chem-bla-ics

CDK-JChemPaint #1: rendering molecules

Egon Willighagen

Published April 5, 2010

Citation
Willighagen, E. (n.d.). In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/zp98c-wkb92

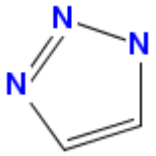
Keywords
Cdk, Jchempaint

Copyright
Copyright © Egon Willighagen 2010. Distributed under the terms of the Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

chem-bla-ics

I reported earlier that the CDK-JChemPaint patch is now a clean add-on from the CDK releases. This means that you download cdk-1.3.4.jar and cdk-jchempaint-8.jar separately, put them in your class path, and get started with, for example, Groovy:

```
$ export CLASSPATH=cdk-1.3.4.jar:cdk-jchempaint-8.jar
$ groovy renderMol.groovy
```



I have tuned to code in this tutorial by Gilleain a bit, resulting in this code:

```
import java.util.List;

import java.awt.*;
import java.awt.image.*;

import javax.imageio.*;

import org.openscience.cdk.*;
import org.openscience.cdk.interfaces.*;
import org.openscience.cdk.layout.*;
import org.openscience.cdk.renderer.*;
import org.openscience.cdk.renderer.font.*;
import org.openscience.cdk.renderer.generators.*;
import org.openscience.cdk.renderer.visitor.*;
import org.openscience.cdk.templates.*;

int WIDTH = 600;
int HEIGHT = 600;

// the draw area and the image should be the same size
Rectangle drawArea = new Rectangle(WIDTH, HEIGHT);
Image image = new BufferedImage(
  WIDTH, HEIGHT, BufferedImage.TYPE_INT_RGB
);

IMolecule triazole = MoleculeFactory.make123Triazole();
StructureDiagramGenerator sdg = new StructureDiagramGenerator();
sdg.setMolecule(triazole);
sdg.generateCoordinates();
triazole = sdg.getMolecule();
```

chem-bla-ics

```
// generators make the image elements
List generators = new ArrayList();
generators.add(new BasicSceneGenerator());
generators.add(new BasicBondGenerator());
generators.add(new BasicAtomGenerator());

// the renderer needs to have a toolkit-specific font manager
AtomContainerRenderer renderer =
  new AtomContainerRenderer(generators, new AWTFontManager());

// the call to 'setup' only needs to be done on the first paint
renderer.setup(triazole, drawArea);

// paint the background
Graphics2D g2 = (Graphics2D)image.getGraphics();
g2.setColor(Color.WHITE);
g2.fillRect(0, 0, WIDTH, HEIGHT);

// the paint method also needs a toolkit-specific renderer
renderer.paint(triazole, new AWTDrawVisitor(g2));

ImageIO.write((RenderedImage)image, "PNG", new File("triazole.png"));
```