLegend, 63
SetResidueSVGMarkupFunctor, 64
SetScale, 65
SetSuperAtomStyle, 65
SetSVGMagnifyResidueInHover, 64
SetWidth, 65
OEGrapheme::OE2DActiveSiteLegendDisplayOptions, 65
Constructors, 66
NumCols, 66
NumRows, 66
OEGrapheme::OE2DPropMap, 66
Constructors, 67
GetLegendFont, 68
GetLegendFontScale, 68
GetLegendLocation, 68
GetMaxValue, 68
GetMinValue, 69
GetNegativeColor, 69
GetPositiveColor, 69
GetRadiusRatio, 69
GetResolution, 70
Render, 67
SetLegendFont, 70
SetLegendFontScale, 71
SetLegendLocation, 72
SetMaxValue, 73
SetMinValue, 74
SetNegativeColor, 75
SetPositiveColor, 76
SetRadiusRatio, 77
SetResolution, 78
OEGrapheme::OE2DPropMapSetup, 177
OEGrapheme::OE2DPropMapSetup::All, 177
OEGrapheme::OE2DPropMapSetup::Legend, 177
OEGrapheme::OE2DPropMapSetup::MaxValue, 178
OEGrapheme::OE2DPropMapSetup::MinValue, 178
OEGrapheme::OE2DPropMapSetup::NegativeColor, 179
OEGrapheme::OE2DPropMapSetup::PositiveColor, 179
OEGrapheme::OE2DPropMapSetup::RadiusRatio, 180
OEGrapheme::OE2DPropMapSetup::Resolution, 180
OEGrapheme::OEActiveSiteRenderStyle, 181
OEGrapheme::OEActiveSiteRenderStyle::BFactorMap, 181
OEGrapheme::OEActiveSiteRenderStyle::ContactMap, 182
OEGrapheme::OEActiveSiteRenderStyle::Default, 181
OEGrapheme::OEActiveSiteRenderStyle::InteractionMap, 182
OEGrapheme::OEActiveSiteRenderStyle::UnpairedMap, 184
OEGrapheme::OEAddComplexSurfaceArcs, 207
OEGrapheme::OEAddGlyph, 209
OEGrapheme::OEAddLigandHighlighting, 210
OEGrapheme::OEAddResidueHighlighting, 216
OEGrapheme::OEAlignedDepictionFrom3DOptions, 79
Constructors, 79
GetClearCoords, 80
GetMaxBondRotations, 80
GetSuppressHydrogens, 80
SetClearCoords, 80
SetMaxBondRotations, 81
SetSuppressHydrogens, 81
OEGrapheme::OEAlphaRainbowArcFxn, 143
Constructors, 144
CreateCopy, 145
operator(), 144
OEGrapheme::OEAtomGlyphBase, 81
Constructors, 82
CreateCopy, 82
RenderGlyph, 82
OEGrapheme::OEAtomGlyphCircle, 82
Constructors, 82
CreateCopy, 83
RenderGlyph, 83
OEGrapheme::OEBondGlyphArrow, 83
Constructors, 84
CreateCopy, 85
RenderGlyph, 85
OEGrapheme::OEBondGlyphBase, 85
Constructors, 85
CreateCopy, 86
RenderGlyph, 86
OEGrapheme::OEBondGlyphCircle, 86
Constructors, 86
CreateCopy, 87
RenderGlyph, 87
OEGrapheme::OEBondGlyphCross, 88
Constructors, 88
CreateCopy, 89
RenderGlyph, 89
OEGrapheme::OEBondGlyphCurvedArrow, 89
Constructors, 89
CreateCopy, 90
RenderGlyph, 90
OEGrapheme::OEBondGlyphScissors, 91

Constructors, 91
CreateCopy, 92
RenderGlyph, 92
OEGrapheme::OEBondGlyphStitch, 92
Constructors, 93
CreateCopy, 94
RenderGlyph, 94
OEGrapheme::OEBondGlyphZigZag, 94
Constructors, 94
CreateCopy, 95
RenderGlyph, 95
OEGrapheme::OEBrickRoadArcFxn, 145
Constructors, 145
CreateCopy, 146
operator(), 146
OEGrapheme::OECastleArcFxn, 146
Constructors, 147
CreateCopy, 148
operator(), 147
OEGrapheme::OECircleStyle, 185
OEGrapheme::OECircleStyle::AlphaRainbow, 185
OEGrapheme::OECircleStyle::BrickRoad, 185
OEGrapheme::OECircleStyle::Castle, 186
OEGrapheme::OECircleStyle::Cog, 186
OEGrapheme::OECircleStyle::Default, 186
OEGrapheme::OECircleStyle::Eyelash, 187
OEGrapheme::OECircleStyle::Flower, 187
OEGrapheme::OECircleStyle::GreekKey, 188
OEGrapheme::OECircleStyle::Necklace, 188
OEGrapheme::OECircleStyle::OliveBranch, 188
OEGrapheme::OECircleStyle::Pearls, 189
OEGrapheme::OECircleStyle::RaceTrack, 189
OEGrapheme::OECircleStyle::RailroadTrack, 190
OEGrapheme::OECircleStyle::Saw, 190
OEGrapheme::OECircleStyle::Simpson, 190
OEGrapheme::OECircleStyle::Stitch, 191
OEGrapheme::OECircleStyle::Sun, 191
OEGrapheme::OECircleStyle::Wreath, 192
OEGrapheme::OEClearSurfaceArcFxn, 217
OEGrapheme::OECogArcFxn, 148
Constructors, 148
CreateCopy, 149
operator(), 149
OEGrapheme::OEColorAtomStyle, 192
OEGrapheme::OEColorAtomStyle::Circle, 192
OEGrapheme::OEColorAtomStyle::Default, 192
OEGrapheme::OEColorAtomStyle::Hidden, 192
OEGrapheme::OEColorForceFieldDisplay, 111
Constructors, 112
GetColor, 112
GetColorTypes, 112
GetName, 112
NumColorTypes, 113
SetColor, 113
SetName, 113
OEGrapheme::OEColorForceFieldLegendDisplayOptions, 113
Constructors, 113
NumCols, 113
NumRows, 114
OEGrapheme::OEColorGradientDisplayOptions, 95
AddLabel, 96
AddMarkedValue, 97
AddMarkedValues, 98
ClearBoxRange, 98
ClearLabels, 98
ClearMarkedValues, 99
Constructors, 96
GetBoxRangeMax, 99
GetBoxRangeMin, 99
GetBoxRangePen, 99
GetColorStopLabelFont, 100
GetColorStopLabelFontScale, 100
GetColorStopPrecision, 100
GetColorStopVisibility, 101
GetLabels, 101
GetMarkedValuePen, 101
GetMarkedValuePrecision, 101
GetMarkedValues, 102
SetBoxRange, 102
SetBoxRangePen, 103
SetColorStopLabelFont, 103
SetColorStopLabelFontScale, 104
SetColorStopPrecision, 105
SetColorStopVisibility, 106
SetMarkedValuePen, 108
SetMarkedValuePrecision, 109
SetMarkedValues, 109
OEGrapheme::OEColorGradientLabel, 110
Constructors, 110
GetLabel, 111
GetValue, 111
OEGrapheme::OEColorOverlapDisplayOptions, 114
ClearQuerySurfaceArcFxn, 115
Constructors, 115
GetClearBackground, 115
GetQuerySurfaceArcFxn, 115
SetClearBackground, 116
SetQuerySurfaceArcFxn, 116
OEGrapheme::OEComplexSurfaceArcFxnBase, 149
Constructors, 150
CreateComplexCopy, 150
CreateCopy, 150
GetDepth, 150
operator(), 150
SetDepth, 150
OEGrapheme::OEConfigure2DPropMap, 218
OEGrapheme::OEDefaultArcFxn, 150

Constructors, 151
CreateCopy, 152
operator(), 151
OEGrapheme::OEDefaultBuriedArcFxn, 152
Constructors, 152
CreateCopy, 153
operator(), 152
OEGrapheme::OEDefaultCavityArcFxn, 153
Constructors, 154
CreateCopy, 154
operator(), 154
OEGrapheme::OEDefaultSolventArcFxn, 154
Constructors, 155
CreateCopy, 156
operator(), 155
OEGrapheme::OEDefaultVoidArcFxn, 156
Constructors, 156
CreateCopy, 157
operator(), 157
OEGrapheme::OEDepictionFrom3DOptions, 116
Constructors, 116
GetCoordsToInertialFrame, 117
GetMaxBondRotations, 117
GetRotateTo2DReference, 117
GetSuppressHydrogens, 117
SetCoordsToInertialFrame, 118
SetMaxBondRotations, 118
SetRotateTo2DReference, 118
SetSuppressHydrogens, 119
OEGrapheme::OEDraw2DSurface, 218
OEGrapheme::OEDrawActiveSiteLegend, 220
OEGrapheme::OEDrawAlphaRainbowCircle, 258
OEGrapheme::OEDrawAlphaRainbowSurfaceArc, 259
OEGrapheme::OEDrawBFactorMapLegend, 221
OEGrapheme::OEDrawBrickRoadCircle, 260
OEGrapheme::OEDrawBrickRoadSurfaceArc, 261
OEGrapheme::OEDrawCastleCircle, 262
OEGrapheme::OEDrawCastleSurfaceArc, 263
OEGrapheme::OEDrawCircle, 222
OEGrapheme::OEDrawCogCircle, 264
OEGrapheme::OEDrawCogSurfaceArc, 265
OEGrapheme::OEDrawColorForceFieldLegend, 222
OEGrapheme::OEDrawColorGradient, 223
OEGrapheme::OEDrawDefaultCircle, 266
OEGrapheme::OEDrawDefaultSurfaceArc, 266
OEGrapheme::OEDrawEyelashCircle, 267
OEGrapheme::OEDrawEyelashSurfaceArc, 268
OEGrapheme::OEDrawFlowerCircle, 269
OEGrapheme::OEDrawFlowerSurfaceArc, 270
OEGrapheme::OEDrawGreekKeyCircle, 271
OEGrapheme::OEDrawIridiumData, 300
OEGrapheme::OEDrawMonomer, 301
OEGrapheme::OEDrawMonomerGraph, 302
OEGrapheme::OEDrawNecklaceCircle, 272
OEGrapheme::OEDrawNecklaceSurfaceArc, 272
OEGrapheme::OEDrawOliveBranchCircle, 273
OEGrapheme::OEDrawOliveBranchSurfaceArc, 274
OEGrapheme::OEDrawPearlsCircle, 275
OEGrapheme::OEDrawPearlsSurfaceArc, 276
OEGrapheme::OEDrawPeptide, 225
OEGrapheme::OEDrawRaceTrackCircle, 277
OEGrapheme::OEDrawRaceTrackSurfaceArc, 278
OEGrapheme::OEDrawRailroadTrackCircle, 279
OEGrapheme::OEDrawRailroadTrackSurfaceArc, 280
OEGrapheme::OEDrawResidues, 227
OEGrapheme::OEDrawROCSScores, 229
OEGrapheme::OEDrawSawCircle, 281
OEGrapheme::OEDrawSawSurfaceArc, 281
OEGrapheme::OEDrawSimpsonCircle, 283
OEGrapheme::OEDrawSimpsonSurfaceArc, 283
OEGrapheme::OEDrawStitchCircle, 285
OEGrapheme::OEDrawStitchSurfaceArc, 285
OEGrapheme::OEDrawSunCircle, 286
OEGrapheme::OEDrawSunSurfaceArc, 287
OEGrapheme::OEDrawSurfaceArc, 230
OEGrapheme::OEDrawUnpairedInteractionMapLegend, 230
OEGrapheme::OEDrawWreathCircle, 289
OEGrapheme::OEDrawWreathSurfaceArc, 289
OEGrapheme::OEEyelashArcFxn, 157
Constructors, 158
CreateCopy, 159
operator(), 158
OEGrapheme::OEFlowerArcFxn, 159
Constructors, 159
CreateCopy, 160
operator(), 160
OEGrapheme::OEGet2DSurfaceArcs, 231
OEGrapheme::OEGetMoleculeSurfaceScale, 233
OEGrapheme::OEGetNearbyAtom, 233
OEGrapheme::OEGetNearbyBond, 234
OEGrapheme::OEGetNearbyResidue, 235
OEGrapheme::OEGetNearestAtom, 235
OEGrapheme::OEGetNearestBond, 236
OEGrapheme::OEGetNearestResidue, 236
OEGrapheme::OEGetSurfaceArcFxn, 236
OEGrapheme::OEGraphemeGetArch, 237
OEGrapheme::OEGraphemeGetLicensee, 237
OEGrapheme::OEGraphemeGetPlatform, 237
OEGrapheme::OEGraphemeGetRelease, 237
OEGrapheme::OEGraphemeGetSite, 237
OEGrapheme::OEGraphemeGetVersion, 237
OEGrapheme::OEGraphemeIsLicensed, 237
OEGrapheme::OEHasSurfaceArcFxn, 237
OEGrapheme::OEHighlightMonomerDisplayOptions, 291
Constructors, 292

GetHighlightUnspecifiedStereo, 293
GetMonomerHighlightColorFunctor, 292
GetMonomerLabelColorFunctor, 292
SetHighlightUnspecifiedStereo, 293
SetMonomerHighlightColorFunctor, 292
SetMonomerLabelColorFunctor, 293
OEGrapheme::OEHighlightMonomers, 303
OEGrapheme::OELegendLocation, 193
OEGrapheme::OELegendLocation::Bottom, 193
OEGrapheme::OELegendLocation::Default, 193
OEGrapheme::OELegendLocation::Hidden, 194
OEGrapheme::OELegendLocation::Left, 194
OEGrapheme::OELegendLocation::Right, 195
OEGrapheme::OELegendLocation::Top, 195
OEGrapheme::OELigandAlignerBase, 293
CreateCopy, 293
operator(), 293
OEGrapheme::OEMonomerColorBase, 294
CreateCopy, 294
operator(), 294
OEGrapheme::OEMonomerColorDefault, 294
Constructors, 294
CreateCopy, 295
operator(), 295
OEGrapheme::OEMonomerColorMonochrome, 295
Constructors, 295
CreateCopy, 295
operator(), 295
OEGrapheme::OEMonomerGlyphStyleBase, 296
CreateCopy, 296
operator(), 296
OEGrapheme::OEMonomerGlyphStyleDefault, 296
Constructors, 296
CreateCopy, 297
operator(), 297
OEGrapheme::OEMonomerGraphDisplayOptions, 297
Constructors, 298
GetInteractiveEffect, 298
GetMonomerColorFunctor, 298
GetMonomerGlyphStyleFunctor, 298
GetMonomerLabelColorFunctor, 298
GetMonomerScale, 298
SetInteractiveEffect, 298
SetMonomerColorFunctor, 298
SetMonomerGlyphStyleFunctor, 299
SetMonomerLabelColorFunctor, 299
SetMonomerScale, 299
OEGrapheme::OENearestResidue, 119
Constructors, 119
GetDisplayPosition, 119
GetDist, 119
GetResidue, 119
IsValid, 119
OEGrapheme::OENecklaceArcFxn, 160
Constructors, 161
CreateCopy, 162
operator(), 161
OEGrapheme::OENoOpLigandAligner, 299
Constructors, 299
CreateCopy, 300
operator(), 300
OEGrapheme::OENullArcFxn, 162
CreateCopy, 162
operator(), 162
OEGrapheme::OEOliveBranchArcFxn, 162
Constructors, 163
CreateCopy, 164
operator(), 163
OEGrapheme::OEPatternDirection, 196
OEGrapheme::OEPatternDirection::Default, 196
OEGrapheme::OEPatternDirection::Inside, 196
OEGrapheme::OEPatternDirection::Outside, 196
OEGrapheme::OEPearlsArcFxn, 164
Constructors, 164
CreateCopy, 165
operator(), 165
OEGrapheme::OEPeptideDisplayOptions, 120
Constructors, 120
GetAminoAcidScale, 121
GetInteractive, 121
GetLabelStyle, 121
SetAminoAcidScale, 121
SetInteractive, 122
SetLabelStyle, 122
OEGrapheme::OEPeptideLabelStyle, 197
OEGrapheme::OEPeptideLabelStyle::Default, 197
OEGrapheme::OEPeptideLabelStyle::SingleLetter, 197
OEGrapheme::OEPeptideLabelStyle::ThreeLetters, 197
OEGrapheme::OEPlotMarker, 123
Constructors, 123
GetPen, 124
GetScale, 124
GetStyle, 124
OEGrapheme::OEPlotMarkerStyle, 197
OEGrapheme::OEPlotMarkerStyle::Circle, 198
OEGrapheme::OEPlotMarkerStyle::Default, 197
OEGrapheme::OEPlotMarkerStyle::Square, 198
OEGrapheme::OEPrepareActiveSiteDepiction, 238
OEGrapheme::OEPrepareAlignedDepictionFrom3D, 238
OEGrapheme::OEPrepareDepictionFrom3D, 241
OEGrapheme::OERaceTrackArcFxn, 165
Constructors, 166
CreateCopy, 167
operator(), 166
OEGrapheme::OERailroadTrackArcFxn, 167

Constructors, 167
CreateCopy, 168
operator(), 168
OEGrapheme::OERamachandranPlot, 124
AddMolecule, 125
AddRamachandran, 125
Constructors, 125
NumDataPoints, 126
OEGrapheme::OERenderActiveSite, 242
OEGrapheme::OERenderActiveSiteMaps, 246
OEGrapheme::OERenderBFactorMap, 247
OEGrapheme::OERenderColorOverlap, 250
OEGrapheme::OERenderContactMap, 249
OEGrapheme::OERenderRamachandranPlot, 252
OEGrapheme::OERenderShapeOverlap, 252
OEGrapheme::OERenderShapeQuery, 254
OEGrapheme::OERenderUnpairedInteractionMap, 256
OEGrapheme::OEResidueSVGMarkupBase, 126
CreateCopy, 126
GetClassName, 126
GetGroupId, 127
OEGrapheme::OEResidueSVGNoMarkup, 127
Constructors, 127
CreateCopy, 127
GetClassName, 127
GetGroupId, 127
OEGrapheme::OEResidueSVGStandardMarkup, 128
Constructors, 128
CreateCopy, 128
GetClassName, 129
GetGroupId, 129
OEGrapheme::OESawArcFxn, 168
Constructors, 169
CreateCopy, 170
operator(), 169
OEGrapheme::OESetSurfaceArcFxn, 258
OEGrapheme::OESetup2DPropMap, 258
OEGrapheme::OEShapeOverlapDisplay, 129
Constructors, 130
GetTitle, 131
IsValid, 131
OEGrapheme::OEShapeOverlapDisplayOptions, 131
ClearQuerySurfaceArcFxn, 132
Constructors, 132
GetClearBackground, 132
GetOverlapColor, 133
GetOverlapDisplayStyle, 133
GetQuerySurfaceArcFxn, 133
SetClearBackground, 133
SetOverlapColor, 134
SetOverlapDisplayStyle, 134
SetQuerySurfaceArcFxn, 134
OEGrapheme::OEShapeOverlapDisplayStyle, 198
OEGrapheme::OEShapeOverlapDisplayStyle::Default, 198
OEGrapheme::OEShapeOverlapDisplayStyle::PropertyCloud, 198
OEGrapheme::OEShapeOverlapDisplayStyle::PropertyMap, 198
OEGrapheme::OEShapeQueryDisplay, 134
Constructors, 135
GetTitle, 136
IsValid, 136
OEGrapheme::OEShapeQueryDisplayOptions, 137
ClearSurfaceArcFxn, 138
Constructors, 137
GetClearBackground, 138
GetColorAtomStyle, 138
GetDepictOrientation, 138
GetSurfaceArcFxn, 139
SetClearBackground, 139
SetColorAtomStyle, 139
SetDepictOrientation, 139
SetSurfaceArcFxn, 140
OEGrapheme::OESimpsonArcFxn, 170
Constructors, 170
CreateCopy, 171
operator(), 171
OEGrapheme::OEStitchArcFxn, 171
Constructors, 172
CreateCopy, 173
operator(), 172
OEGrapheme::OESunArcFxn, 173
Constructors, 173
CreateCopy, 174
operator(), 174
OEGrapheme::OESurfaceArc, 141
Constructors, 141
GetAtomDisplay, 142
GetBgnAngle, 142
GetCenter, 142
GetEndAngle, 142
GetRadius, 142
SetAtomDisplay, 142
SetBgnAngle, 143
SetCenter, 143
SetEndAngle, 143
SetRadius, 143
OEGrapheme::OESurfaceArcFxnBase, 174
CreateCopy, 175
operator(), 175
OEGrapheme::OESurfaceArcScale, 199
OEGrapheme::OESurfaceArcScale::Default, 200
OEGrapheme::OESurfaceArcScale::Maximum, 200
OEGrapheme::OESurfaceArcScale::Minimum, 200
OEGrapheme::OESurfaceArcStyle, 200

OEGrapheme::OESurfaceArcStyle::AlphaRainbow, 200
OEGrapheme::OESurfaceArcStyle::BrickRoad, 201
OEGrapheme::OESurfaceArcStyle::Castle, 201
OEGrapheme::OESurfaceArcStyle::Cog, 201
OEGrapheme::OESurfaceArcStyle::Default, 202
OEGrapheme::OESurfaceArcStyle::Eyelash, 202
OEGrapheme::OESurfaceArcStyle::Flower, 203
OEGrapheme::OESurfaceArcStyle::Necklace, 203
OEGrapheme::OESurfaceArcStyle::OliveBranch, 203
OEGrapheme::OESurfaceArcStyle::Pearls, 204
OEGrapheme::OESurfaceArcStyle::RaceTrack, 204
OEGrapheme::OESurfaceArcStyle::RailroadTrack, 205
OEGrapheme::OESurfaceArcStyle::Saw, 205
OEGrapheme::OESurfaceArcStyle::Simpson, 205
OEGrapheme::OESurfaceArcStyle::Stitch, 206
OEGrapheme::OESurfaceArcStyle::Sun, 206
OEGrapheme::OESurfaceArcStyle::Wreath, 207
OEGrapheme::OEWreathArcFxn, 175
    Constructors, 176
    CreateCopy, 177
    operator(), 176
operator()
    OEGrapheme::OEAlphaRainbowArcFxn, 144
    OEGrapheme::OEBrickRoadArcFxn, 146
    OEGrapheme::OECastleArcFxn, 147
    OEGrapheme::OECogArcFxn, 149
    OEGrapheme::OEComplexSurfaceArcFxnBase, 150
    OEGrapheme::OEDefaultArcFxn, 151
    OEGrapheme::OEDefaultBuriedArcFxn, 152
    OEGrapheme::OEDefaultCavityArcFxn, 154
    OEGrapheme::OEDefaultSolventArcFxn, 155
    OEGrapheme::OEDefaultVoidArcFxn, 157
    OEGrapheme::OEEyelashArcFxn, 158
    OEGrapheme::OEFlowerArcFxn, 160
    OEGrapheme::OELigandAlignerBase, 293
    OEGrapheme::OEMonomerColorBase, 294
    OEGrapheme::OEMonomerColorDefault, 295
    OEGrapheme::OEMonomerColorMonochrome, 295
    OEGrapheme::OEMonomerGlyphStyleBase, 296
    OEGrapheme::OEMonomerGlyphStyleDefault, 297
    OEGrapheme::OENecklaceArcFxn, 161
    OEGrapheme::OENoOpLigandAligner, 300
    OEGrapheme::OENullArcFxn, 162
    OEGrapheme::OEOliveBranchArcFxn, 163
    OEGrapheme::OEPearlsArcFxn, 165
    OEGrapheme::OERaceTrackArcFxn, 166
    OEGrapheme::OERailroadTrackArcFxn, 168
    OEGrapheme::OESawArcFxn, 169
    OEGrapheme::OESimpsonArcFxn, 171
    OEGrapheme::OEStitchArcFxn, 172
    OEGrapheme::OESunArcFxn, 174
    OEGrapheme::OESurfaceArcFxnBase, 175
    OEGrapheme::OEWreathArcFxn, 176

R

Render
    OEGrapheme::OE2DPropMap, 67
RenderGlyph
    OEGrapheme::OEAtomGlyphBase, 82
    OEGrapheme::OEAtomGlyphCircle, 83
    OEGrapheme::OEBondGlyphArrow, 85
    OEGrapheme::OEBondGlyphBase, 86
    OEGrapheme::OEBondGlyphCircle, 87
    OEGrapheme::OEBondGlyphCross, 89
    OEGrapheme::OEBondGlyphCurvedArrow, 90
    OEGrapheme::OEBondGlyphScissors, 92
    OEGrapheme::OEBondGlyphStitch, 94
    OEGrapheme::OEBondGlyphZigZag, 95

S

SetAminoAcidScale
    OEGrapheme::OEPeptideDisplayOptions, 121
SetAtomDisplay
    OEGrapheme::OESurfaceArc, 142
SetBackgroundColor
    OEGrapheme::OE2DActiveSiteDisplayOptions, 63
SetBgnAngle
    OEGrapheme::OESurfaceArc, 143
SetBoxRange
    OEGrapheme::OEColorGradientDisplayOptions, 102
SetBoxRangePen
    OEGrapheme::OEColorGradientDisplayOptions, 103
SetCenter
    OEGrapheme::OESurfaceArc, 143
SetClearBackground
    OEGrapheme::OEColorOverlapDisplayOptions, 116
    OEGrapheme::OEShapeOverlapDisplayOptions, 133
    OEGrapheme::OEShapeQueryDisplayOptions, 139
SetClearCoords
    OEGrapheme::OEAlignedDepictionFrom3DOptions, 80
SetColor
    OEGrapheme::OEColorForceFieldDisplay, 113
SetColorAtomStyle
    OEGrapheme::OEShapeQueryDisplayOptions, 139
SetColorStopLabelFont

OEGrapheme::OEColorGradientDisplayOptions, 103
SetColorStopLabelFontScale
OEGrapheme::OEColorGradientDisplayOptions, 104
SetColorStopPrecision
OEGrapheme::OEColorGradientDisplayOptions, 105
SetColorStopVisibility
OEGrapheme::OEColorGradientDisplayOptions, 106
SetCoordsToInertialFrame
OEGrapheme::OEDepictionFrom3DOptions, 118
SetDepictOrientation
OEGrapheme::OEShapeQueryDisplayOptions, 139
SetDepth
OEGrapheme::OEComplexSurfaceArcFxnBase, 150
SetDimensions
OEGrapheme::OE2DActiveSiteDisplayOptions, 63
SetEndAngle
OEGrapheme::OESurfaceArc, 143
SetHeight
OEGrapheme::OE2DActiveSiteDisplayOptions, 63
SetHighlightUnspecifiedStereo
OEGrapheme::OEHighlightMonomerDisplayOptions, 293
SetInteractive
OEGrapheme::OEPeptideDisplayOptions, 122
SetInteractiveEffect
OEGrapheme::OEMonomerGraphDisplayOptions, 298
SetLabelStyle
OEGrapheme::OEPeptideDisplayOptions, 122
SetLegendFont
OEGrapheme::OE2DPropMap, 70
SetLegendFontScale
OEGrapheme::OE2DPropMap, 71
SetLegendLocation
OEGrapheme::OE2DPropMap, 72
SetLigandAlignerFunctor
OEGrapheme::OE2DActiveSiteDisplayOptions, 63
SetMarkedValuePen
OEGrapheme::OEColorGradientDisplayOptions, 108
SetMarkedValuePrecision
OEGrapheme::OEColorGradientDisplayOptions, 109
SetMarkedValues
OEGrapheme::OEColorGradientDisplayOptions, 109
SetMaxBondRotations
OEGrapheme::OEAlignedDepictionFrom3DOptions, 81
OEGrapheme::OEDepictionFrom3DOptions, 118
SetMaxValue
OEGrapheme::OE2DPropMap, 73
SetMinValue
OEGrapheme::OE2DPropMap, 74
SetMonomerColorFunctor
OEGrapheme::OEMonomerGraphDisplayOptions, 298
SetMonomerGlyphStyleFunctor
OEGrapheme::OEMonomerGraphDisplayOptions, 299
SetMonomerHighlightColorFunctor
OEGrapheme::OEHighlightMonomerDisplayOptions, 292
SetMonomerLabelColorFunctor
OEGrapheme::OEHighlightMonomerDisplayOptions, 293
OEGrapheme::OEMonomerGraphDisplayOptions, 299
SetMonomerScale
OEGrapheme::OEMonomerGraphDisplayOptions, 299
SetName
OEGrapheme::OEColorForceFieldDisplay, 113
SetNegativeColor
OEGrapheme::OE2DPropMap, 75
SetOptimizeLigandOrientation
OEGrapheme::OE2DActiveSiteDisplayOptions, 63
SetOverlapColor
OEGrapheme::OEShapeOverlapDisplayOptions, 134
SetOverlapDisplayStyle
OEGrapheme::OEShapeOverlapDisplayOptions, 134
SetPositiveColor
OEGrapheme::OE2DPropMap, 76
SetQuerySurfaceArcFxn
OEGrapheme::OEColorOverlapDisplayOptions, 116
OEGrapheme::OEShapeOverlapDisplayOptions, 134
SetRadius
OEGrapheme::OESurfaceArc, 143
SetRadiusRatio
OEGrapheme::OE2DPropMap, 77
SetRenderInteractiveLegend
OEGrapheme::OE2DActiveSiteDisplayOptions, 63

SetResidueSVGMarkupFunctor
    OEGrapheme::OE2DActiveSiteDisplayOptions, 64
SetResolution
    OEGrapheme::OE2DPropMap, 78
SetRotateTo2DReference
    OEGrapheme::OEDepictionFrom3DOptions, 118
SetScale
    OEGrapheme::OE2DActiveSiteDisplayOptions, 65
SetSuperAtomStyle
    OEGrapheme::OE2DActiveSiteDisplayOptions, 65
SetSuppressHydrogens
    OEGrapheme::OEAlignedDepictionFrom3DOptions, 81
    OEGrapheme::OEDepictionFrom3DOptions, 119
SetSurfaceArcFxn
    OEGrapheme::OEShapeQueryDisplayOptions, 140
SetSVGMagnifyResidueInHover
    OEGrapheme::OE2DActiveSiteDisplayOptions, 64
SetWidth
    OEGrapheme::OE2DActiveSiteDisplayOptions, 65
shapeoverlap2pdf.cs
    Example Code, 51