API Tooling – Graduation Review

Darin Wright

Component Lead

IBM

January 14, 2008

Outline

- API Tooling Overview
- Active Communities
- Open Source Operation
- The Eclipse Way
- Technical Architecture
- Future

API Tooling Overview

- API Tooling is a set of features designed to assist bundle (plug-in) developers with API maintenance:
  - Defining API contracts
    - what types are API and what usage (implement, extend, instantiate) is allowed
  - Discovering binary incompatibility issues between bundle versions
  - Discovering illegal use of APIs
  - Updating bundle version numbers
  - Adding @since tags
  - Determining lower version bounds on required bundles
  - Searching existing products for specific use of a bundle

API Tooling Overview (more)

- Integrated tooling in the Eclipse SDK
  - An incremental builder creates problem markers
- Command line (batch mode) tools
  - A release engineering tool could leverage API tooling function to generate binary compatibility and API usage reports in an automated build
- API Tooling is part of the PDE Subproject
  - API tooling is not a standalone project. It is a new component that falls under the PDE umbrella.
  - For Ganymede, API tooling will focus on providing function for bundle (plug-in) developers, but is designed to be used with standalone Java code as well.

Active Communities

- Developer Community
  - API tooling has been driven the desire of Eclipse committers to explicitly document and validate API contracts and the need to ensure binary compatibility between releases.
  - API tooling realizes and automates API evolution practices developed by the Eclipse community
    - http://wiki.eclipse.org/Evolving_Java-based_APIs
  - API tooling developers are committers from the Eclipse Project.
- User Community
  - API tooling is in its infancy and does not yet have a large user community. But it is something that all bundle (plug-in) developers will use.

Active Communities (more)

- Open Source Community
  - API Tooling leverages the ASM open source code base (a byte code analysis and manipulation utility)
  - We have received feedback from an ASM developer on the pde-ui-dev mailing list to help improve API tooling's use of ASM.
    - Helps improve our understanding of ASM and the performance of API tooling

Open Source Operation

- API Tooling development has been ongoing in the PDE Incubator component of the Eclipse Project Incubator
  - Use Bugzilla (bugs.eclipse.org) for workflow using the “PDE – Incubators” product and component
  - Maintain WIKI, which includes work schedules:
    - http://wiki.eclipse.org/PDE_UI_Incubator_ApiTools
    - http://wiki.eclipse.org/ApiTools_Milestones
  - Use equinox-dev and pde-dev mailing lists for developer discussions

The Eclipse Way

- API Tooling developers are from the Eclipse project who understand and adhere to the Eclipse Development Process, committer responsibilities and due diligence rules, as well as the Eclipse IP Policy.
  - Note the successful use of parallel IP to approve third party code (ASM 3.0).
  - IP Log: http://dev.eclipse.org/viewcvs/index.cgi/pde-incubator/api-tooling/plugins/org.eclipse.pde.api.tools.doc/iplog.html?content-type=text%2Fplain&view=co
- Developed automated test suite in parallel (@400 tests)
- The developers take part in the larger community such as teaching tutorials and giving talks at EclipseCon, writing articles, participating in news groups and mailing lists, and participating in other projects.
- The tooling assists developers in maintaining API and adhering to guidelines that are fundamental to the evolution of Eclipse.

Technical Architecture

- For details on the technical architecture of API tooling, please see the WIKI:
  - http://wiki.eclipse.org/ApiTools_Architecture
- Also see the User Guide for a description of how developers will use the tooling:
  - http://wiki.eclipse.org/API_Tooling_User_Guide

Future

- Focus for Eclipse 3.4 is to integrate API tooling in the SDK to and build process to deliver critical function (as shown in overview)
  - We do not plan to publish an API for the tooling in the initial release (we would rather have more adopters/users before we commit to any APIs)
- We envision a bright future for API tooling, its application, and evolution
  - For example, imagine a central repository/server where API profiles are stored for every Eclipse based project (and perhaps many Eclipse based products). The profiles can be analyzed/searched for use of APIs allowing developers to understand the potential impact of various API changes.