Protégé Plugin Development

Ray Fergerson

Overview

- Part I
  - What is a Plugin?
  - How Plugins Work
  - Plugin Types and Capabilities
- PART II
  - Plugin Packaging
  - Plugin Bundling
  - Plugin Licensing
  - Coming Changes

Out of Scope

- Standard Java Development
    - Coding
    - Packaging (jars)
    - Utilities
- Implementation mechanisms
- Development environments
- Non-plugin Protégé extensions

What is a Plugin?

- Extension to Protégé
  - Requires no source code modifications
  - Loaded and managed by system
  - Changes way Protégé works
- Implementation of a Java *interface*
- Packaged as *jars*
- Installed in subdirectory of Protégé *plugins*

How Plugins Work

- Protégé, at startup, loads jars directly below *plugins* subdirectory
- Jars contain description of contained plugins
  - meta_inf/manifest.mf
- System creates instances of plugin
- System calls plugin methods when needed “Don’t call us, we’ll call you.”

Types of Plugins

- TabWidget
- SlotWidget
- KnowledgeBaseFactory (“Backend”)
- ProjectPlugin
- ExportPlugin
- CreateProjectPlugin

Plugin: TabWidget

- What is it?
  - Large piece of screen real-estate
  - Can interact with domain KB

    browse, change, delete, corrupt
- What are its limitations?
  - Difficult to supplement or even interact with other tabs
- How hard is it to create?
  - Easy (1 day)

TabWidget Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin: SlotWidget

- What is it?
  - UI Control which allows the user to display and modify a slot value
  - Follows a protocol for hiding interaction KB
- What are its limitations?
  - Works best with a *single* slot
- How hard is it to create?
  - Easy (1 day)

SlotWidget Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin Type: KnowledgeBaseFactory

- What is it?
  - Replacement for standard storage mechanisms
    - Database
    - External server
    - ...
  - Allows for parsing of different file formats
- What are its limitations?
  - Difficult to manipulate UI
  - Implementations tend to be buggy
- How hard is it to create?
  - Hard (>= 1 month)
  - Consider Import/Export plugin instead

KnowledgeBaseFactory Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin Type: ProjectPlugin

- What is it?
  - Code that executes when “things happen” to a project (create, load, display, close, etc)
  - Get access to project, view, menu bar, tool bar and can modify them as you like
- How hard is it to create?
  - Easy (1 day)

ProjectPlugin Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin Type: ExportPlugin

- What is it?
  - Code that saves (part of) a knowledge-base in any format to *somewhere else*
    - files, servers, web, ...
  - No change of the current backend
  - No guarantee of “lossless round trip”
  - No “live” connection
- How hard is it to create?
  - Medium (1 week)

ExportPlugin Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin Type: ImportPlugin

- What is it?
  - Code that creates a knowledge-base from information from *somewhere else*
    - files, servers, web, ...
  - No change of the current backend
  - No guarantee of “lossless round trip”
  - No “live” connection
- How hard is it to create?
  - Medium (1 week)

ImportPlugin Example

For code see:

http://protege.stanford.edu/conference/2005/slides

Plugin Packaging

- Plugin can contain doc and “about box” URL’s or pages to integrate into the system
- Create a directory structure like:

```
edu.stanford.smi.protegex.myproject/
    myproject.jar
    otherlibrary.jar
    myproject_doc.html
    myproject_about.html
    plugin.properties
```

- Zip it up and give it to your friends

Plugin Bundling

- Plugins of general usefulness can be “bundled” with the full release and made available to all users
- Advantage:
  - You may get a lot of users quickly
- Disadvantage:
  - You may get a lot of users quickly
- In order to be bundled the plugin must be:
  - *Well Formed*
  - *Well Behaved*
  - *Well Maintained*

Plugin Bundling – *Well Formed*

- jar file in an appropriate, recognizable directory
  - approriate: “edu.myorg.mygroup.myproject”, not “foo”
  - recognizable: last directory element: “mytab” not “foo”
- About Box and Documentation entries
- Minimal size
  - minimal documentation
    - links to more extensive documentation on web
    - no PDF, MS Word, large image files
  - no source
  - at most one small example project
  - readme.txt file if necessary
- isSuitable implemented if appropriate
  - Is it requires certain sorts of projects or additional installation (shared libraries, etc)

Plugin Bundling – *Well Behaved*

- Must “work” (not crash on startup) with the current release
- Minimal information (just errors) printed to the console window
  - Single startup line is ok (but certainly not required)
  - No tracing
- Must start up and shut down smoothly
  - No time consuming code executed in static initializer
  - No long start up delays or modal dialogs that block the rest of the system
  - Must free acquired resources in “dispose()”

Plugin Bundling – *Well Maintained*

- Developer/maintainer “responsive” to problems.

- Does not mean that you offer 24x7x365 free support.

Plugin Licensing

- Plugins are not affected by the Mozilla Public License (MPL)
- You can adopt whatever license you want for your plugin
  - Open source (GPL, MPL, BSD)
  - Proprietary
- You can (try to) sell your plugin
- See FAQ for more information on plugin and non-plugin licensing issues
  - http://protege.stanford.edu/faq.html#08.00

Coming Changes

- Major:
  - Revision of the Export plugin interface
- Minor:
  - Allow users to disable installed plugins
  - Additional optional “static interface” methods:
    - isSuitable() for other plugin types
    - buildString() for macro substitution on About Box page
  - Optional localization support for plugins
  - Documented procedures for bundling

Summary

- Plugins provide flexible and powerful mechanisms for extending Protege in many ways.
- Plugins are easy to develop.
- When you encounter places where the default UI is inadequate or clumsy for your needs (and you will!) think about developing a plugin.
- Consider contributing your plugin it back to the community.