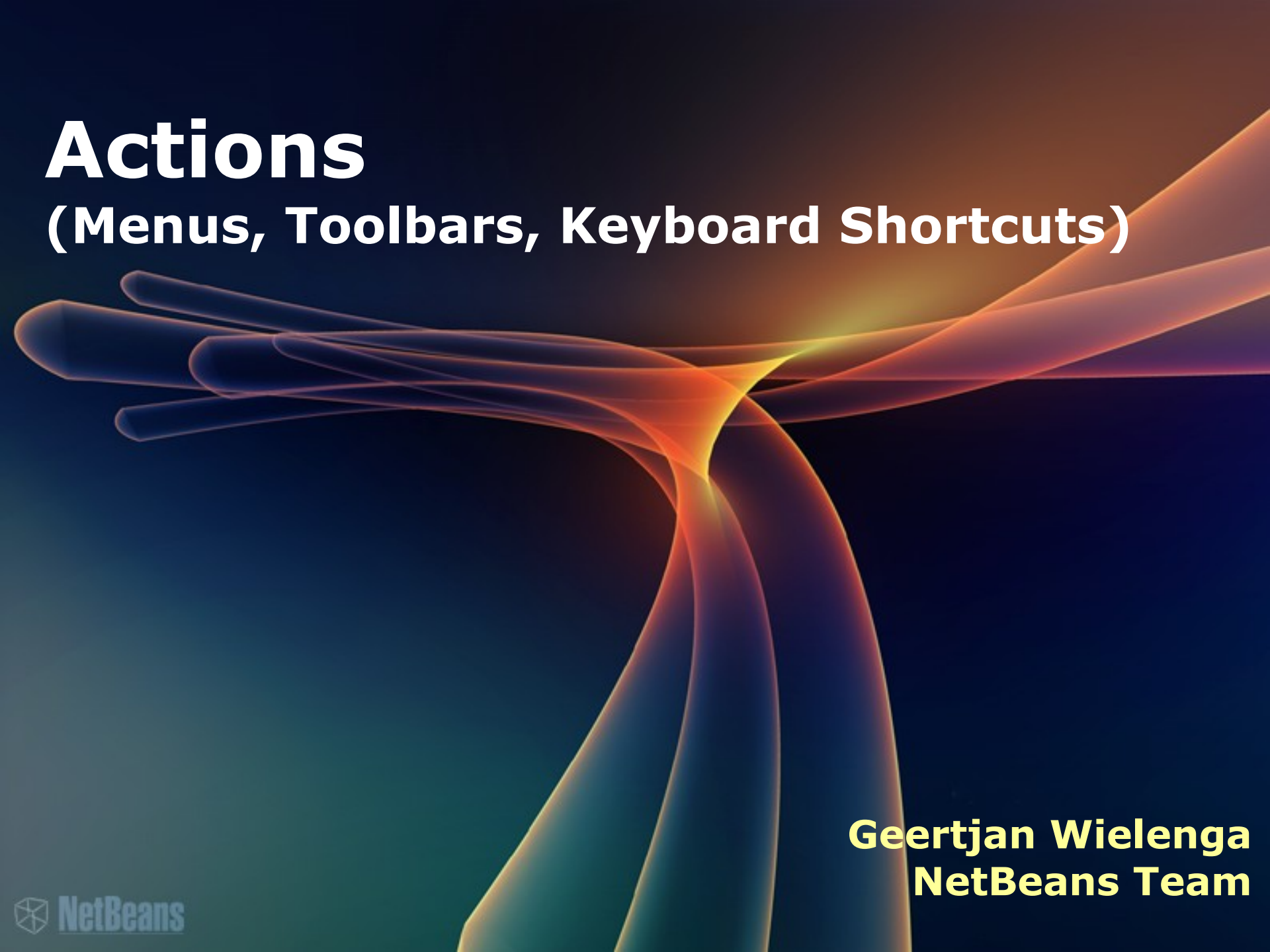
Actions
(Menus, Toolbars, Keyboard Shortcuts)

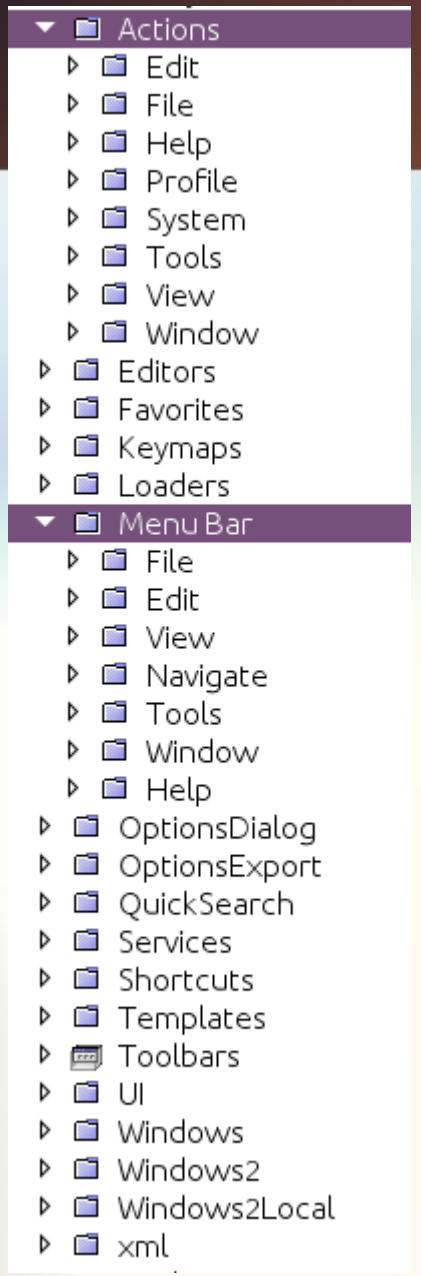


Geertjan Wielenga
NetBeans Team

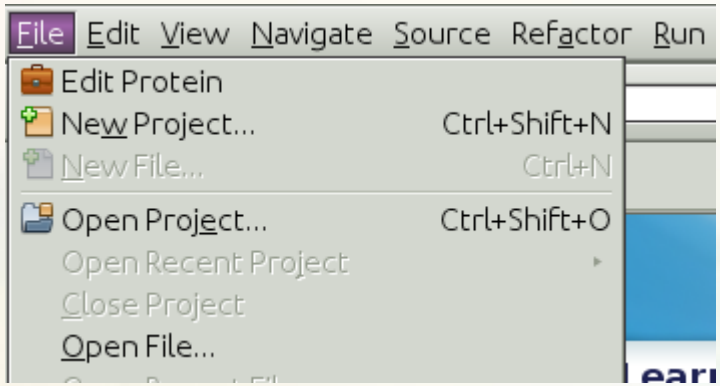
NetBeans

Agenda

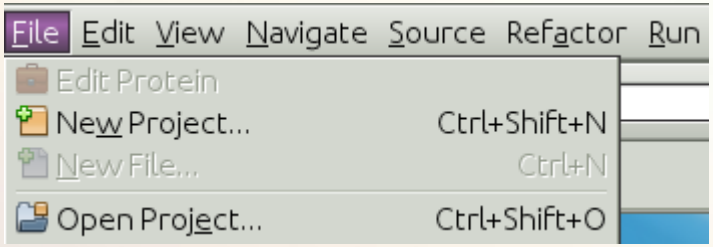
- Introduction
- Menus
- Toolbars
- Keyboard Shortcuts

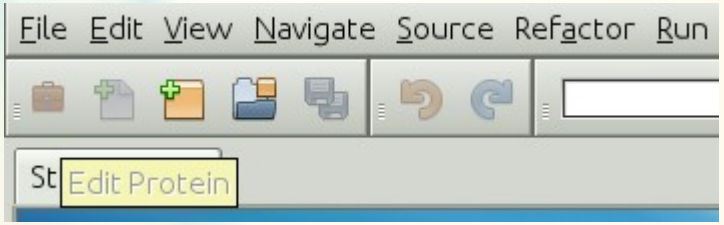


Plugging Actions Into NetBeans

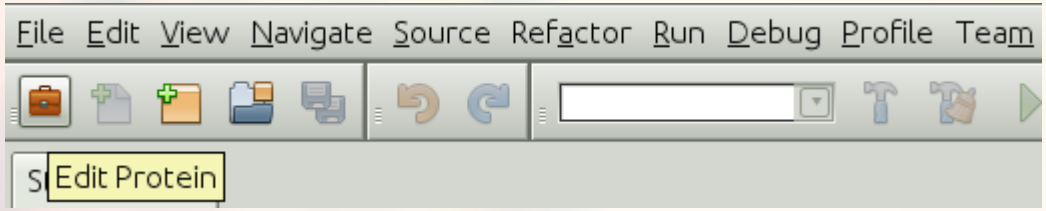


Menus



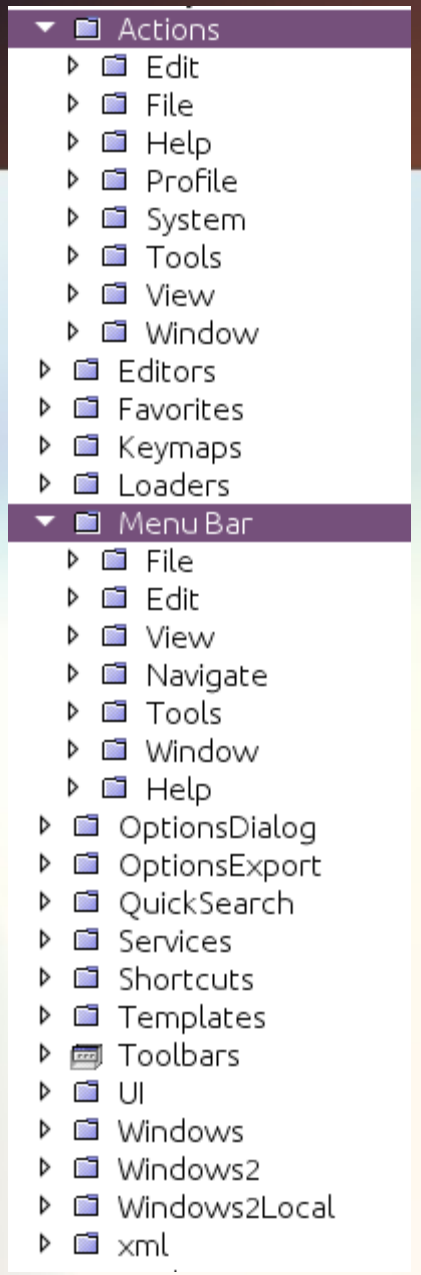


Toolbars



How It Works

- Register an ActionListener into the NetBeans Platform's registry



How It Works

- Register an ActionListener into the NetBeans Platform's registry
- The NetBeans Platform finds the registration and:
  - creates a menu item if in “Menu”
  - creates a toolbar button if in “Toolbars”
  - creates a keyboard shortcut if in “Shortcuts”

How It Works

- Register an ActionListener into the NetBeans Platform's registry
- The NetBeans Platform finds the registration and:
  - creates a menu item if in “Menu”
  - creates a toolbar button if in “Toolbars”
  - creates a keyboard shortcut if in “Shortcuts”
- Depending on how you register, you get enablement

How It Works

- Register an ActionListener into the NetBeans Platform's registry
- The NetBeans Platform finds the registration and:
  - creates a menu item if in “Menu”
  - creates a toolbar button if in “Toolbars”
  - creates a keyboard shortcut if in “Shortcuts”
- Depending on how you register, you get enablement
- When invoked, ActionListener performs

Plugging an Action in the Menu Bar

```
@ActionID(category = "Protein", id = "org.mypkg.EditProteinAction")
@ActionRegistration(iconBase = "org/mypkg/EditProtein.png",
displayName = "#CTL_EditProteinAction")
@ActionReferences({
    @ActionReference(path = "Menu/File", position = 0)
})
@Messages("CTL_EditProteinAction=Edit Protein")
public final class EditProteinAction implements ActionListener {

    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO implement action body
    }

}
```

Plugging an Action in the Menu Bar

```
<folder name="Actions">
   <folder name="Protein">
      <file name="org-mypkg-EditProteinAction.instance">
         <attr bundlevalue="org.mypkg.Bundle#CTL_EditProteinAction" name="displayName"/>
         <attr methodvalue="org.openide.awt.Actions.alwaysEnabled" name="instanceCreate"/>
         <attr name="delegate" newvalue="org.mypkg.EditProteinAction"/>
         <attr name="iconBase" stringvalue="org/mypkg/jxgraph-in-nb-scenario.png"/>
         <attr boolvalue="false" name="noIconInMenu"/>
      </file>
   </folder>
</folder>
<folder name="Menu">
   <folder name="File">
      <file name="org-mypkg-EditProteinAction.shadow">
         <attr name="originalFile" stringvalue="Actions/Protein/org-mypkg-EditProteinAction.instance"/>
         <attr intvalue="0" name="position"/>
      </file>
   </folder>
</folder>
```

Plugging an Action in the Tool Bar

```
@ActionID(category = "Protein", id = "org.mypkg.EditProteinAction")
@ActionRegistration(iconBase = "org/mypkg/EditProtein.png",
displayName = "#CTL_EditProteinAction")
@ActionReferences({
    @ActionReference(path = "Menu/File", position = 0),
    @ActionReference(path = "Toolbars/File", position = 0)
})
@Messages("CTL_EditProteinAction=Edit Protein")
public final class EditProteinAction implements ActionListener {

    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO implement action body
    }

}
```

Plugging an Action in the Tool Bar

```
<folder name="Actions">
   <folder name="Protein">
      <file name="org-mypkg-EditProteinActionListener.instance">
         <attr bundlevalue="org.mypkg.Bundle#CTL_EditProteinActionListener" name="displayName"/>
         <attr methodvalue="org.openide.awt.Actions.alwaysEnabled" name="instanceCreate"/>
         <attr name="delegate" newvalue="org.mypkg.EditProteinActionListener"/>
         <attr name="iconBase" stringvalue="org/mypkg/briefcase_16.png"/>
         <attr boolvalue="false" name="noIconInMenu"/>
      </file>
   </folder>
</folder>
<folder name="Toolbars">
   <folder name="File">
      <file name="org-mypkg-EditProteinActionListener.shadow">
         <attr name="originalFile" stringvalue="Actions/Protein/org-mypkg-EditProteinActionListener.instance"/>
         <attr intvalue="0" name="position"/>
      </file>
   </folder>
</folder>
```

Plugging an Action in a Node

```
@Override
public Action[] getActions(boolean context) {
        List<? extends Action> proteinActions = Utilities.actionsForPath("Actions/Protein");
        return proteinActions.toArray(new Action[proteinActions.size()]);
}
```

Enablement

```
<folder name="Actions">
    <folder name="Edit">
        <file name="org-mypkg-EditProteinActionListener.instance">
            <attr methodvalue="org.openide.awt.Actions.alwaysEnabled" name="instanceCreate"/>
            <attr name="delegate" newvalue="org.mypkg.EditProteinActionListener"/>
            <attr name="iconBase" stringvalue="org/mypkg/briefcase_16.png"/>
        </file>
    </folder>
</folder>
```

Enablement

```
<folder name="Actions">
    <folder name="Edit">
        <file name="org-mypkg-EditProteinActionListener.instance">
            <attr methodvalue="org.openide.awt.Actions.alwaysEnabled" name="instanceCreate"/>
            <attr name="delegate" newvalue="org.mypkg.EditProteinActionListener"/>
            <attr name="iconBase" stringvalue="org/mypkg/briefcase_16.png"/>
        </file>
    </folder>
</folder>

<folder name="Actions">
    <folder name="Edit">
        <file name="org-mypkg-EditProteinActionListener.instance">
            <attr methodvalue="org.openide.awt.Actions.context" name="instanceCreate"/>
             <attr name="type" stringvalue="nl.vu.domain.Protein"/>
             <attr methodvalue="org.openide.awt.Actions.inject" name="delegate"/>
             <attr name="injectable" stringvalue="org.mypkg.EditProteinActionListener"/>
             <attr name="selectionType" stringvalue="EXACTLY_ONE"/>
            <attr name="iconBase" stringvalue="org/mypkg/briefcase_16.png"/>
        </file>
    </folder>
</folder>
```

Enablement

```
@ActionID(category = "Edit",
id = "org.mypkg.EditProteinActionListener")
@ActionRegistration(iconBase = "org/mypkg/briefcase_16.png",
displayName = "#CTL_EditProteinActionListener")
@ActionReferences({
    @ActionReference(path = "Menu/File", position = 0),
    @ActionReference(path = "Toolbars/File", position = 0)
})
@Messages("CTL_EditProteinActionListener=Edit Protein")
public final class EditProteinActionListener implements ActionListener {

    private Protein p;

    public EditProteinActionListener(Protein p) {
        this.p = p;
    }

    public void actionPerformed(ActionEvent e) {
          JOptionPane.showMessageDialog(null, “Hello “ + p.getId);
    }

}
```

Multiple Selected

```
@ActionID(category = "Edit",
id = "org.mypkg.EditProteinActionListener")
@ActionRegistration(iconBase = "org/mypkg/briefcase_16.png",
displayName = "#CTL_EditProteinActionListener")
@ActionReferences({
    @ActionReference(path = "Menu/File", position = 0),
    @ActionReference(path = "Toolbars/File", position = 0)
})
@Messages("CTL_EditProteinActionListener=Edit Protein")
public final class EditProteinActionListener implements ActionListener {

    private List<Protein> proteins;

    public EditProteinActionListener(List<Protein> proteins) {
        this.proteins = proteins;
    }

    public void actionPerformed(ActionEvent e) {
          //Iterate through the list of proteins
    }

}
```

Callback System Action

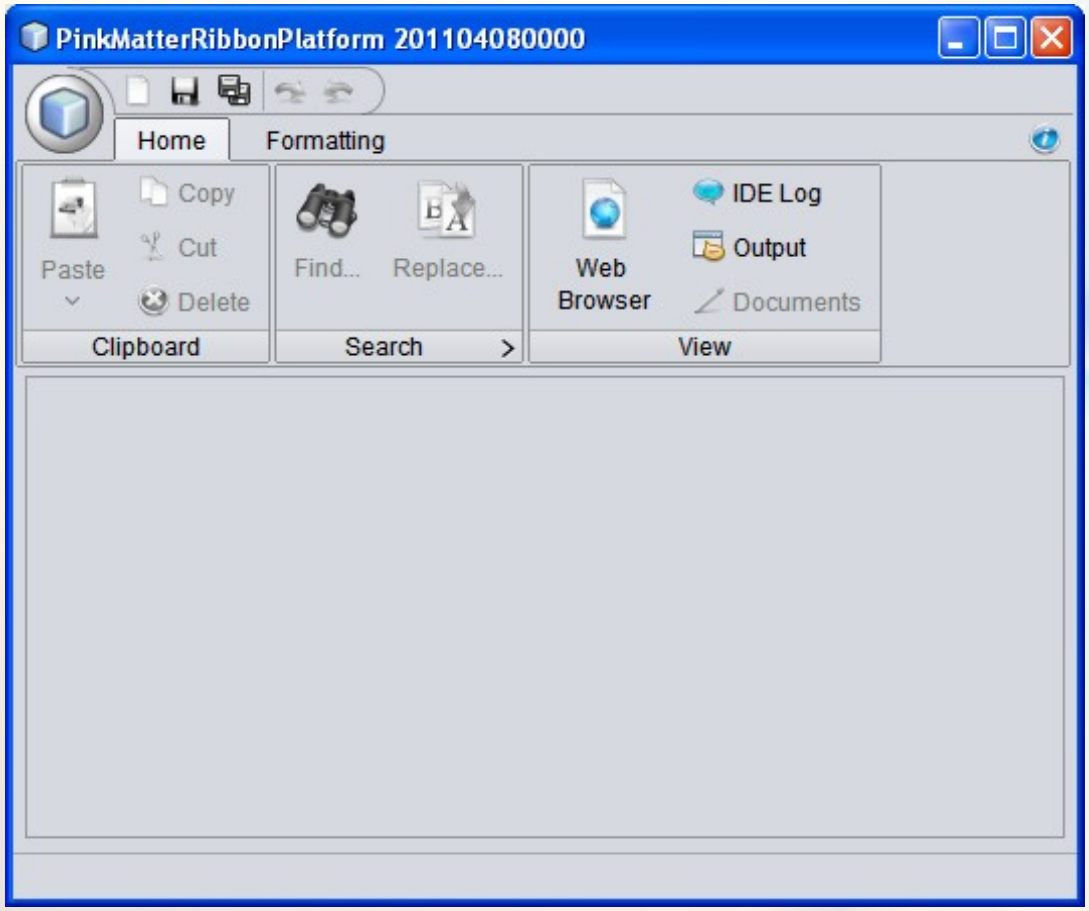
```
ActionMap actionMap = getActionMap();
actionMap.put("cut-to-clipboard", new AbstractAction("") {
    @Override
    public void actionPerformed(ActionEvent e) {
        // cut something
    }
});
associateLookup(ExplorerUtils.createLookup(em, actionMap));
```

See: http://wiki.netbeans.org/FaqEditorMacros

Presenters: Custom Views

- Presenter.Toolbar
- Presenter.Popup
- Presenter.Menu

Ribbon Bar



http://platform.netbeans.org/tutorials/nbm-ribbonbar.html

Summary

- Menus, Toolbars, Keyboard Shortcuts
- Always Enabled Action
- Conditionally Enabled Action
- Callback System Action
- Presenters