Project system APIs and Maven Project Type
A case study on modular design
Agenda

- Project system APIs design
- Maven project type
- Platform application built by Maven
Project system overview

- Abstraction over a set of resources
- Extensible
- Constraints
  - Based on folders, 1 project per folder
  - Files and folders belong always to 1 project
Base APIs

- Project
  > Management of projects
- Project UI
  > Display projects in UI
- Queries
  > Basic queries that delegate to projects
Project API

- ProjectManager maintains the list of projects known to the system.
- Project is the base project abstraction.
  - `getProjectDirectory()`
  - `getLookup()`
- FileOwnerQuery finds project that owns files and folders
Project SPI

• ProjectFactory identifies and creates Project instances
  > Implementations registered as global services

• FileOwnerQueryImplementation defines strategy to find the owning project.

• ... and various services to be provided in project's getLookup()
• OpenProjects maintains a list of projects opened and displayed in the UI.
  > open(Project[])  
  > close(Project[])  
  > setMainProject(Project)  
  > getMainProject()
CustomizerProvider allows to create a dialog to customize the project.
LogicalViewProvider creates logical view of project's content
Defining file templates for project
  > RecommendedTemplates
  > PrivilegedTemplates
Queries API

- FileBuiltQuery tells if a particular file was built or not.
- FileEncodingQuery returns the encoding for files.
- VisibilityQuery knows if file shall be visible in UI.
- SharabilityQuery checks if the file or folder shall be shared.
Domain specific APIs

- Java ClassPath
- J2EE deployment
- Debugger, Profiler
- Web, EJB, EAR specific contracts
- WebServices
- Module development
Ant based projects

- Additional APIs to simplify creation of new Ant based project types.
- Support classes to ease development
  - AntProjectHelper
- Allow extending project from 3rd party modules
  - AntBuildExtender
  - ProjectClassPathModifier
Projects for non IDE use

- Viable option if domain model and UI workflow similar to IDE.
- Create own project type
Agenda

- Project system APIs design
- **Maven project type**
- Platform application built by Maven
Maven support overview

**Bridge Modules**
- J2EE
- Persistence
- Module Development
- Graphs
- Web Frameworks

**Base support**

**3rd party libraries**
Bridge modules

- For functionality outside the “core business”
- Increased integration
- Scales with the IDE
- Gets downloaded and enabled based on availability of bridging modules
Base Support

- Implements Project and basic queries
- Uses spi support classes to make project extensible
  - project's lookup
  - Nodes in logical view
  - Panels in project customizer
- Defines own extension points
J2EE Support

- Adds J2EE domain specific SPI implementations into project lookup
  - only if applicable (war, ear, ejb)
  - corresponding file templates enabled
  - Run/Debug project action maps to J2EE deployment
- Run configuration panel added
Example: ActionProvider

- Found in project's lookup
- LookupProvider for additional sources of lookup, registered in layer filesystem.
- LookupMerger<ActionProvider> to define strategy to merge the implementations
Spring framework

- Project agnostic
- Registers itself in projects' lookup when Spring on classpath.
- Configuration panel in customizer.
  - CompositeCategoryProvider in layers
  - AuxiliaryConfiguration from project for storage
Agenda

- Project system APIs design
- Maven project type
- **Platform application built by Maven**
Apache Maven

- Project management system
- Build driven by plugins
- Dependencies in repositories
Module Development

- NetBeans modules handled by nbm-maven-plugin
- Allows building modules, creating NBM files
- Generate update site content
- Application branding
Dependency management

- Transitive vs. direct
- Modules vs. libraries
- Remote repository at deadlock.netbeans.org
Project templates

- Single NetBeans module
- Suite setup for platform based application
IDE Support

- Most File templates available
- Layers browsing
- Public packages marked
- Javadoc, source available in remote repository
What's missing?

- Final application packaging
  - Installer
  - MacOSX Application bundle
  - JNLP
- Project customization panels
Demo

- Maven based NetBeans platform development
Questions & Answers