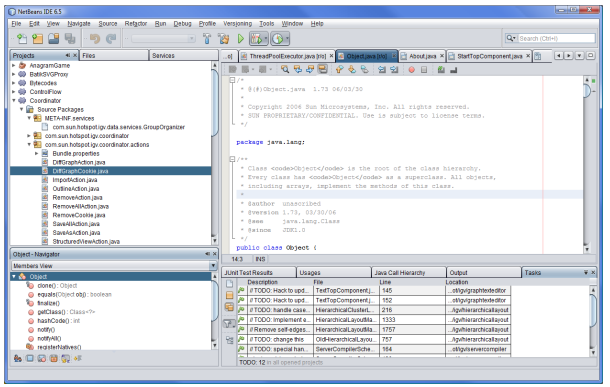
Introduction to the NetBeans Platform



Institute for System Software
Johannes Kepler University Linz, Austria
http://ssw.jku.at

Thomas Wuerthinger
wuerthinger@ssw.jku.at

NetBeans

- IDE (Java, C/C++, PHP, JavaScript, Groovy, Ruby, ...)
- Rich Client Platform
- Only Java (Swing)
- Supported by Sun Microsystems

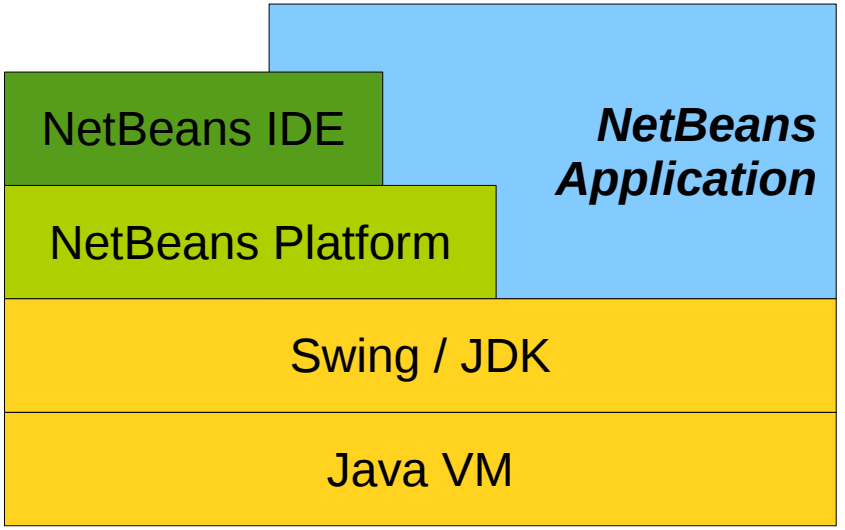
Important Concepts:

- Modules
- Filesystem
- Lookup
- Nodes and Actions

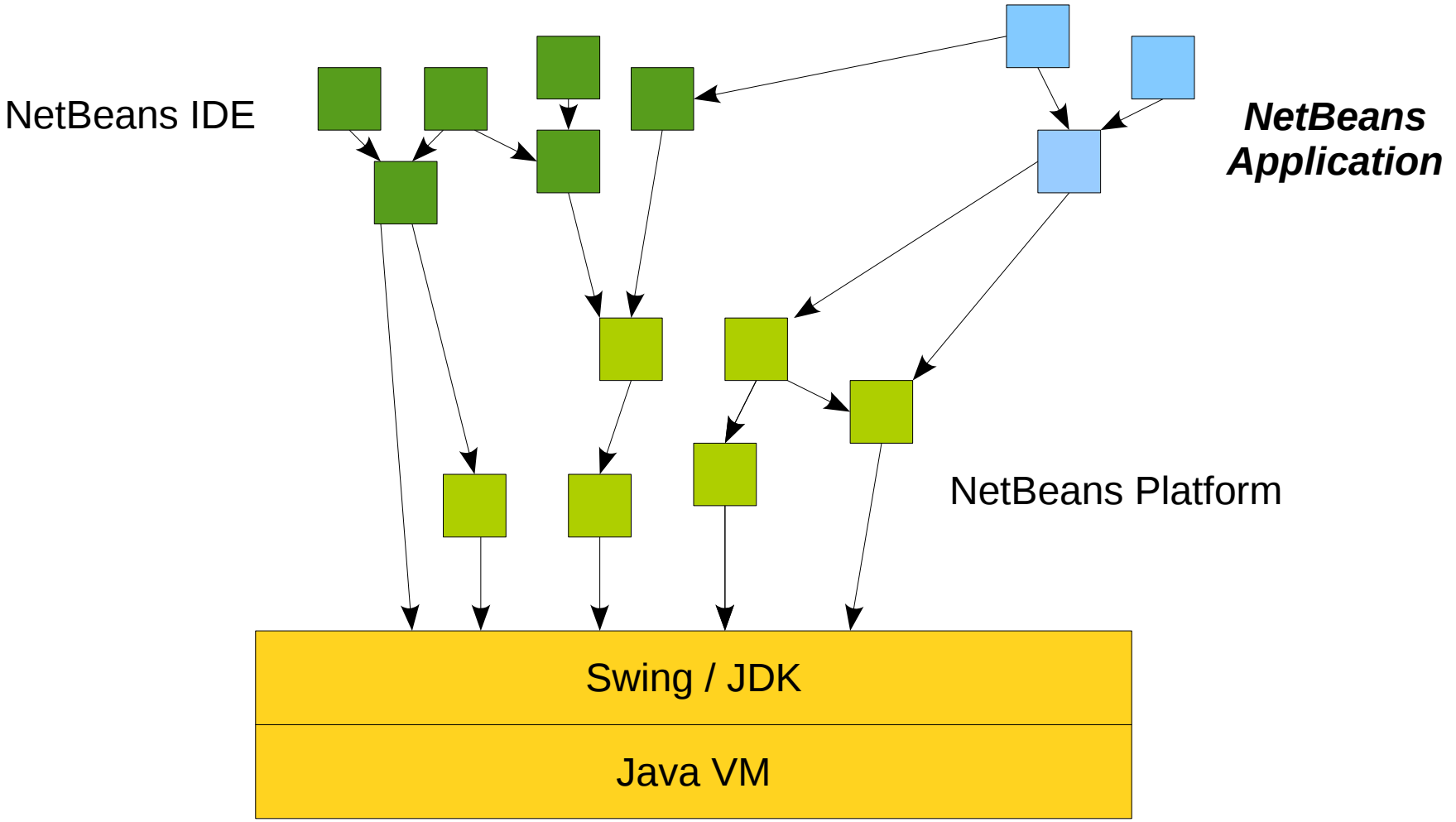
Sources of Information

- NetBeans source code (http://www.netbeans.org/downloads/zip.html)
- API Javadoc (http://bits.netbeans.org/dev/javadoc/index.html)
- Planet NetBeans (http://planetnetbeans.org/de/index.html)
- Numerous NetBeans bloggers (e.g. http://blogs.sun.com/geertjan/)

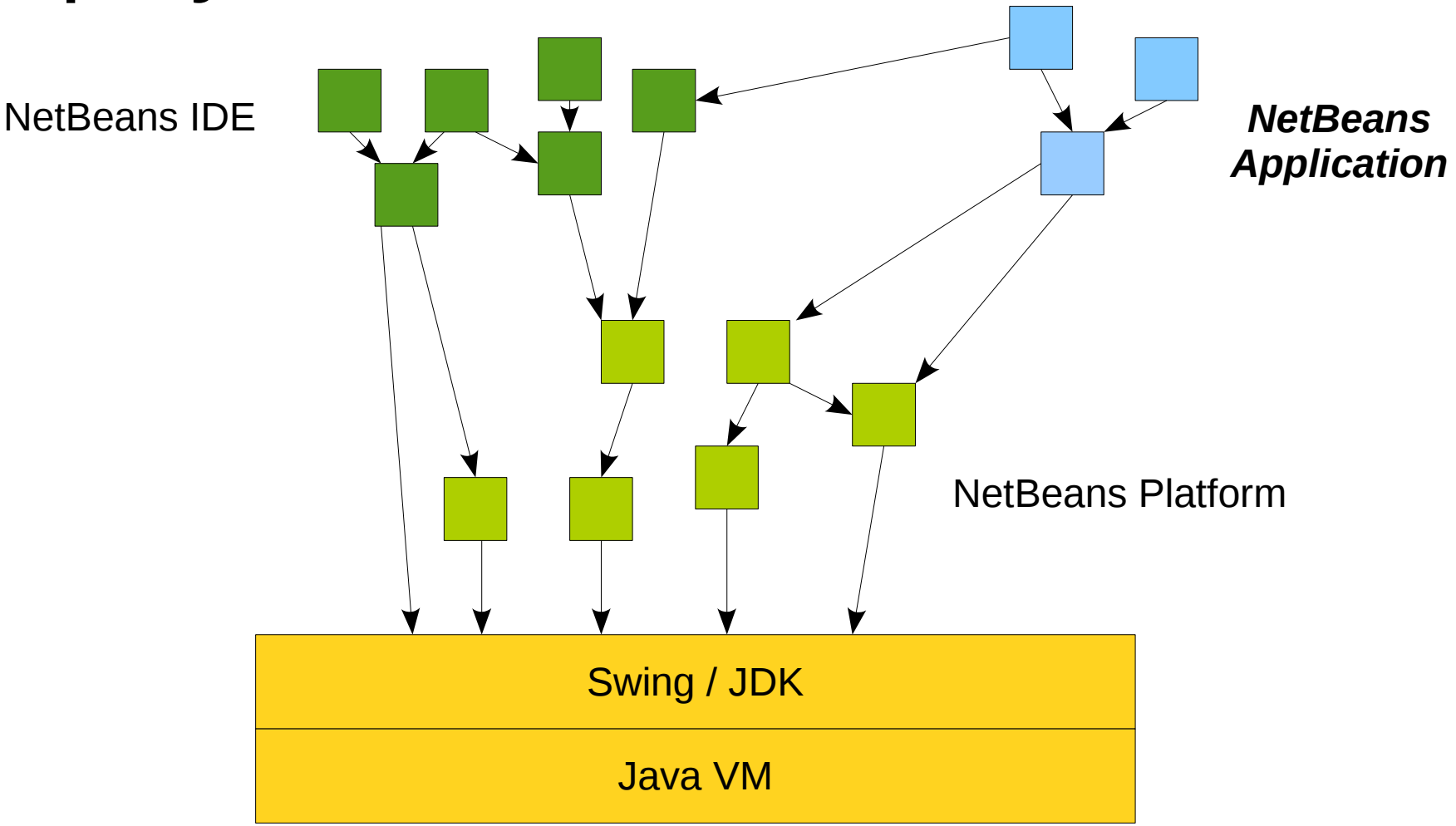
Architecture (1)



Architecture (2)



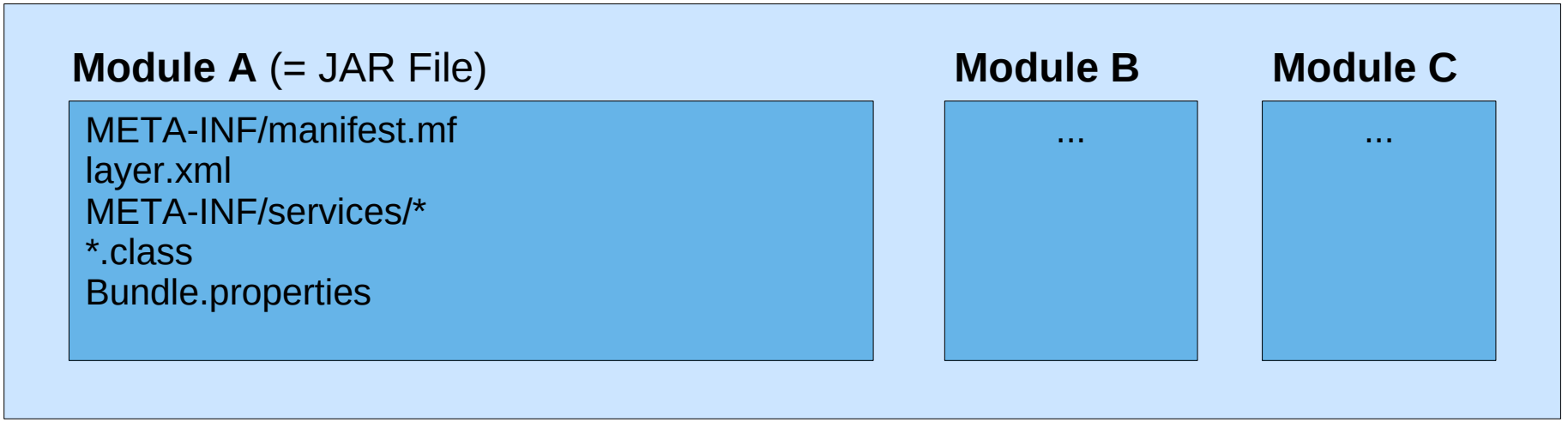
Deployment



- **Standalone:** Deployment including required platform / IDE modules
- **Plugin:** Deployment only with user-defined modules

Module System

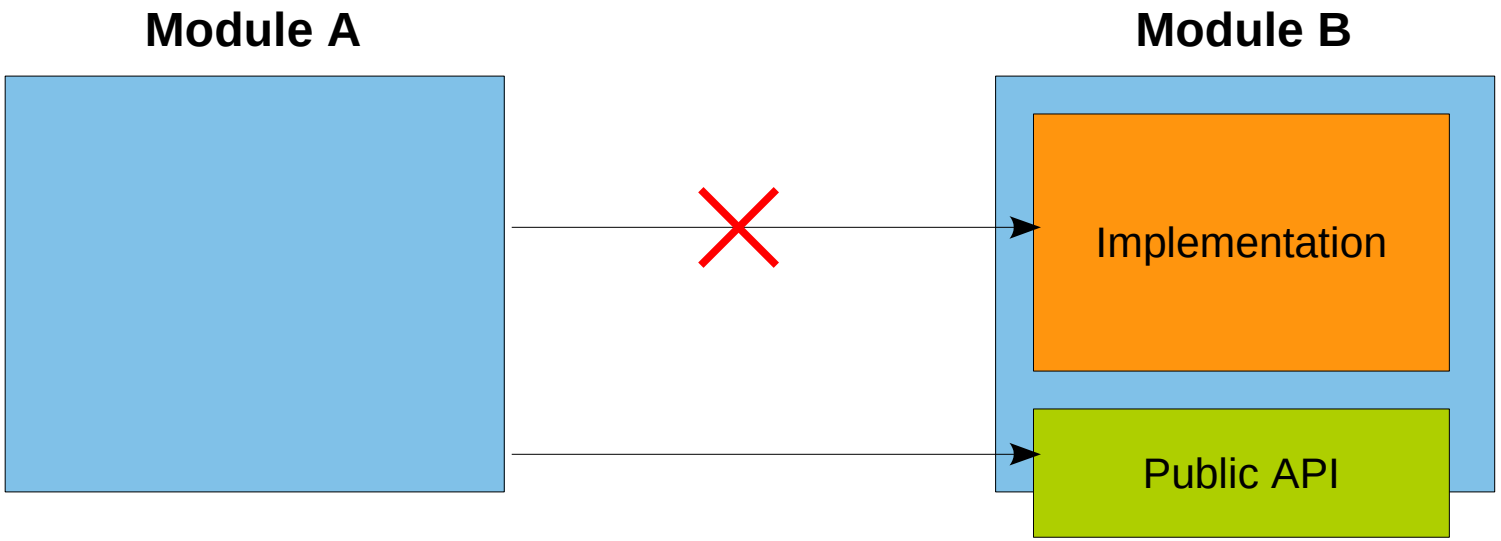
Module Suite (= Deployment Unit)



- Well-defined module dependencies
- Lazy loading / Unloading
- layer.xml for declarative registrations (file system)
- Bundle.properties for internationalization

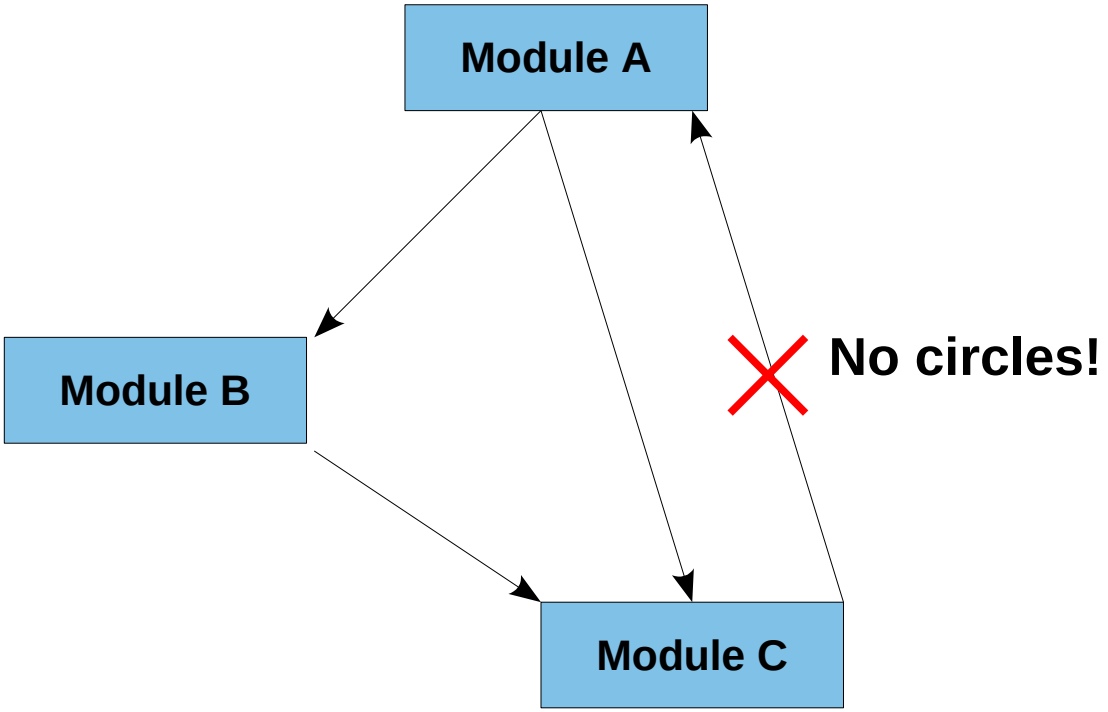
Information Hiding

Public packages are explicitely defined in manifest.mf (project.xml).

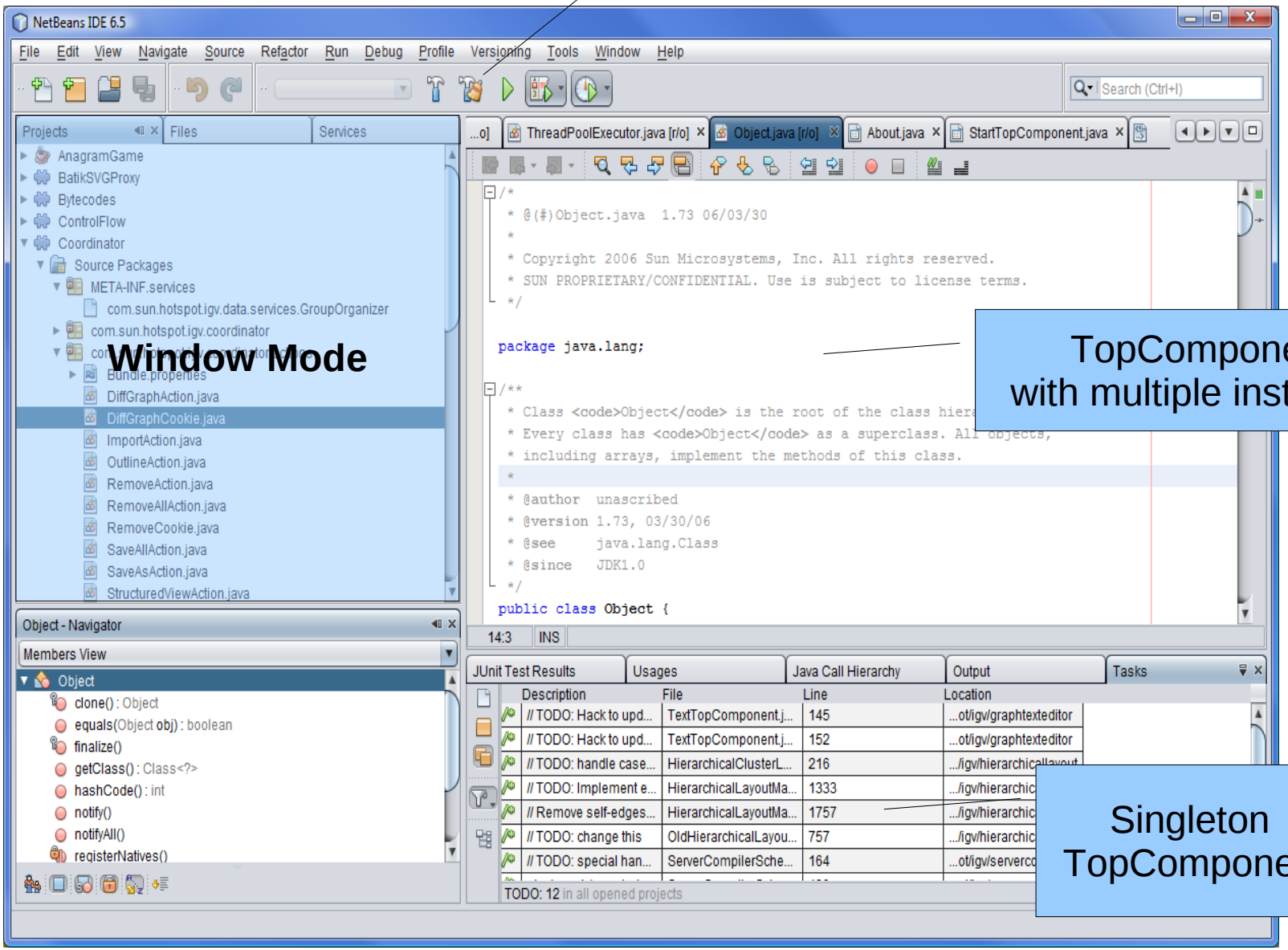


Module Dependencies

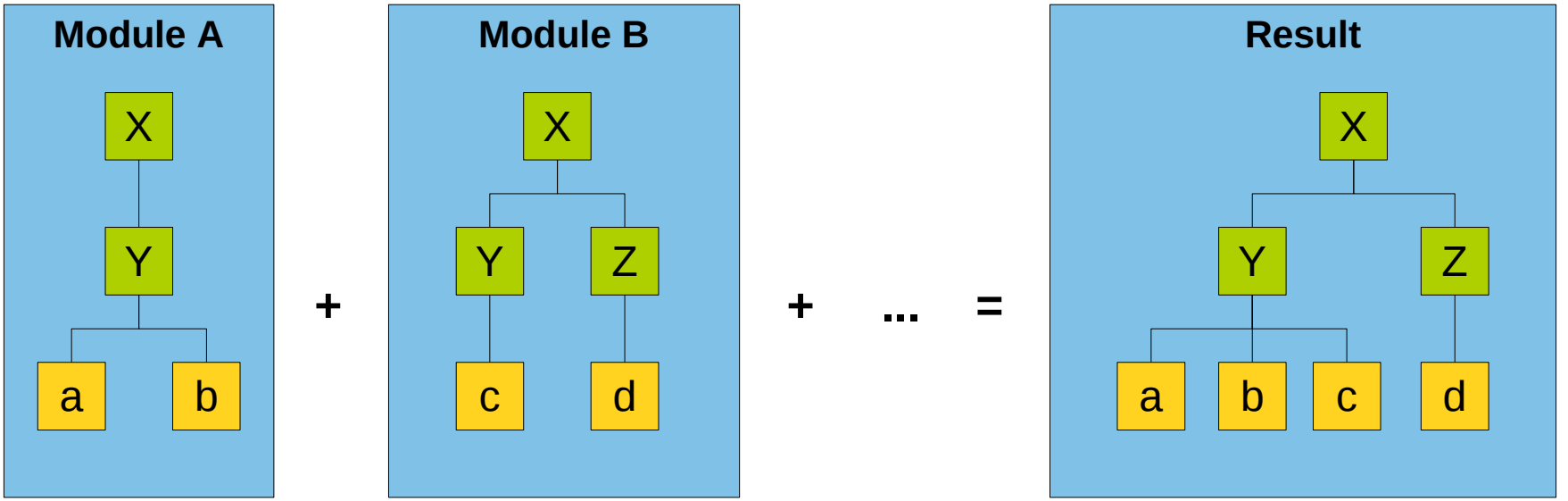
Modules can only use classes of modules they explicitely depend on.



Window System



File System (1)



- Declarative specifications of virtual folders and files
- File system is union of file systems of all current modules

File System (2)

```
<filesystem>

  <folder name="Actions">
    <folder name="Window">
      <file name="TestAction.instance">
        <attr name="displayName" value="Test"/>
      </file>
    </folder>
  </folder>

  <folder name="Menu">
    <folder name="Window">
      <file name="TestAction.shadow">
        <attr name="originalFile"
          stringvalue="Actions/Window/sample-TestAction.instance"/>
      </file>
    </folder>
  </folder>

  <folder name="Windows2">
    <folder name="Components">
      <file name="TopComponent.settings" url="TopComponentSettings.xml"/>
    </folder>
  </folder>

</filesystem>
```

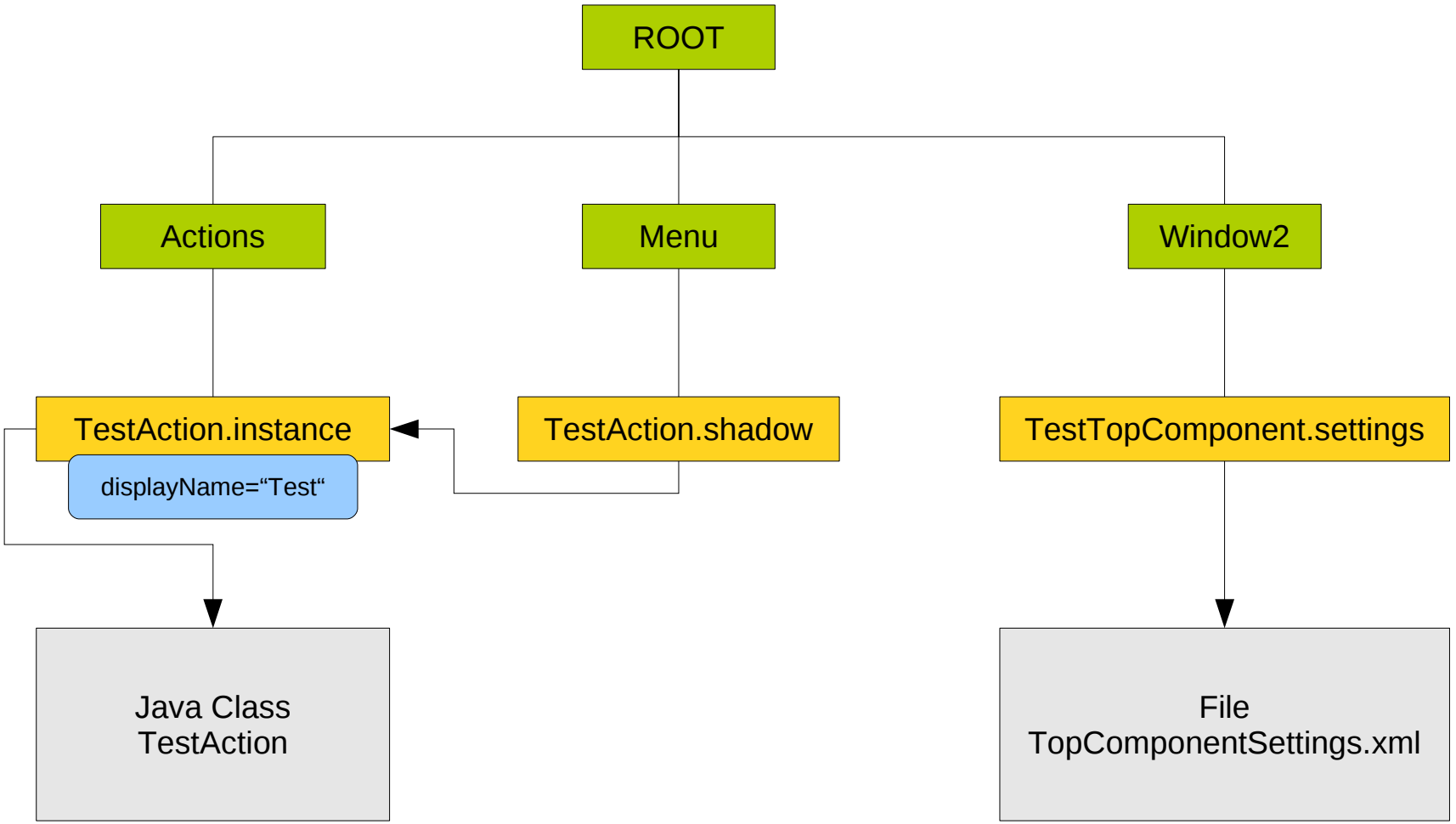
Reference to Java class

Reference to other file

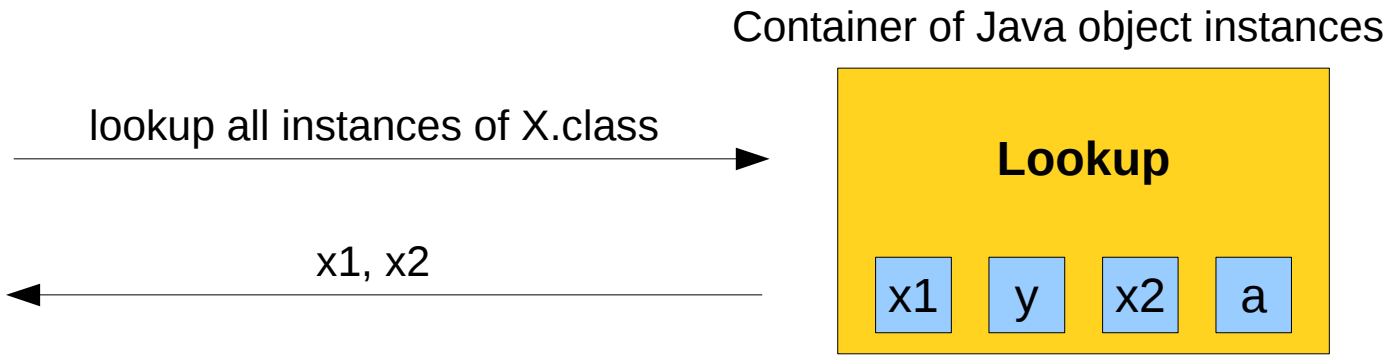
Reference to physical file

Use .instance_hidden to hide existing entries

File System (3)

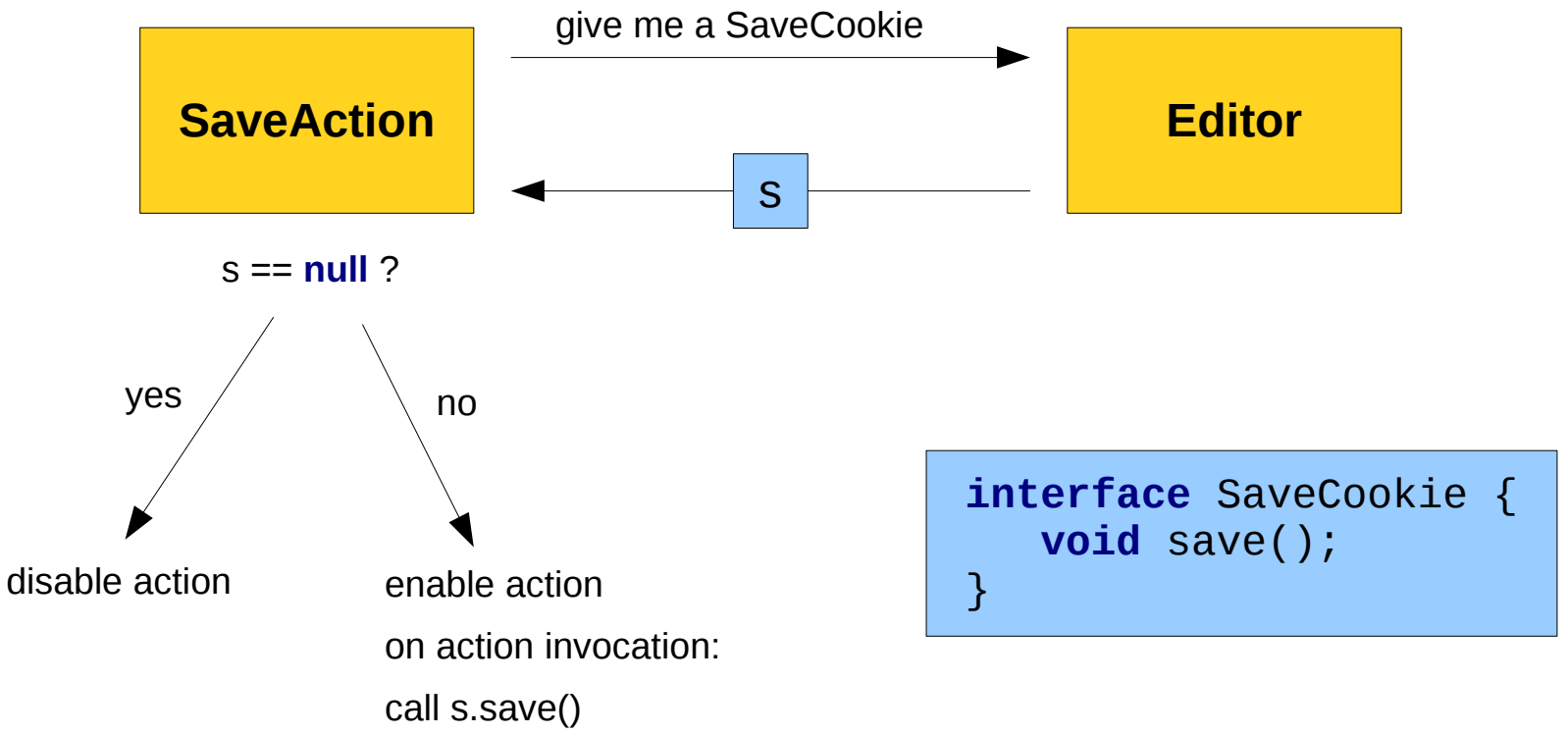


Lookup System



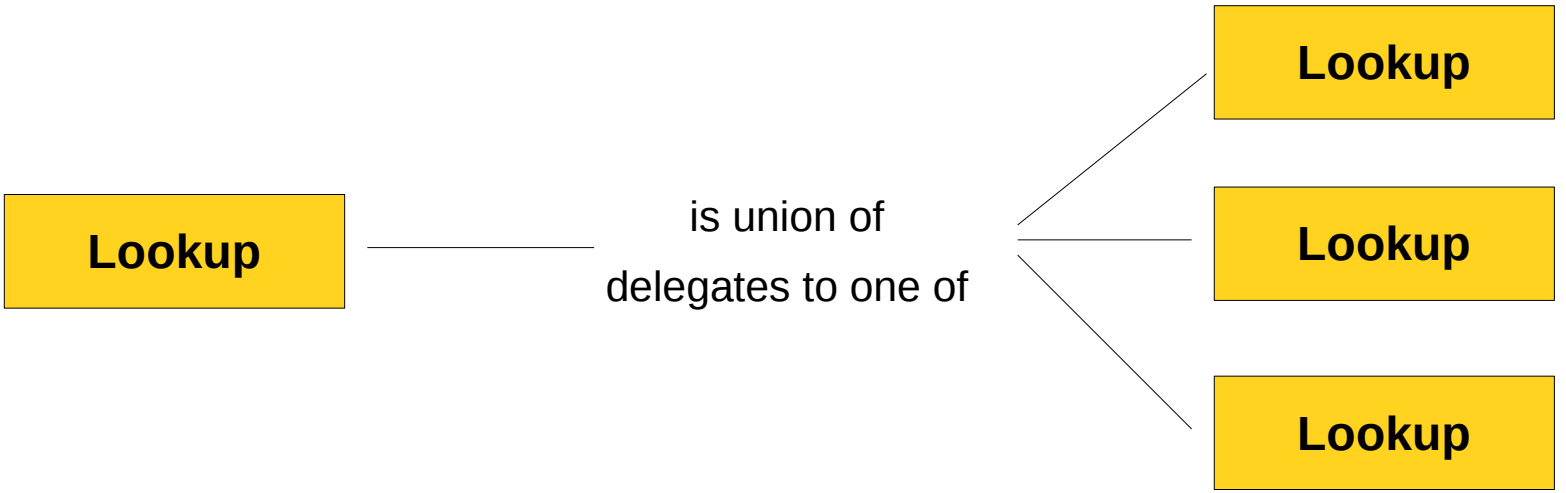

```
InstanceContent content = new InstanceContent();
Lookup lookup = new AbstractLookup(content);

Collection<? extends Integer> result;
result = lookup.lookupAll(Integer.class);
                                                    // empty list
content.add(2);
content.add(3);
result = lookup.lookupAll(Integer.class);
                                                    // {2, 3}
content.add("vier");
result = lookup.lookupAll(Integer.class);
                                                    // {2, 3}
Collection c = lookup.lookupAll(Object.class);
                                                    // {2, 3, "vier"}
content.remove(3);
result = lookup.lookupAll(Integer.class);
                                                    // {2}
```

Lookup Example Usage




```
interface SaveCookie {
   void save();
}
```

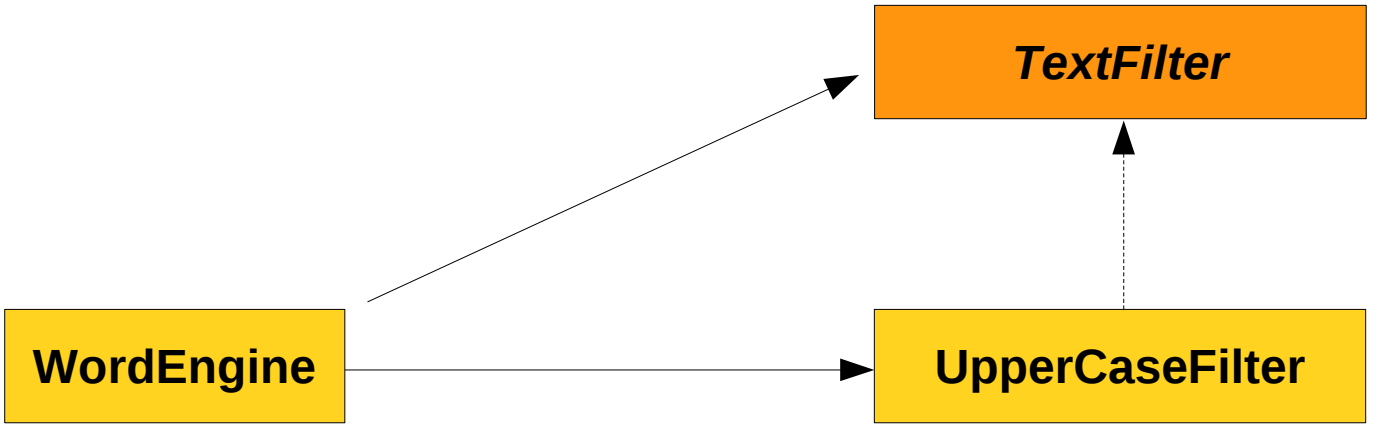
Proxy Lookups



Frequently used lookups in NetBeans

- *Lookup.getDefault()* is the global lookup
- *Utilities.actionsGlobalContext()* delegates to lookup of current active window
- Lookup of a view (e.g. ListView, TreeView) delegates to lookup of current selected item

Dependency Removal



File META-INF/services/at.ssw.TextFilter

```
at.ssw.UpperCaseFilter
```

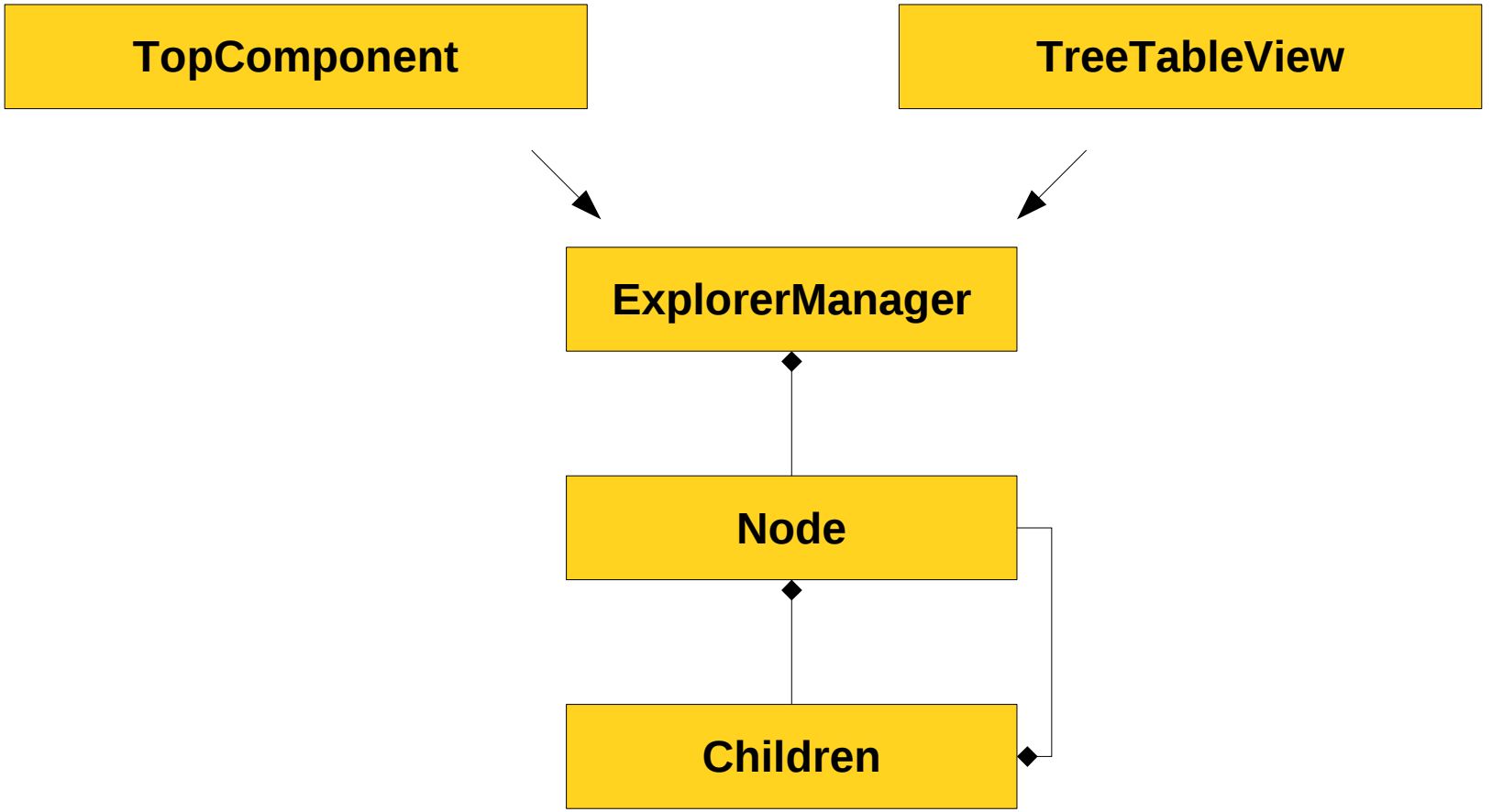
```
TextFilter filter = Lookup.getDefault().lookup(TextFilter.class);
String s = filter.process("test");
```

Lookup Listening


```
Lookup.Result<Integer> result = lookup.lookupResult(Integer.class);
result.addLookupListener(new MyLookupListener());
```

```
class MyLookupListener {

   public void resultChanged(LookupEvent e) {

      Lookup.Result<Integer> result = (Lookup.Result<Integer>)e.getSource();
      System.out.println("Lookup changed!");

      for (Integer i : result.allInstances()) {
        System.out.println(i);
      }
   }

}
```

Explorer and Nodes API

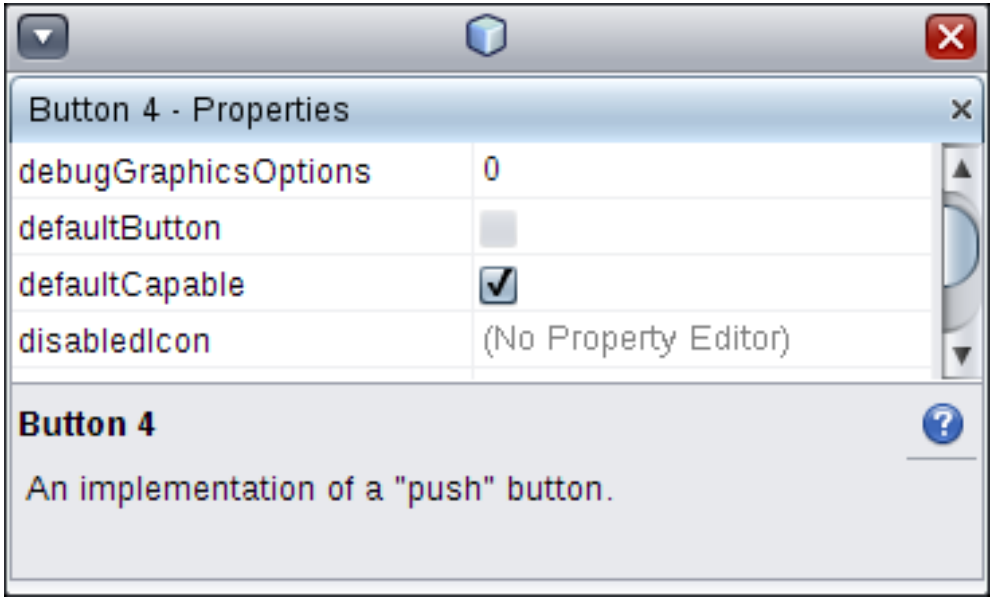


JavaBeans

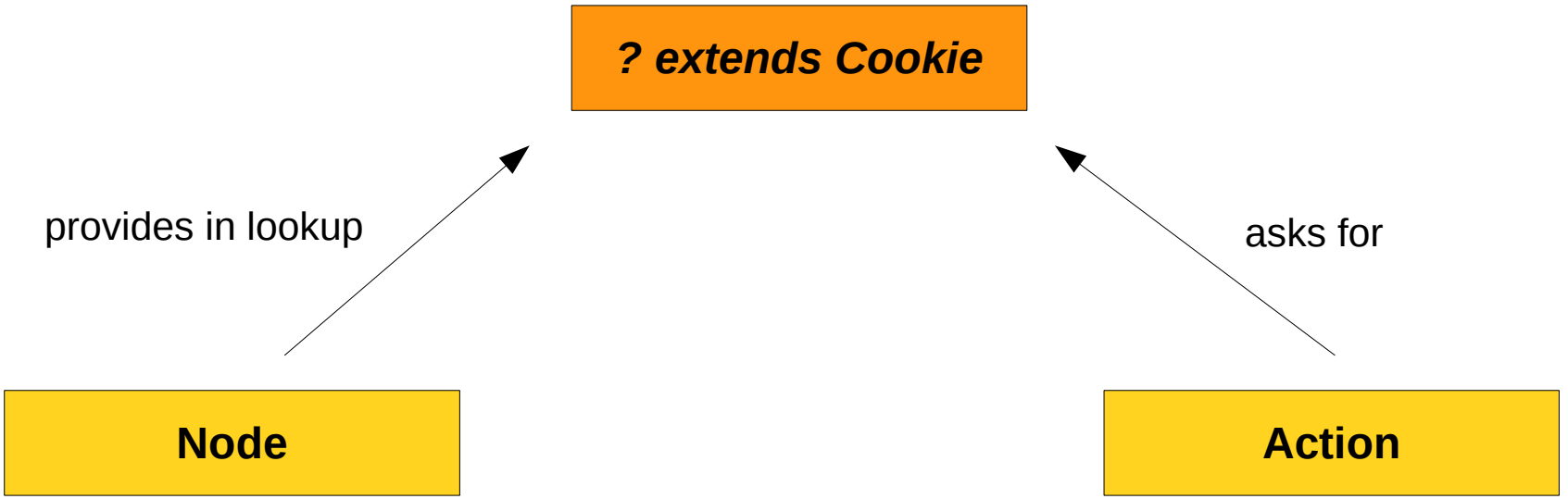
- Specification of a JavaBean
  - via special public Java methods (Introspection)
  - via BeanInfo object

- JavaBeans expose Properties and Events

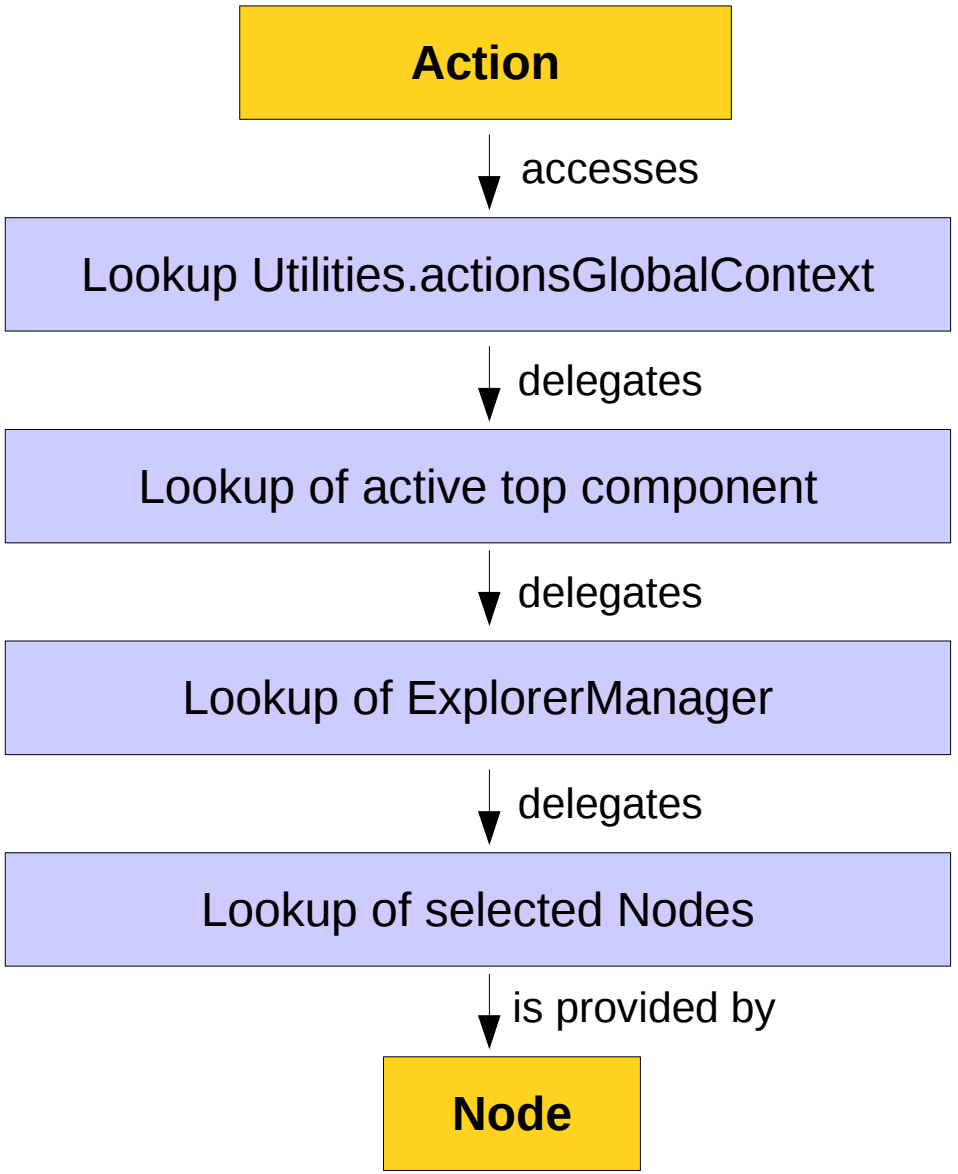
- Persistence



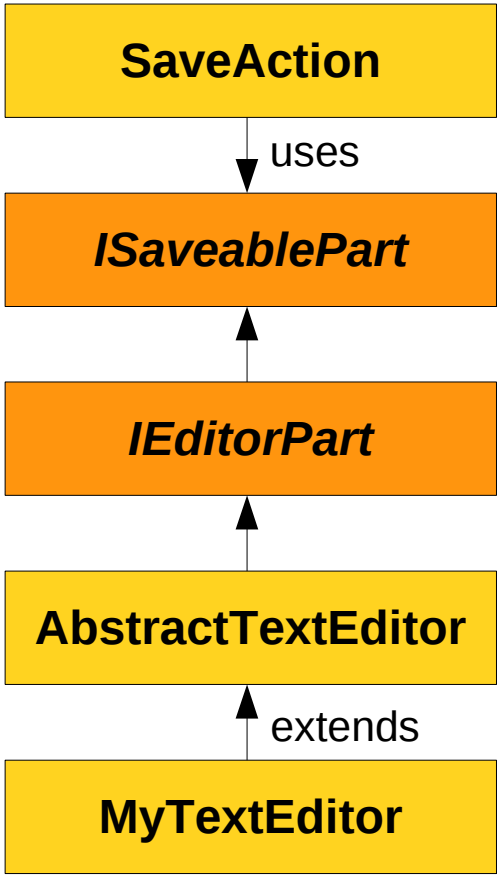
Nodes and Actions (1)



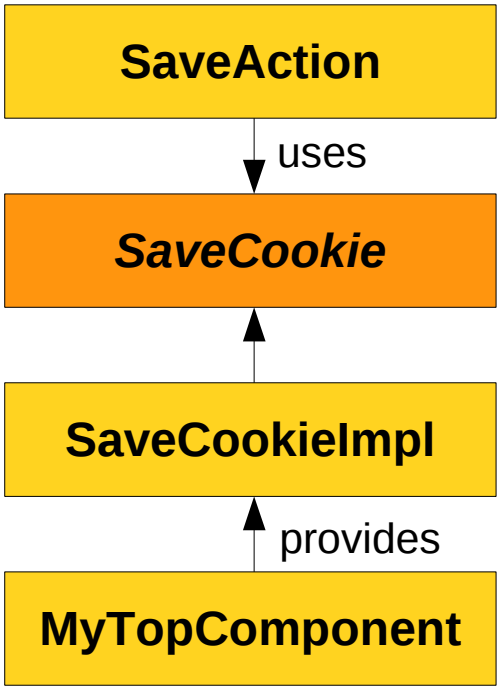
Nodes and Actions (2)



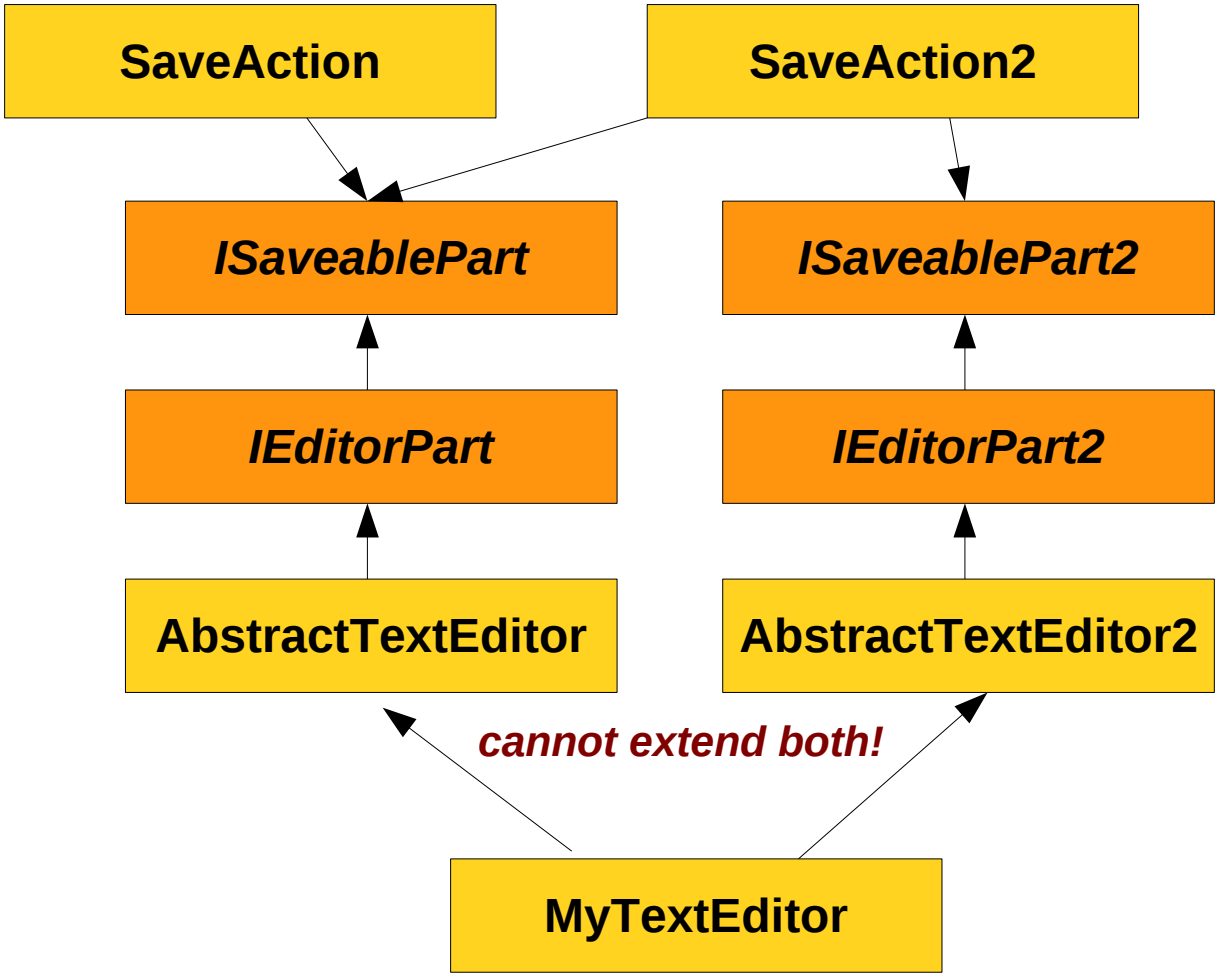
Backward Compability (1)



Introducing a new method in save interface?



Backward Compability (2)



Backward Compability (3)