Authorization in RCP Applications

Requirements and Solutions
Problem

Most commercial applications need to handle different permissions for different users

RCP provides some support for Actions and Commands

You may circumvent these restrictions by other means:

Use the *Open Perspective/View* menu entry
Be creative, add a plugin to start every view/editor

Eclipse Bugzilla Enhancement Request 201052
Requirements

Role based permissions on higher level components
Commands, Views, Editors, Wizards, Perspectives
Not GUI components like buttons, fields, menus …

Permissions should be specified without coding
Declarative in plugin.xml
Based on component IDs

Authentication will be addressed in the future
Simple solution: Login using the splash screen
Using Equinox Transformations

- Equinox provides a way to transform `plugin.xml`
- This happens during bundle loading
- You may prevent all or part of a bundle from loading
- Must add a bundle to your application
  + Secure, there actually is no code for the restricted functions
  + Works with every Equinox application
  - Has to be run before the application is fully loaded
  - Doesn’t allow change of permissions during runtime
Sample Configuration Snippet

```xml
<view
    name="Public View"
    class="rcp.auth.transformations.example.PublicView"
    id="RCPAuthTransformExample.publicView"
    userRights="read">
</view>

<view
    class="rcp.auth.transformations.example.SecretView"
    id="RCPAuthTransformExample.secretView"
    userRights="write"
    name="Secret View">
</view>
```
Using Activities and Expressions

Supported for Commands in Eclipse 3.3
Extended for required components

Use the Expression framework to specify the permissions needed for a component
  + Works during complete runtime
  + Highly customizable
O needs programming to use its full potential
- Needs Eclipse Ganymede (M5 partial, M6 full support planned)
Sample Configuration Snippet

<activity
    id="publicViewActivityId"
    name="Public View Activity">
    <enabledWhen>
        <with
            variable="accessId">
            <iterate
                ifEmpty="false"
                operator="or">
                <equals
                    value="publicView">
                </equals>
            </iterate>
        </with>
    </enabledWhen>
</activity>
Sample Configuration Snippet

<activityPatternBinding
    activityId="publicViewActivityId"
    isEqualityPattern="true"
    pattern="rcp.authorization.activities.example/RCPAuthActivitiesExample.publicView">
</activityPatternBinding>

<activityPatternBinding
    activityId="secretViewActivityId"
    pattern=".*RCPAuthActivitiesExample\secretView">
</activityPatternBinding>
Summary

- Both solutions can solve the problem
- Both can be used declarative
- Activities are much more flexible
Further information

Discussion papers and sample programs are available at

Further questions and discussions:
Meet me at the BREDEX booth

Thank you for your interest.