Visual Library
What is the Visual Library?

- Generic Visualization Library
- Especially useful for displaying Graphs
- Widgets as Graphic primitives
What is the Visual Library?

- Can be used in any Swing Application
  > Only one dependency – Lookup (util)
- Documentation & Demo Projects:
  > http://graph.netbeans.org/documentation.html
  > http://graph.netbeans.org/examples.html
- Used in NetBeans & many independent applications
How does it work?

- Similar to Swing: Components organized in a tree
- Widgets = Graphics primitive (like JComponent)
  - Scene (Root element)
    - ObjectScene
      - Graph Scene
        - GraphPinScene
  - LayerWidget
  - LabelWidget
  - ImageWidget
  - ...

NetBeans
How does it work?

• Layouts:
  > AbsoluteLayout, FlowLayout, CardLayout, OverlayLayout

• Actions:
  > MoveAction, HoverAction, ZoomAction, PanAction...
  > Every Widget has a WidgetAction.Chain to receive Events and forward them to the matching Action
  > widget.getActions().addAction(ActionFactory.createMoveAction());
Custom Behaviour – Providers

• Use Providers to create Custom Behaviour:
  ActionFactory.createXXXAction(new MyXXXProvider());

• Example:
  private static class MyHoverProvider implements HoverProvider {
    public void widgetHovered(Widget widget) {
      if (widget != null) {
        widget.setForeground(Color.RED);
      }
    }
  }
Back to Data Objects: Editors
Editors

- Multiview vs. “single” view
- Since NB 7.1 better support for multiview editors
  > simple adding of new views to existing editors
  > even not necessary to have sources of the existing editor

whether to generate support for multiview or “single” editor
Adding “single” view own editor

- line in constructor of DataObject
  - registerEditor(“mime-type”, false)
    - second parameter for choosing between single and multiview editors
  - adds support for std text editor
- to add different editor
  - remove the call registerEditor and add
  cookies.add((Node.Cookie) new MyOpenSupport(getPrimaryEntry()));
- add class MyOpenSupport
- create MyTopComponent
  - in the mode editor
  - change its superclass to CloneableTopComponent
class MyOpenSupport extends OpenSupport implements OpenCookie, CloseCookie {

    public MyOpenSupport(MyDataObject.Entry entry) {
        super(entry);
    }

    protected CloneableTopComponent createCloneableTopComponent(){
        MyDataObject dobj = (MyDataObject) entry.getDataObject();
        MyTopComponent tc = new MyTopComponent();
        tc.setDisplayName(dobj.getName());
        return tc;
    }
}
Adding new view for multiview

- Via annotation `@MultiviewElement.Registration`

```java
@MultiViewElement.Registration(displayName = "#LBL_Visualizer",
iconBase = "org/java/vis/icon.gif",
mimeType = "text/x-java",
persistenceType = TopComponent.PERSISTENCE_NEVER,
preferredID = "JavaVisualizer",
position = 3000)
@NbBundle.Messages({
   "LBL_Visualizer=Visualizer"
})
public class JavaVisualizer extends JPanel implements MultiViewElement {
...
```