APACHE SLING & FRIENDS TECH MEETUP

BERLIN, 26-28 SEPTEMBER 2012

APACHE SLING & SCALA

Jochen Fliedner

About the speaker

Jochen Fliedner
Senior Developer

pro!vision GmbH
Wilmersdorfer Str. 50-51
10627 Berlin
http://www.pro-vision.de

PRO!VISION
SOFTWARE CRAFTSMANSHIP

Spezialized on Adobe CQ and it‘s technology stack since 2003

adaptTo()

Using VM languages in Sling/CQ5

- Want to use Scala, Groovy … for CMS implementations– “templating”.
- For that differentiate between…
  - **Scripting** – textual resource (content)
  - **Components** – precompiled type (bytecode in OSGi bundle)

Using scripts/scripting languages in Java

"We have all heard heated arguments between developers who use scripting languages and developers who use Java. One of the reasons for the war between these two factions is that the process of integrating the two was so difficult that developers on both sides were almost forced to choose one or the other. …"

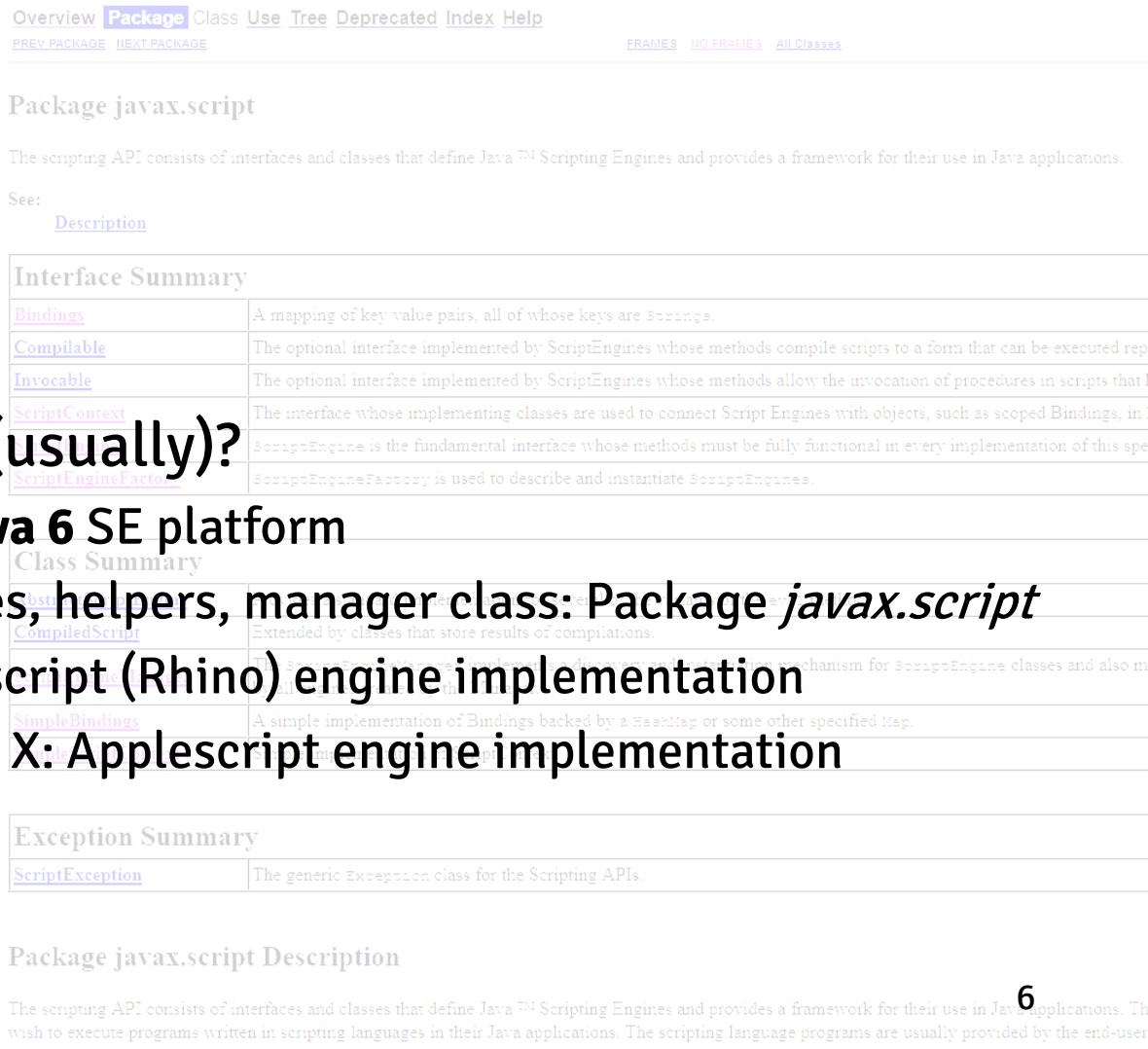
http://www.drdobbs.com/jvm/jsr-223-scripting-for-the-java-platform/215801163

- What to do? Call scripts from Java.
- And the solution for that:
  "Scripting for the Java Platform" JSR 223

Scripting part: “JSR 223: Scripting for the Java Platform”

- Script engine?
  - set of **interface**s describing methods **to execute scripts**
  - providing context to script: **set** and **get** data
  - service provider mechanism: register engine (factory)
  - *javax.script.ScriptEngineManager:* retrieve engine instance
  - supply script string
  - provide *javax.script.Bindings* (or simply put key-value pairs)
  - **eval**uation
  - retrieve result by any obj reference held (*Bindings*)

Scripting part: Script engines implementations


Overview Package Class Use Tree Deprecated Index Help

PREV PACKAGE NEXT PACKAGE FRAMES NO FRAMES All Classes

Package javax.script

The scripting API consists of interfaces and classes that define Java ™ Scripting Engines and provides a framework for their use in Java applications.

See:

Description

Interface Summary

Bindings	A mapping of key value pairs, all of whose keys are Strings.
Compilable	The optional interface implemented by ScriptEngines whose methods compile scripts to a form that can be executed rep
Invocable	The optional interface implemented by ScriptEngines whose methods allow the invocation of procedures in scripts that
ScriptContext	The interface whose implementing classes are used to connect Script Engines with objects, such as scoped Bindings, in
ScriptEngine	ScriptEngine is the fundamental interface whose methods must be fully functional in every implementation of this spe
ScriptEngineFactory	ScriptEngineFactory is used to describe and instantiate ScriptEngines.

Class Summary

CompiledScript	Extended by classes that store results of compilations.
SimpleBindings	A simple implementation of Bindings backed by a HashMap or some other specified Map.

Exception Summary

ScriptException	The generic Exception class for the Scripting APIs.

Package javax.script Description

The scripting API consists of interfaces and classes that define Java ™ Scripting Engines and provides a framework for their use in Java applications. Th
wish to execute programs written in scripting languages in their Java applications. The scripting language programs are usually provided by the end-user

- Where to find (usually)?
  - Comes with **Java 6** SE platform
  - Set of interfaces, helpers, manager class: Package *javax.script*
  - Included: Javascript (Rhino) engine implementation
  - Included on OS X: Applescript engine implementation

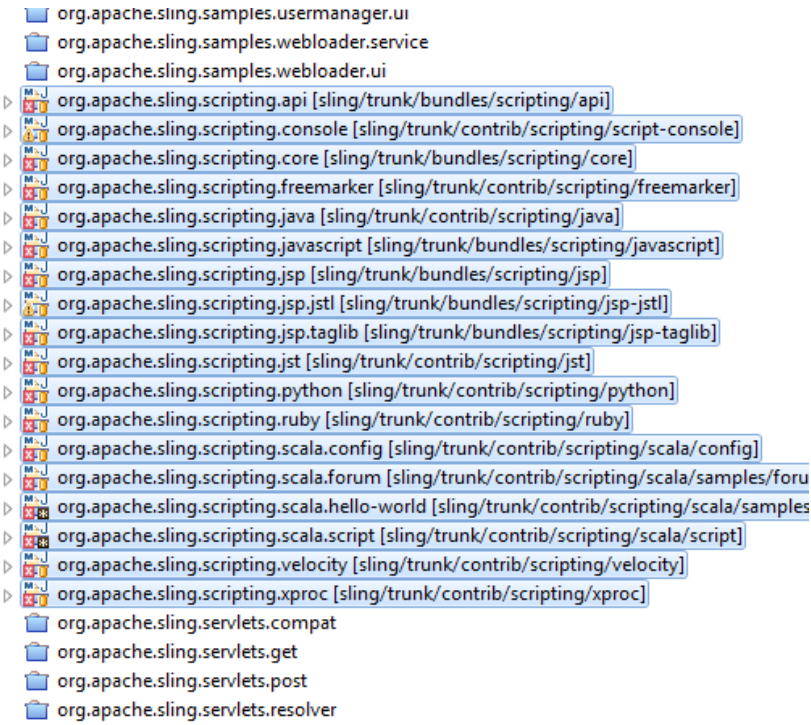
Scripting part: Meet JSR 223 in Sling

- *sling.api <- sling.scripting.api/sling.scripting.core*

- "DefaultSlingScript"

  (sling.scripting API) manager class used to select script engine

- (you already use) JSP (+ Taglib)
- Python, Ruby (as VM languages)
- XProc („xml pipeline processor")
- Freemarker, Velocity (template languages)
- Scala (but using *guggla* now)

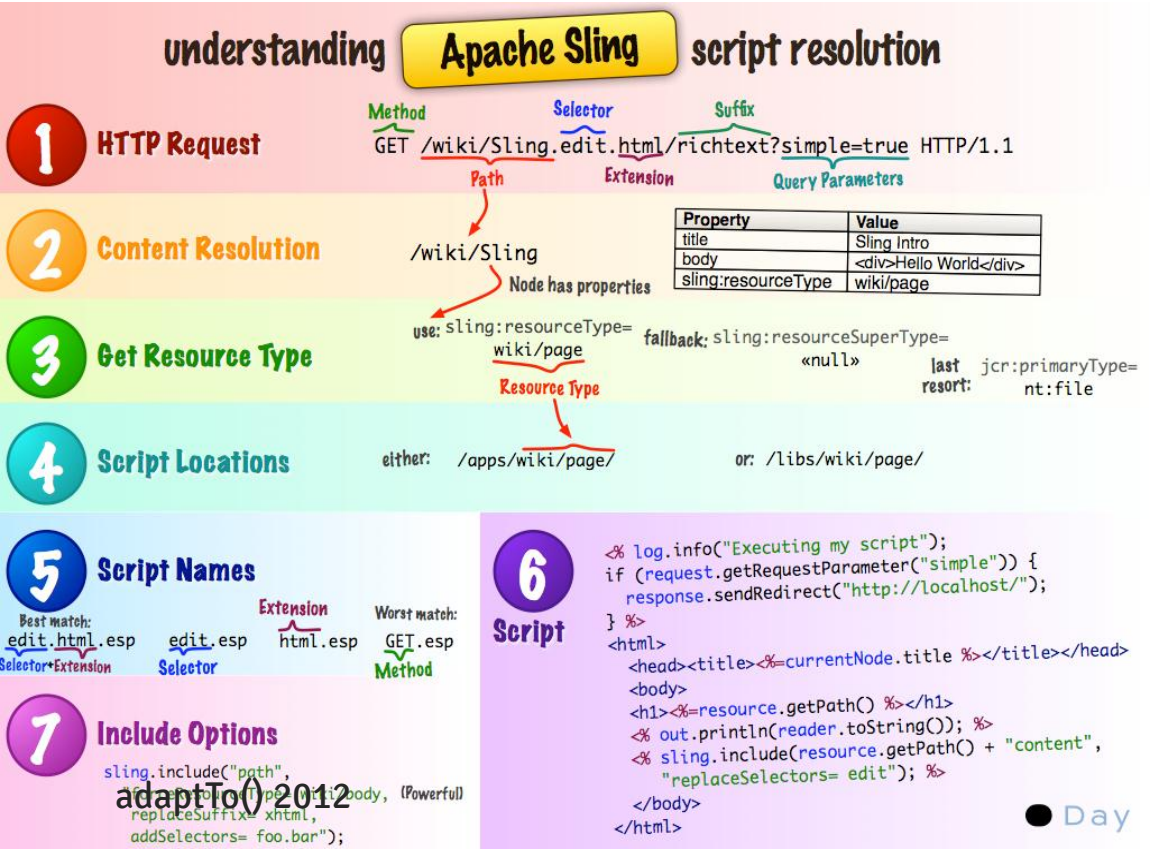


Scripting part: script resolution

Examples - typically with CQ5:

/apps/my_template_set/components/my_component/html.scala

/apps/my_template_set/components/pages/home/sitemap.xml.scala



- Script is a textual resource
- Can be edited with CRX
- Resides in OSGi bundle as its initial **content**
- Script resolution is applied
- Engine selected by “file extension”

Scripting part: Bindings

- What's in for the script? „Bindings" supplied by Sling
- generally:
  org.apache.sling.api.scripting.SlingBindings
  "out", "request", "response", "resource", "sling", "log", "reader"
- JCR specific:
  org.apache.sling.jcr.resource.internal.scripting.JcrObjects BindingsValuesProvider
  "currentNode", "currentSession"

Component part: OSGi

- Precompiled types in bundle
- *javax.servlet.Servlet* interface
- OSGi component, using *Service Component Runtime*
- Needs to be *declared* to the framework as service and component: OSGI-INF/serviceComponents.xml (referenced in manifest file)
- Optionally references to other services have to be *declared* and supported in the code

Component part: SCR Tooling

- Easing by *scr annotations* and *maven-scr-plugin*
- mvn build process with plugin:
  - Identifies types
  - After “compile” look up target bytecode for these classes
  - Scan classes for annotations
  - Generate service component xml
  - Enhance bytecode: “bindRefXY” / “unbindRefXY”
- *@SlingServlet* (service component plus sling metadata)
- *scrplugin 1.8.0 (*FELIX-3550)

Excursion/Example: „CQ5 Groovy Console“ (internals)

- Sling-Application implemented in Groovy
  - org.codehaus.groovy:groovy-all-2.0.1
- Nicely done – and example in many ways:
  - Probably first 3rd-party open sourced stand-alone app for Sling/CQ
  - Serverside logic done in Groovy (scripting AND component)
  - Use of Groovy DSL features (demonstrated later)
  - Nice and lightweight webfrontend (Bootstrap framework, ACE editor)
  - And: Build process and tooling (!)

Scala: guggla – generic Scala script engine

- by Adobe's Michael Dürig
- fka „org.apache.sling.scripting.scala.script“ (sling contrib)
- disconnected from Sling, generic for Java runtime or OSGi:

  The scripting engine supports running in an OSGi container or as part of a stand alone application. It has optional support for compiling from/to a JCR repository.

- more unit tests

adaptTo()

Scala: guggla – a Scala *script*?

- Scala code snippet to be interpreted

- But there is no concept of “script” in Scala
- As there is no *ScriptRunner* or *ScriptShell* class

- Decision: *script* will be code in constructor of a simple object
- With that *script* (String) a greater Scala snippet is “synthesized” (script-args-runner, #preProcess)
- #compile – on classpath – invoked by reflection

Scala Templating demonstration

- Targeting Scala leverage in a sample that is an actual Template set (not a standalone Sling app)
- Using guggla for scripts
- Reintegration into Sling as of *org.apache.sling.scripting.scala.config:*
  - JCR classpath used for compiled scripts
- Own Context type
- overriding the *ScalaInterpreter* impl, for own *Context* type
- Also use Scala *components*, declarations by current scrplugin
- Build and deployment by maven build process
- Make available for download, anyone could reproduce and test

adaptTo()

Example *component*

- '//geometrixx/de.gsitemap.xml'
- create google sitemap xml

- Scala idioms used:
  - a) enriching a type by implicits - here: Page (cq wcm)
  - b) flatten and map methods on Scala collection (Iterable, Array)
  - c) Scala XML literal integrating String values

adaptTo()

Example *script*

- resourceType ‘geometrixx/components/title’
- Exchanged for a Scala script impl. -> modification to geometrixx templating

- Scala idioms used:
  - a) implicitly enriching javax.jcr.Node
  - b) using Scala XML literal integrating String values

- *(see type conversion in “synthetic” code)*

Example: some drawbacks

- *maven-scr-plugin*
  - supports service component xml generation only with a tweak
  - Bytecode manipulation - (un)boundXY - will probably not work with Scala
- Idiomatic Scala may be a little expensive (heavy on heap, gc)
- Possible synchronization problems with *ScalaCompiler* (M. Dürig)

adaptTo()

Summing up…

- Can use Java VM languages in Sling
  - For components and more or less for scripting
  - Start to use usage only for certain parts of the project?
  - Can build up knowledge for any future project
- Scala as a language really is groundbreaking - but better start with basic Scala skillset, read a book
- Groovy is for pragmatists (lightweight, easier to learn - sucessively)

Summing up…

- Recent developments
  - Scr generator reimplemention, plugin improvements
  - Guggla – generic Scala script engine
- Scala CQ5 integration and example code available soon
  - Check out and try, please contribute and criticize
- Check out and see internals
  https://github.com/Citytechinc/cq5-groovy-console

Thanks to…

- Michael Dürig, Bertrand Delacrétaz (sling contrib, guggla)
- Carsten Ziegeler (Felix SCR, generator, annotations, tooling…)

- Prior connected adaptTo() talks:
  - R. Bartl and P. Mannel (Sling introduction, mentioning scripts and components)
  - Todd Haser (Groovy console introduction)

- And last but not least…

APACHE SLING & FRIENDS TECH MEETUP

BERLIN, 26-28 SEPTEMBER 2012

Thank you for your attention!

Any questions?