Eclipse Plug-ins Development

Radu Farcas – Eclipse Plug-Ins Developer
Catalin Udma – Linux Developer

02-July-2013

Plug-in Development Concepts

- Plug-in
  - Used to group your code into a modular, extendable and sharable unit.
- Feature
  - Used to package a group of plug-ins together into a single installable and updatable unit
- Extensions and Extension Points
  - When a plug-in wants to allow other plug-ins to extend or customize portions of its functionality, it will declare an extension point
  - Plug-ins that want to connect to that extension point must implement that contract in their extension
- Fragment
  - Used to replace or extend the functionality of an existing plug-in
- Target Platform
  - Refers to the plug-ins which your workspace will be built and run against

Process Viewer Example

- **Requirement**: Create an Eclipse view with the following functionality:
  - Display the list of processes on the current Linux platform
  - Offer the possibility to attach and display PC for a process
  - Capture system calls made by a Linux process

- **Background**: System call
- **Background**: ptrace

Create the Processes Plug-in

- Use New Project Wizard and create a new plug-in with a view
  - Name the view “Processes”
- Create the following actions:
  - Refresh
  - Attach
  - Capture system calls
- Add the following columns to the table in the Processes view
  - PID
  - Command
  - PC

Brief Look at SWT and JFace

- ***SWT*** is a graphical widget toolkit for use with the Java platform (currently maintained by Eclipse Foundation)
  - Creation of Dialogs, Windows, Combos, Lists, Buttons, etc.
  - Same look and feel as the native widgets on any platform
- ***JFace*** provides classes and frameworks which simplify common *SWT* use cases
  - simplify the mapping of a data model to a visual representation
  - *Viewers* are used to display a model in a SWT widget (list, combo, tree or table). ***TableViewer*** is used in our example.
- Provide content for the viewer :
  - *IStructuredContentProvider* – provides the items of a table, list or combo
  - *ILabelProvider* – provides icons and labels for every element

JFace in Processes View

- *TableViewer* is created in *createPartControl()* method

- Provide content for the view:
  - *viewer.setContentProvider(new ViewContentProvider());*
    - *public Object[] getElements(Object parent)*
  - *viewer.setLabelProvider(new ViewLabelProvider());*
    - *public String getColumnText(Object obj, int index)*
    - *public Image getColumnImage(Object obj, int index)*

- Provide a different implementation for the *TableViewer*
  - *ProcessesTableViewer* with three columns

Provide Content for Processes View

- Declare and create an array of objects that will hold the information about processes (PID, Command and PC)
  - ProcessData class and array of processes
- Return the array of processes as the elements for our table
  - *ViewContentProvider. getElements(Object parent)*
  - Return process information as text for every column of the table
    - *ViewLabelProvider. getColumnText(Object obj, int index)*
      - PID for column 0, Command for column 1 and PC for column 2
  - Return an icon only for column 0
    - *ViewLabelProvider. getColumnImage(Object obj, int index)*

Linux Interactor Class

- Wrapper over ProcessBuilder Java lang class
  - ProcessBuilder - class is used to create and interact with operating system processes
- LinuxInteractor interface
  - *executeCommand(String, boolean)*
  - *convertStreamToStr(InputStream)*
  - *postCommand(String)*
- Command to obtain process list
  - *“ps ax -o pid= -o comm=“*
- Command to attach to a process
  - *sudo ~/linux_tools/ptrace_attach %s*
- Command to obtain system calls for a process
  - *sudo strace -p %s 2> /tmp/%s*

How Do I Work with Extension Points?

- Find extension points
- – Documentation
  - – Plugin.xml editor – ‘Add…’
  - – Plug-ins registry editor
  - – Load SDK plug-in as projects with source folders
- Read extension point description
- Look for examples
  - – Existing places where the extension points have been previously used
- Deploying your solution
  - – export wizard via *File → Export → Plug-in Development → Deployable plug-ins and fragments*.