OSGi for Eclipse Developers

Chris Aniszczyk (EclipseSource)
Martin Lippert (akquinet it-agile GmbH)
Bernd Kolb (SAP AG)

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - Import-Package vs. Require-Bundle
  - Dynamic Bundles
  - Extensions and Services
  - Compendium Services
  - OSGi Tooling
- Conclusion

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Introduction

- OSGi Alliance
  - Worldwide consortium of technology innovators that advances OSGi technology

- OSGi Technology
  - Set of *specifications* that define a dynamic component system for Java

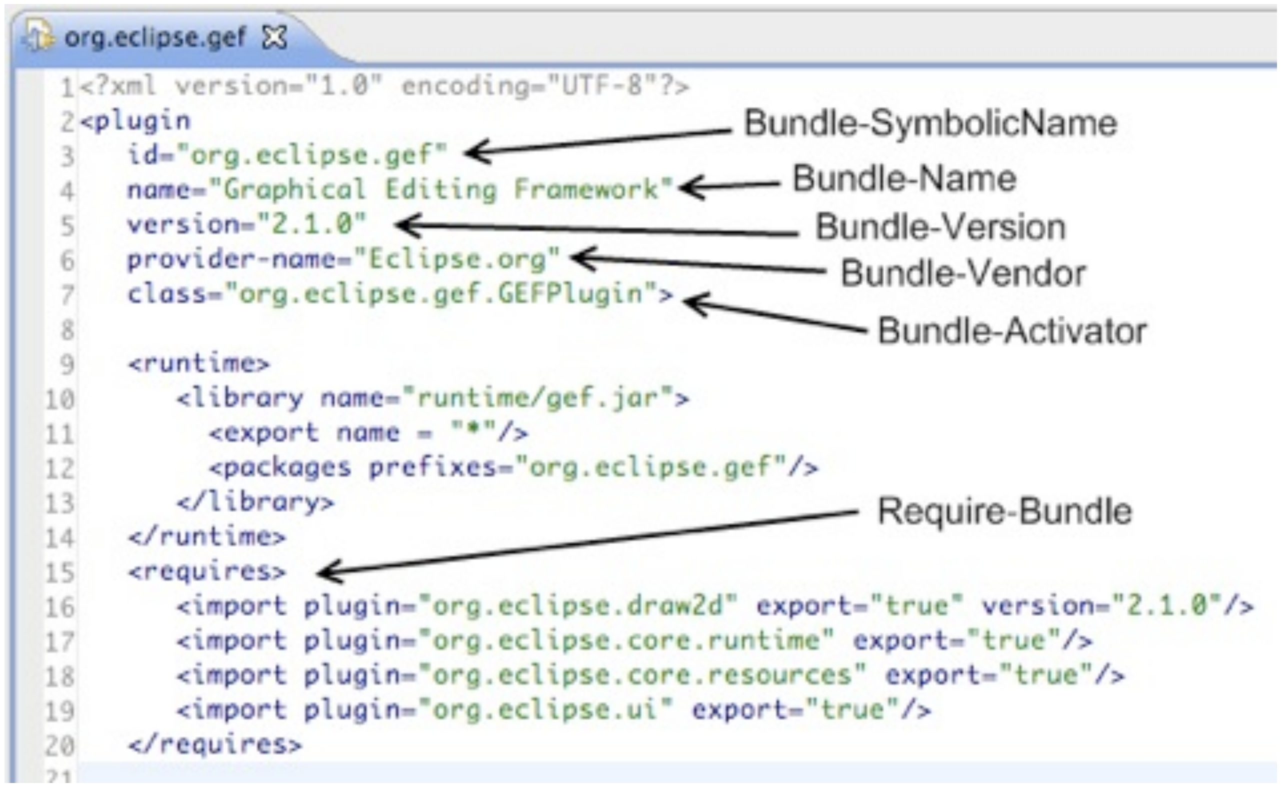
OSGi is a lot more than Eclipse

- Many Eclipse developers oblivious to OSGi topics

- How about starting with a history lesson?

A Blast from the Past...

- Eclipse had it's own non-standard plug-in model
- OSGi and old Eclipse plug-in model were similar



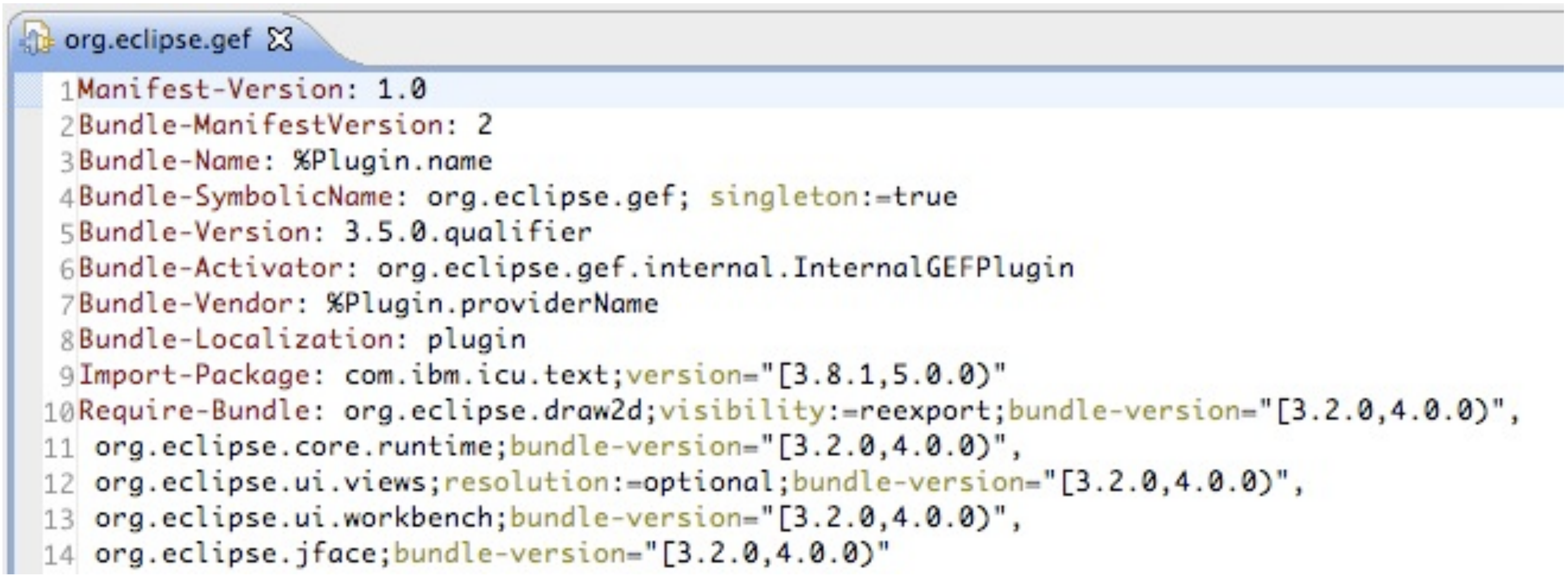
```
<?xml version="1.0" encoding="UTF-8"?>
<plugin
   id="org.eclipse.gef"
   name="Graphical Editing Framework"
   version="2.1.0"
   provider-name="Eclipse.org"
   class="org.eclipse.gef.GEFPlugin">

   <runtime>
      <library name="runtime/gef.jar">
        <export name = "*"/>
        <packages prefixes="org.eclipse.gef"/>
      </library>
   </runtime>
   <requires>
      <import plugin="org.eclipse.draw2d" export="true" version="2.1.0"/>
      <import plugin="org.eclipse.core.runtime" export="true"/>
      <import plugin="org.eclipse.core.resources" export="true"/>
      <import plugin="org.eclipse.ui" export="true"/>
   </requires>
```

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Eclipse + OSGi

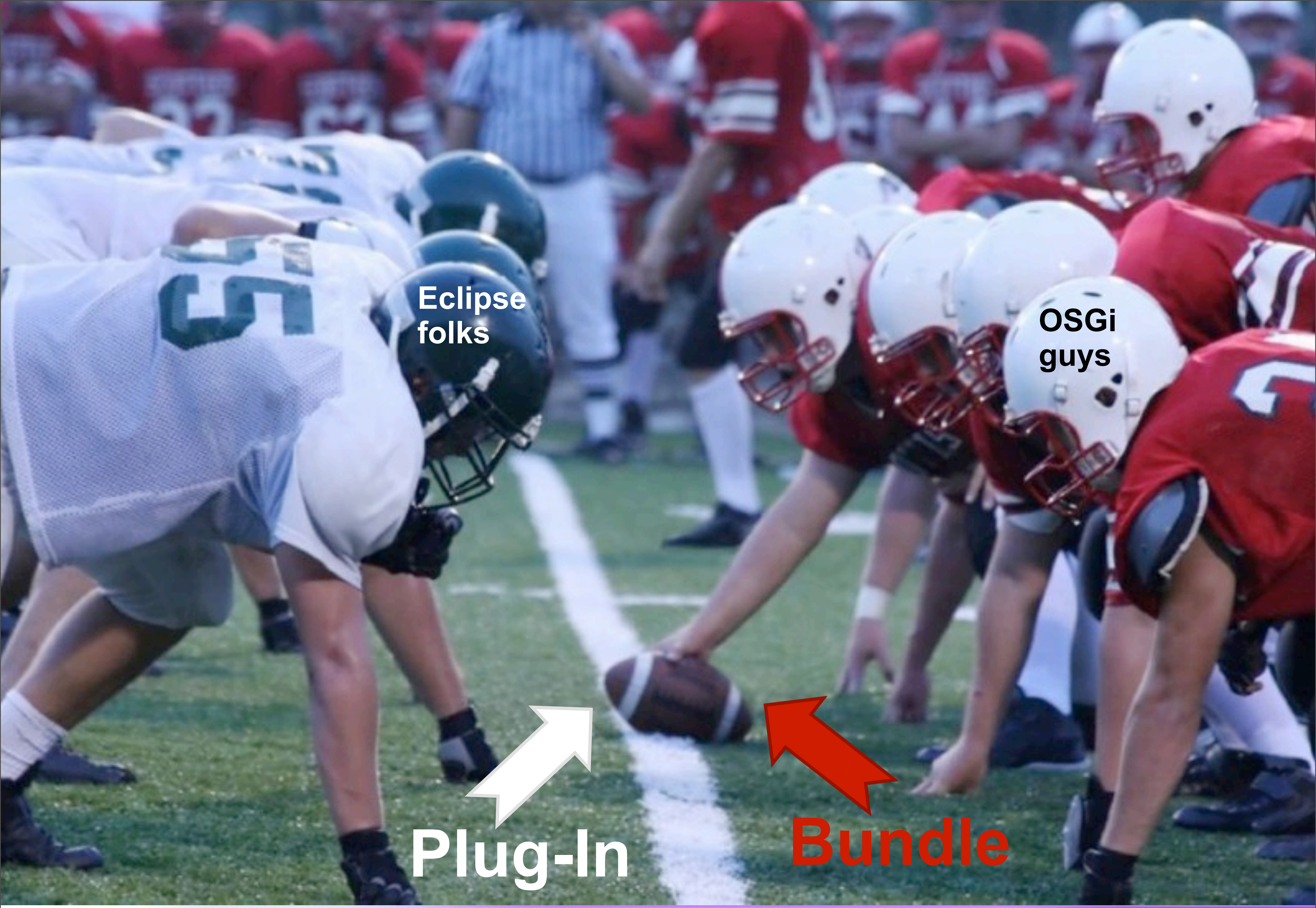
- Eclipse went to OSGi in 3.0*
  - The transition went "*smoothly*"



```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: %Plugin.name
Bundle-SymbolicName: org.eclipse.gef; singleton:=true
Bundle-Version: 3.5.0.qualifier
Bundle-Activator: org.eclipse.gef.internal.InternalGEFPlugin
Bundle-Vendor: %Plugin.providerName
Bundle-Localization: plugin
Import-Package: com.ibm.icu.text;version="[3.8.1,5.0.0)"
Require-Bundle: org.eclipse.draw2d;visibility:=reexport;bundle-version="[3.2.0,4.0.0)",
 org.eclipse.core.runtime;bundle-version="[3.2.0,4.0.0)",
 org.eclipse.ui.views;resolution:=optional;bundle-version="[3.2.0,4.0.0)",
 org.eclipse.ui.workbench;bundle-version="[3.2.0,4.0.0)",
 org.eclipse.jface;bundle-version="[3.2.0,4.0.0)"
```

**http://portal.acm.org/citation.cfm?id=1086616*

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009



© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - **Import-Package vs. Require-Bundle**
  - Dynamic Bundles
  - Extensions and Services
  - Compendium Services
  - OSGi Tooling
- Conclusion

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

The Alien called "Import-Package"

- Eclipse
  - Dependencies are declared using `Require-Bundle`
  - Never heard of `Import-Package`, sounds strange
- OSGi
  - Uuuha, no, please don't use `Require-Bundle` at all
  - Instead, define dependencies using `Import-Package`

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

What is the difference?

- Require-Bundle
  - Imports all exported packages of the bundle, including re-exported and split bundle packages
- Import-Package
  - Import just the package you need

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

What does it mean?

- Require-Bundle
  - Defines a dependency on the produce
  - Broad scope of visibility

- Import-Package
  - Defines a dependency on what you need
  - **Doesn't matter where it comes from!**

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

When to use what?

- Prefer using Import-Package
  - Lighter coupling between bundles
  - Less visibilities
  - Eases refactoring

- Require-Bundle, when necessary
  - Higher coupling between bundles
  - Use only for very specific situations:
    - split packages (same package in different bundles)

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Versioning

- On Bundle level
  - Each Bundle has a version
  - You should set a version range when using require-bundle

- On Package level
  - Packages should also have a version when exported
    - Remember: Import-Package
  - Package imports should have version ranges as well!

- Summary
  - Version everything!

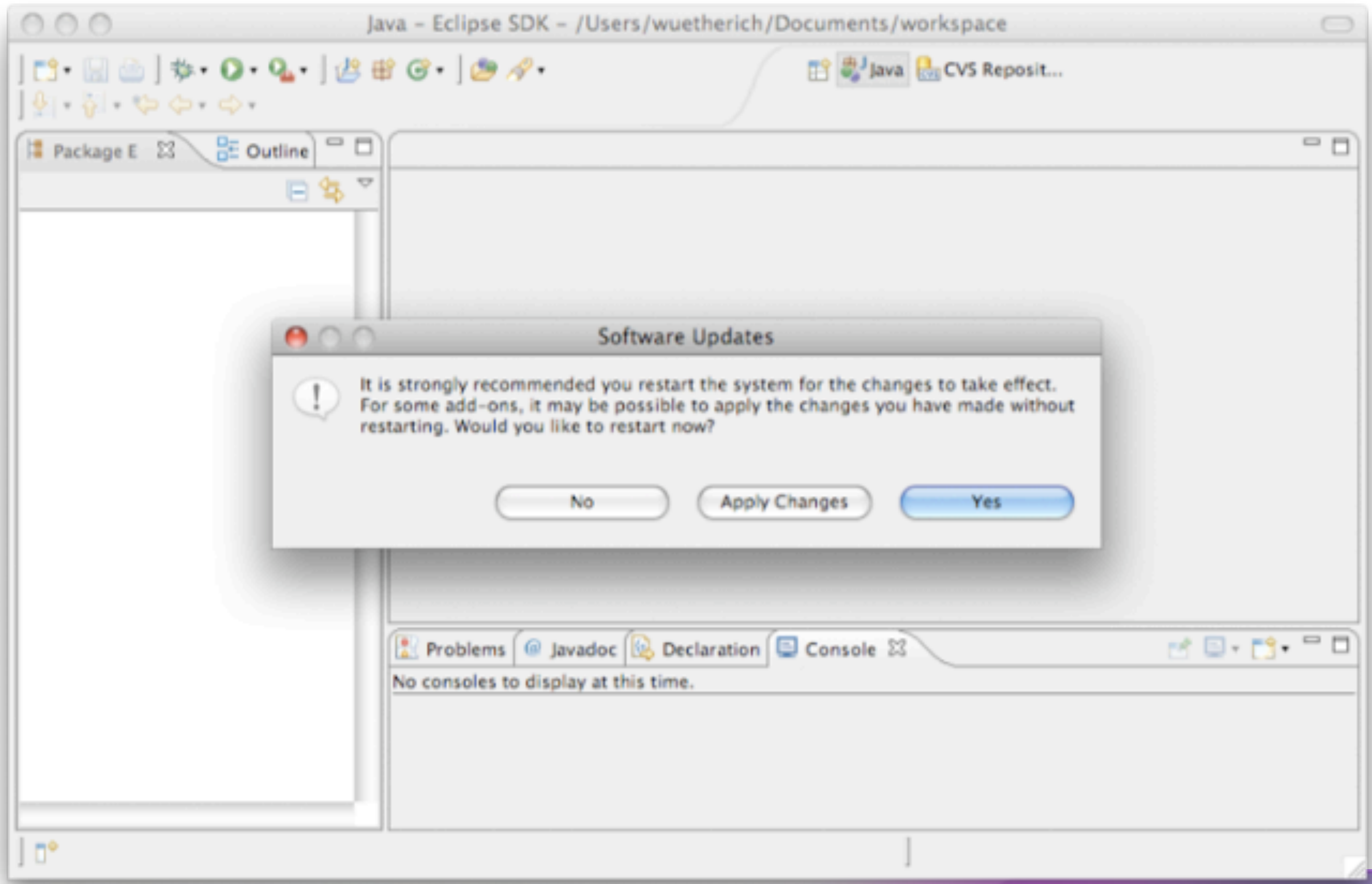
© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - Import-Package vs. Require-Bundle
  - **Dynamic Bundles**
  - Extensions and Services
  - Compendium Services
  - OSGi Tooling
- Conclusion

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Bundles are dynamic? You're kidding…



It is strongly recommended you restart the system for the changes to take effect. For some add-ons, it may be possible to apply the changes you have made without restarting. Would you like to restart now?

No consoles to display at this time.

Dynamics with OSGi

- OSGi allows you to manage bundles at runtime
  - Install
  - Update
  - Uninstall

- But there is no magic behind the scenes
  - nothing is changed automatically
  - objects stay the same
  - references remain valid

- This means you need to cleanup after yourself so the GC can help you!

Updating a bundle at runtime means...

- Dependent bundles (with wires to the updated bundle via `Require-Bundle` or `Import-Package`) are stopped and re-started

- The consequence:
  - updating a bundle might cause the system to “restart”
  - this is not what I associate with “cool dynamics”

➔ When programming anticipate OSGi’s dynamics

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Think about dependencies

- Less is more...!!!
    - Less dependencies
    - DIP (Dependency Inversion Principle)

- Think more about APIs
    - API in separate bundle
    - dependency only on API bundle
    - implementation can change

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - Import-Package vs. Require-Bundle
  - Dynamic Bundles
  - **Extensions and Services**
  - Compendium Services
  - OSGi Tooling
- Conclusion

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

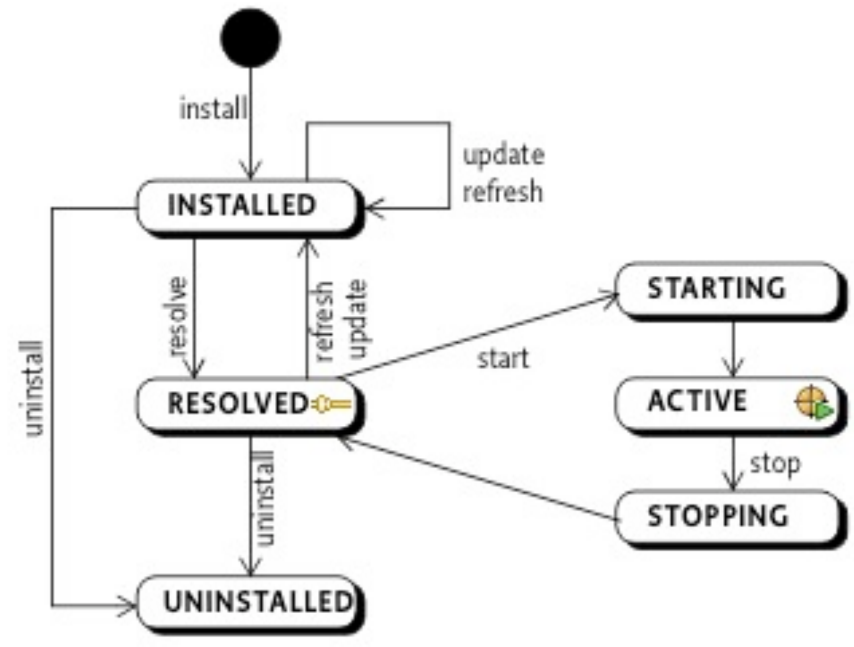
OSGi Services – another alien on planet…

- Services? SOA? Oh no... please no buzzwords!!!
- Something like Extension Points? Hm... What also do we need..?!?
- **Hey, wait!**
    - OSGi Services are *the key feature* to build *modular* and *dynamic* apps

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

OSGi Services – your best friend

- OSGi Service providers:
  - implement an interface and register an implementation
- OSGi Service consumers:
  - lookup a service via the interface
- Bound to the bundle active state
  - Extensions are available already in resolved state



They come and go

- A bundle is started:
  - services are registered
  - and available from that on
- A bundle is stopped:
  - services are unregistered
  - no longer available
- OSGi services are dynamic by definition!!!
  - + dozens of techniques to deal with these dynamics

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

OSGi Services – versioned contracts

- the service interface is the contract
  - many consumers possible
  - many producers possible

- this contract is versioned
  - multiple versions of service might be available
  - you get only those that matches your dependencies

➔You cannot get that with Extension points
- There you always get the latest version

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

OSGi Services – declarative and lazy

- OSGi services are bound to the active state
  - they need class loading to happen
  - they need objects to be created
- There are declarative approaches for OSGi services
  - OSGi Declarative Services
  - Spring Dynamic Modules (aka Blueprint Service)
  - iPOJO

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

When to use what?

- OSGi Services:
    - Dependencies between bundles
    - Dynamics
    - Looser coupling
    - "I provide a service for anybody out there"
    - "I need a service and don't care who delivers it"

- Extension-Registry:
    - UI contributions (too small for OSGi services)
    - Non-code contributions
    - "I open up myself for extensions that I don't know upfront"
    - If you have tons of thousand of extensions

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Say goodbye to your buddies

- Forget everything about buddy loading
  - its Equinox specific

- Think about other ways to go
  - class registering
  - OSGi services
  - dynamic import

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Be aware of special headers…

- Forget about:
  - Eclipse-BuddyPolicy
  - Eclipse-PatchFragment
  - Eclipse-SourceBundle
  - Eclipse-…

  ➔Otherwise you are tied to Equinox

- Tip: PAX Runner to test against multiple frameworks
  - http://wiki.ops4j.org/display/ops4j/Pax+Runner

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - Import-Package vs. Require-Bundle
  - Dynamic Bundles
  - Extensions and Services
  - **Compendium Services**
  - OSGi Tooling
- Conclusion

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Compendium Services

- OSGi has spec'd 20+ services

- LogService
- EventAdmin
- HttpService
- Declarative Services
- Configuration Admin

OSGi Service Platform
Service Compendium
The OSGi Alliance

Release 4, Version 4.1
April 2007

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

LogService

- A general purpose message logger (20kb)
- LogService
  - Log message, level, exception, service ref, bundle
- LogReaderService
  - Retrieve current or previous log entries
- Note: ExtendedLogService (bug 260672)
  - named loggers
  - extended log entry (e.g., thread id)
  - filters for log listeners

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

EventAdmin

- An inter-bundle pub-sub system (30kb)

- EventAdmin
  - publish events synchronously and asynchronously
    - `postEvent(new Event("com/acme/timer", time ));`
    - `sendEvent(new Event("com/acme/timer", time ));`

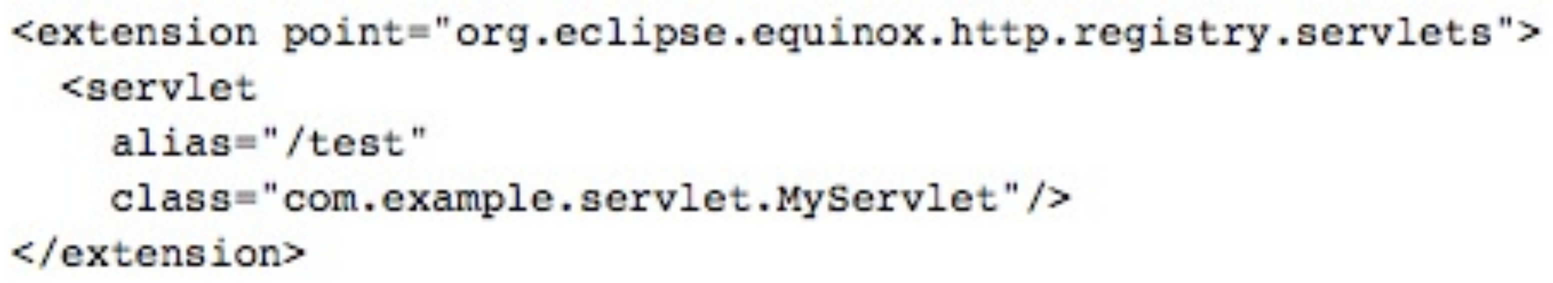
- EventHandler
  - handle events based on topics
    - `handleEvent(Event event)`

- Event
  - has topic and properties as attributes

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

HttpService

- A way to register servlets and resources
- HttpService
  - Register servlets and resources
- Http Registry (org.eclipse.equinox.http.registry)

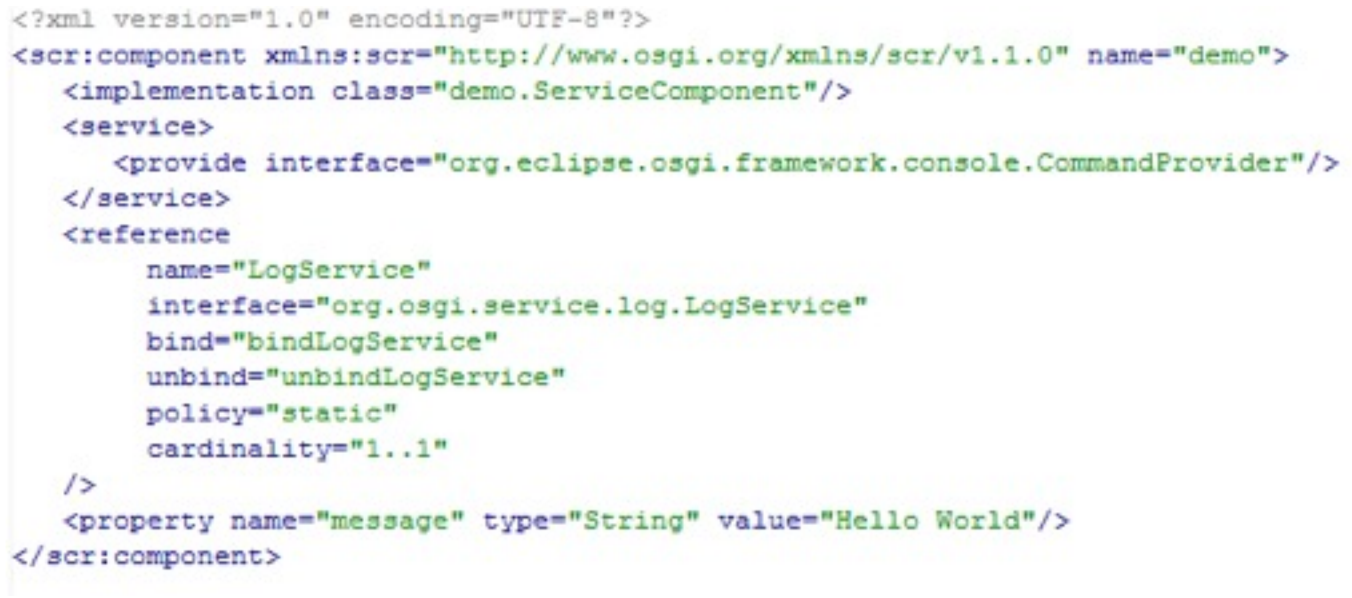


```
<extension point="org.eclipse.equinox.http.registry.servlets">
  <servlet
    alias="/test"
    class="com.example.servlet.MyServlet"/>
</extension>
```

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Declarative Services

- A declarative model for publishing, finding and binding to OSGi services (150kb)
- ServiceTracker's – the programmatic way to get a service – suck

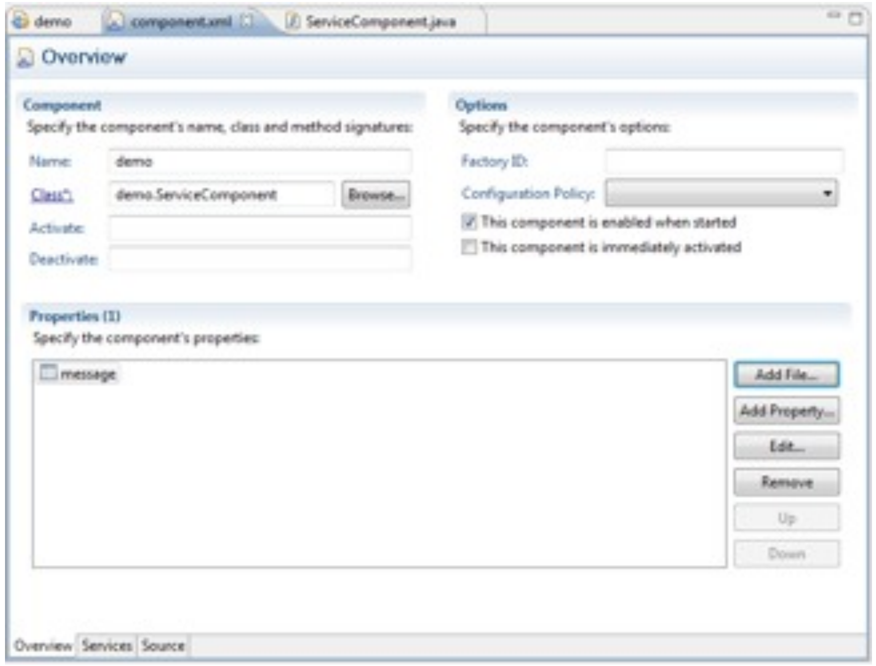


```xml
<?xml version="1.0" encoding="UTF-8"?>
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0" name="demo">
   <implementation class="demo.ServiceComponent"/>
   <service>
      <provide interface="org.eclipse.osgi.framework.console.CommandProvider"/>
   </service>
   <reference
        name="LogService"
        interface="org.osgi.service.log.LogService"
        bind="bindLogService"
        unbind="unbindLogService"
        policy="static"
        cardinality="1..1"
   />
   <property name="message" type="String" value="Hello World"/>
</scr:component>
```

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Declarative Services Tooling

- Graphical Editor
- Validation
- Source Editing

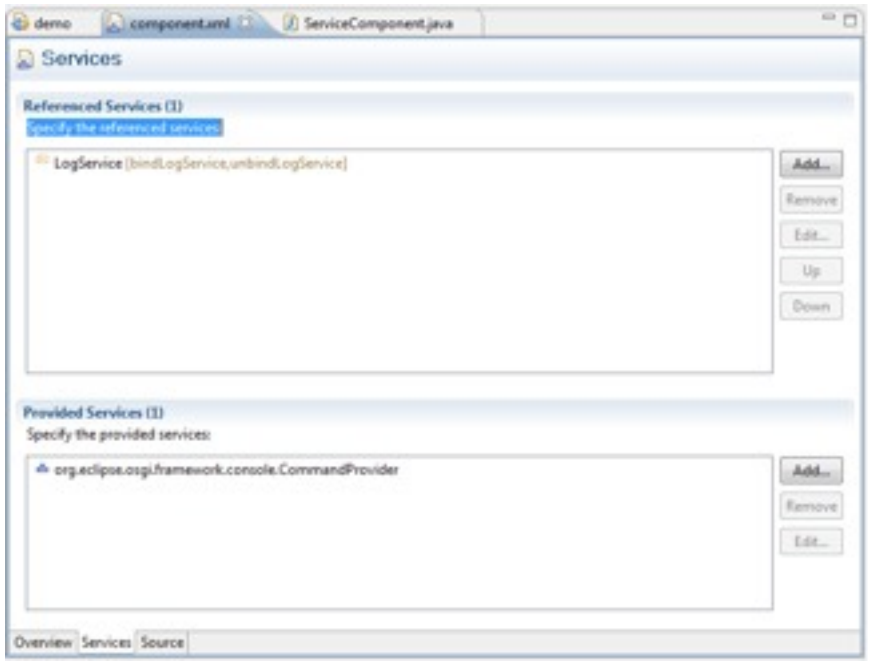


Overview

Component
Specify the component's name, class and method signatures:
Name: demo
Class*: demo.ServiceComponent
Activate:
Deactivate:

Options
Specify the component's options:
Factory ID:
Configuration Policy:
☑ This component is enabled when started
☐ This component is immediately activated

Properties (1)
Specify the component's properties:
message



Services

Referenced Services (1)
Specify the referenced services
LogService [bindLogService,unbindLogService]

Provided Services (1)
Specify the provided services:
org.eclipse.osgi.framework.console.CommandProvider

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

ConfigAdmin

- A service to configure components (bundles)
  - A configuration is a list of key-value pairs
- The configuration admin service persists and distributes these configurations to interested parties
- Components to be configured register a ManagedService
- To apply several configurations of the same kind you could use a ManagedServiceFactory

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Overview

- Introduction
- Topics
  - Import-Package vs. Require-Bundle
  - Dynamic Bundles
  - Extensions and Services
  - Compendium Services
  - **OSGi Tooling**
- Conclusion

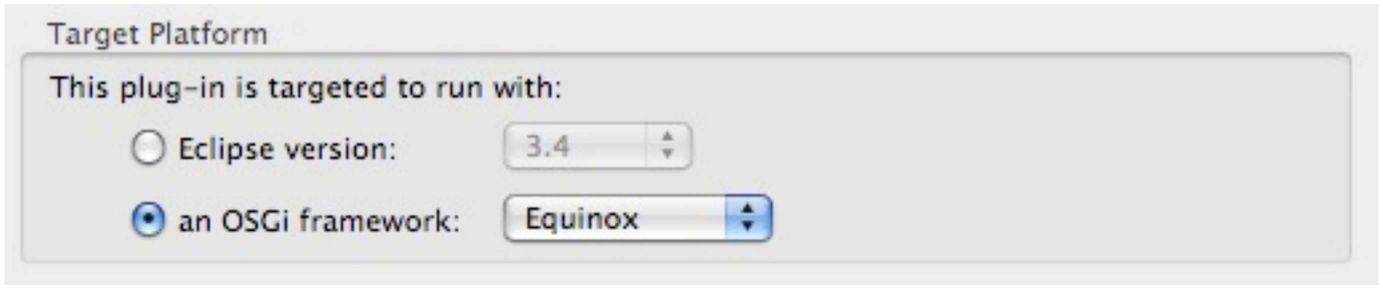
© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

PDE

- Eclipse has been tooling OSGi forever with PDE
  - **Plug-ins == Bundles! Blugins?**

- PDE Tools:
  - Bundles
  - Fragments
  - Declarative Services

- New Plug-in Project wizard has OSGi love



Target Platform

This plug-in is targeted to run with:

- Eclipse version: 3.4
- [✓ an OSGi framework:] Equinox

BND

- Bundle Tool (BND)
  - creates and diagnoses OSGi bundles
  - Maven, Eclipse and Ant integration
  - http://www.aqute.biz/Code/Bnd

- Relies on specification (.bnd file) + classpath

```
Export-Package: aQute.service.*
Import-Package: javax.servlet.http;version="[2,3)", *
```

- Generates bundle artifacts like manifests

- Useful for converting third party libs to bundles

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Sigil

- Provides OSGi Tooling
    - http://sigil.codecauldron.org/
    - driven by sigil.properties file
    - BND used under the covers

- bundles fetched from repositories
    - based on your Import-Package statements

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009

Thank you for your attention!

- Questions and feedback welcome!

Chris Aniszczyk: zx@eclipsesource.com
Bernd Kolb: bernd.kolb@sap.com
Martin Lippert: lippert@acm.org

© 2009 by C. Aniszczyk, B. Kolb, M. Lippert; made available under the EPL v1.0 | March 24th, 2009