Sling Architecture

Felix Meschberger

Day Management AG

8. April 2008

Topics

- Modules and Extension Points
- Request Processing
- Resource and ResourceResolver
- Runtime Framework
- Get Sling
- Questions

Modules

Sling API

jcr/api

sling/core

sling/event

jcr/resource

sling/servlet-resolver

sling/scheduler

jcr/jackrabbit

sling/servlets-*

sling/threads

scripting/api

sling/adapter

sling/bundleresource

scripting/*

sling/i18n

commons/*

Maven 2 Plugins

osgi/*

Extension Points

- Servlets and Scripts
- Servlet Filters
- ScriptEngine[Factory]
- ResourceProvider
- JcrDefaultResourceTypeProvider
- AuthenticationHandler
- LocaleResolver
- etc.

Main Components

- SlingMainServlet
  - Outermost Request Handler
  - Starts Request Processing
- ResourceResolver
  - Resolves the URL to a Resource
- ServletResolver
  - Resolver the Resource Type to a Servlet/Script

Basic Request Processing Steps

- Resolve the Resource
  - Source: Request URI
- Resolve Servlet or Script
  - Source: Resource Type
- Call Servlet Filters
- Call Servlet or Script

Resource

- Resource is Sling's abstraction of the thing addressed by the request URI
- Properties of Resources
  - Path, e.g. JCR Item path
  - Type, e.g. sling:resourceType
  - Super Type, e.g. sling:resourceSuperType
  - Adapters
  - Metadata, e.g. last modification date

ResourceResolver

- Accesses Resources
- Abstracts the path resolution
- Abstracts access to the Persistence
- Currently there is a 1:1 mapping between the ResourceResolver and a single JCR Session
- Tasks:
  - Finding Resources
  - Getting Resources
  - Simplification of Query Execution

Kinds of Resources

- JCR Items (Node, Property)
- Servlets (Registered as OSGi Services)
- Synthetic Resources
- Provided Resources (ResourceProvider)

ApacheCon

Servlet Resolution

- Servlets and Scripts are Equal
- Resolution Steps
  - Turn Type of Request Resource to path (e.g. sling:redirect ==> sling/redirect)
  - Apply search path (e.g. [„/libs“, „/apps“])
  - Servlet Name from Extension or Method (e.g. html.jsp, POST.esp)

Leading the Wave of Open Source

Servlet Resolution (Example)

- Search Path : [„/libs“, „/apps“]
- Resource Type : „myapp:sample“
- Request Extension: „html“
- Request Method: „GET“

Servlet Resolution (Example)

```
/apps/myapp/sample/html[.*]
/libs/myapp/sample/html[.*]
/apps/myapp/sample/GET[.*]
/libs/myapp/sample/GET[.*]
-- above for resource super type
/apps/sling/servlet/default/html[.*]
/libs/sling/servlet/default/html[.*]
/apps/sling/servlet/default/GET[.*]
/libs/sling/servlet/default/GET[.*]
```

Runtime Framework: Heritage

- Sling stemms from Communiqué 4
- Communiqué 4 status
  - Some modularisation
  - Incomplete Lifecycle Support
  - Problematic Quick Fixing
  - Restarts required often

Runtime Framework: Requirement

- Modularization
- Dependency Management
  - Code
  - Services
- Lifecycle Management
- Dynamic System Changes
- Configuration Management

Runtime Framework: OSGi

- Core
  - Modularization – Capabilities & Requirements
  - Lifecycle – Install, Start, Stop, Update, Uninstall
  - Services – Get, Use, Unget
  - (Security – JAAS based, not used by Sling)
- Compendium
  - Configuration Admin Service
  - Declarative Services

How is Sling Delivered ?

- OSGi Bundles
- Executable JAR File
- Web Application Archive

But: Parts of Sling are Static !

- The Launcher (5 simple classes)
- The OSGi Framework implementation
- The OSGi core and compendium libraries

- Total: ca. 720KB

- Everything else is a Bundle

Does Sling require the Launcher ?

- No
- Sling can be deployed in any compliant OSGi R4 framework.

- e.g. Integration of Sling into the ServiceMix 4 Framework instance
- e.g. Equinox