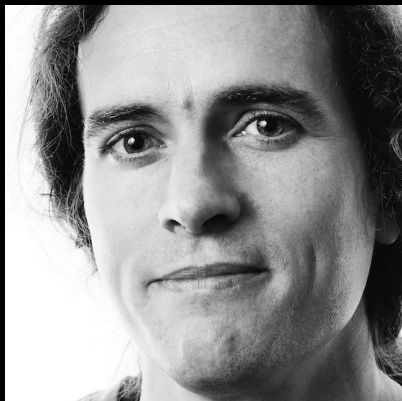




# Tutorial

Florian Waibel, Johannes Eickhold,  
Markus Knauer

# Who we are



**Florian**



**Johannes**



**Markus**

# Introduction Round

1. Who am I?
2. What is my EclipseRT background?
3. What is my motivation to visit the tutorial?

# Roadmap

1. Install Virgo Server and IDE + Tooling
2. Walkthrough greenpages
3. “Hands On”

# Installing Prerequisites

1. Virgo Server for Apache Tomcat
2. Spring Tool Suite™ with Virgo Tooling



+



eclipse

# Installing Virgo (3.6.2.RELEASE)

Goto <https://www.eclipse.org/virgo/download/>

```
unzip virgo-tomcat-server-3.6.2.RELEASE.zip
```

**USB stick:**

```
/virgo-tomcat-server-3.6.2.RELEASE.zip
```

# Installation of 3rd party libraries

## USB stick:

```
cp /USB Stick/par-provided/*.jar ${VIRGO_HOME}/repository/usr
```

# Install Eclipse and Virgo Tooling

## Install Eclipse

- Eclipse Kepler (4.3.2) SR2 Packages
- OR
- Spring Tool Suite™ (3.4.0.RELEASE)

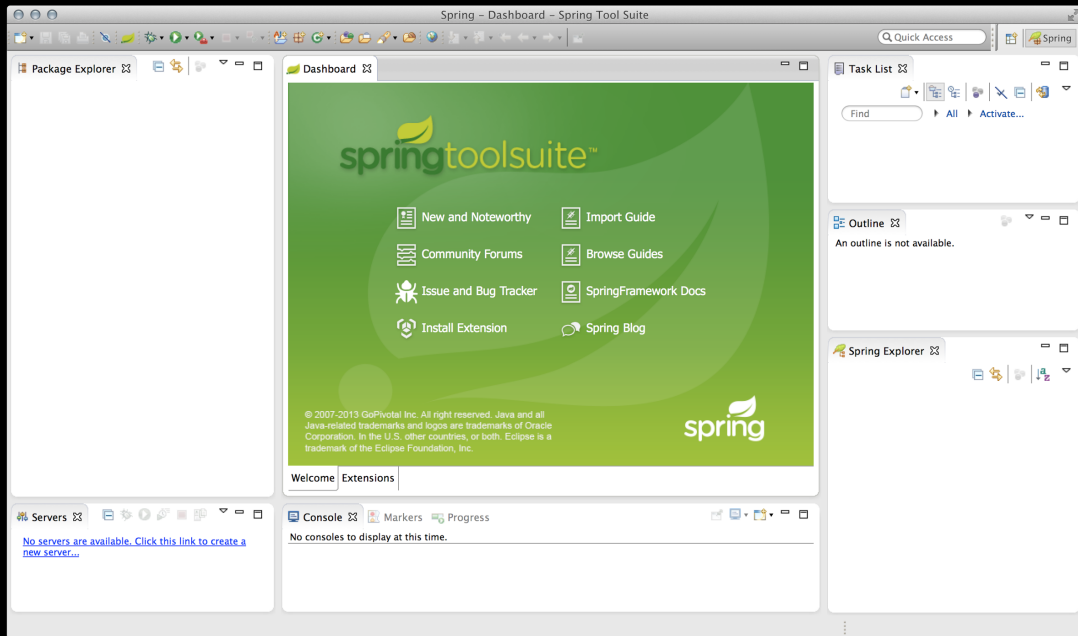
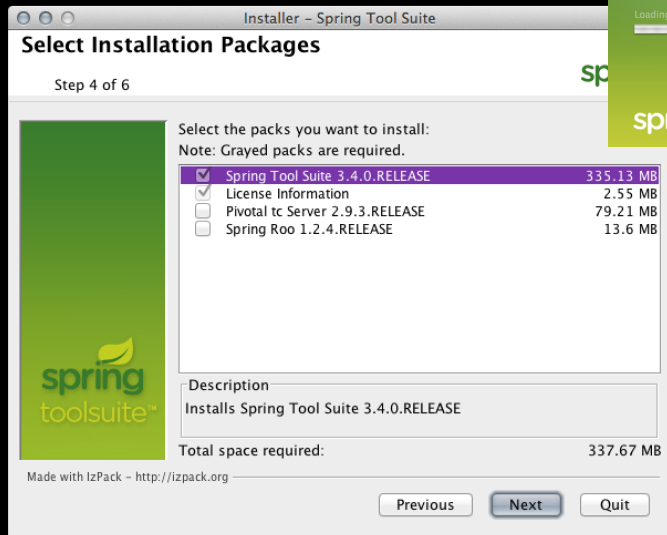


**USB stick:**

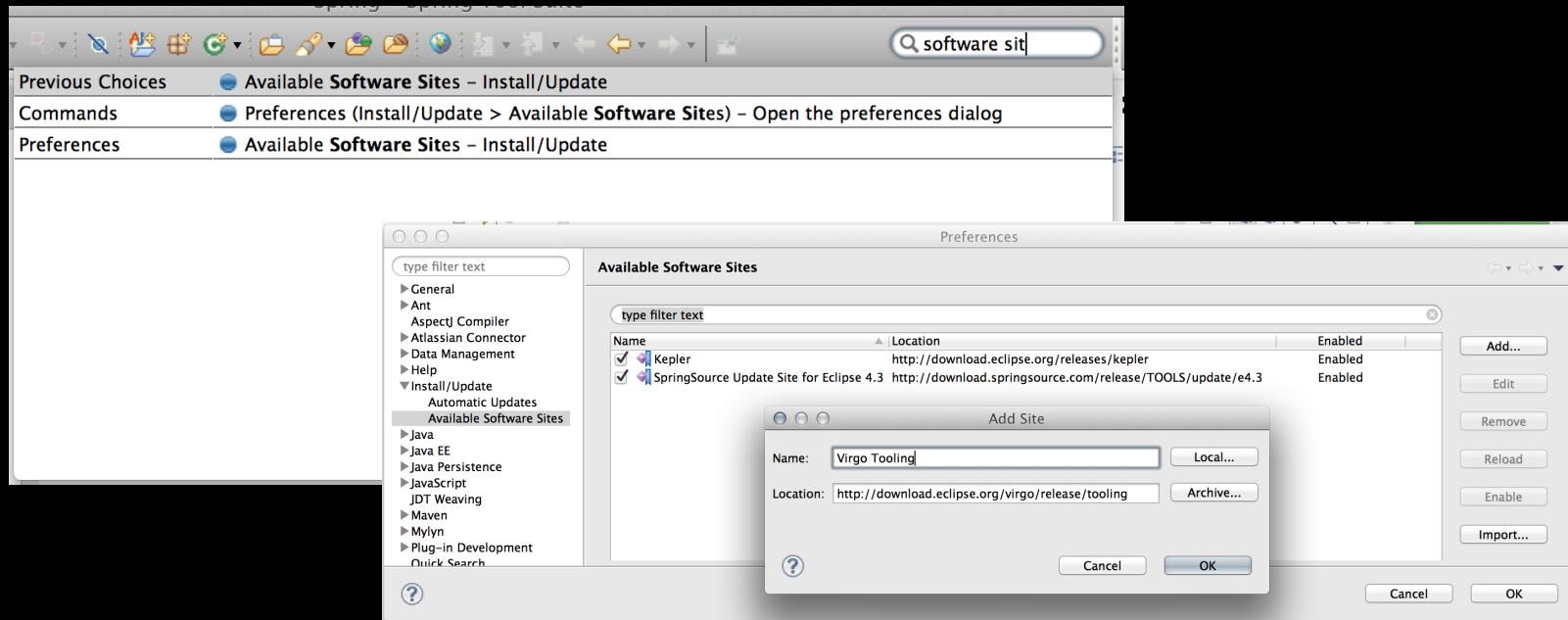
```
unzip <platform>/spring-tool-suite...
```



# Install STS



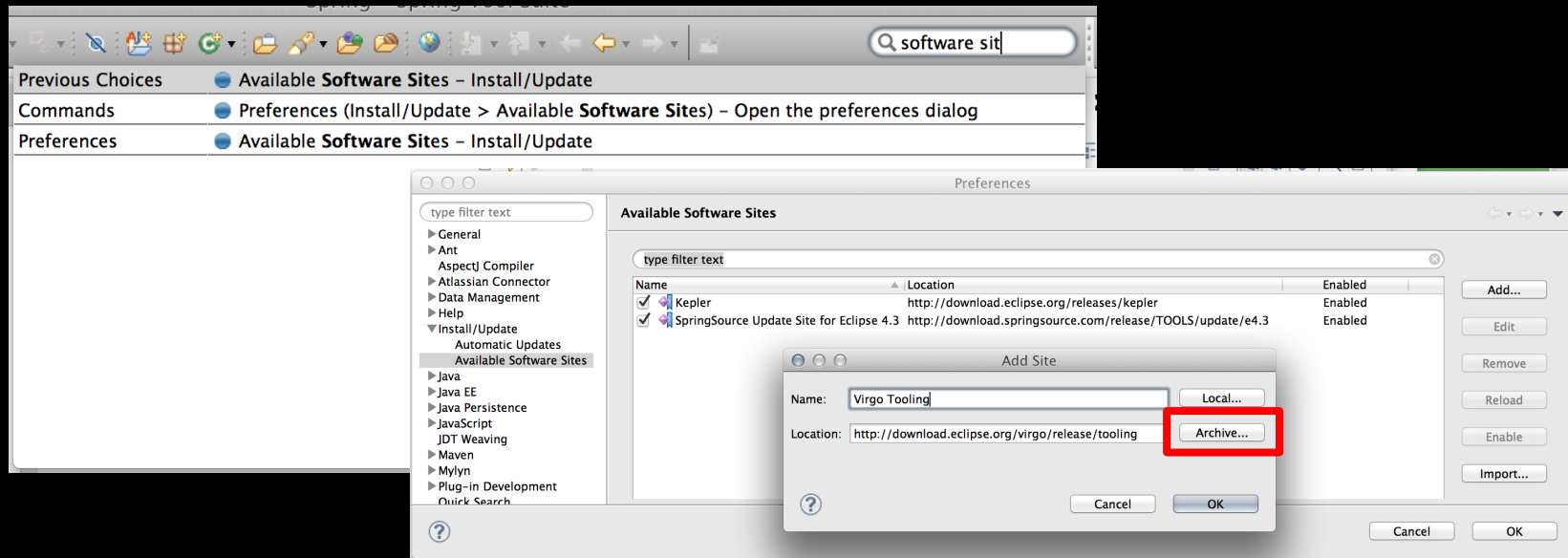
# <ctrl>+<3> software site



Update Site:

<http://download.eclipse.org/virgo/release/tooling>

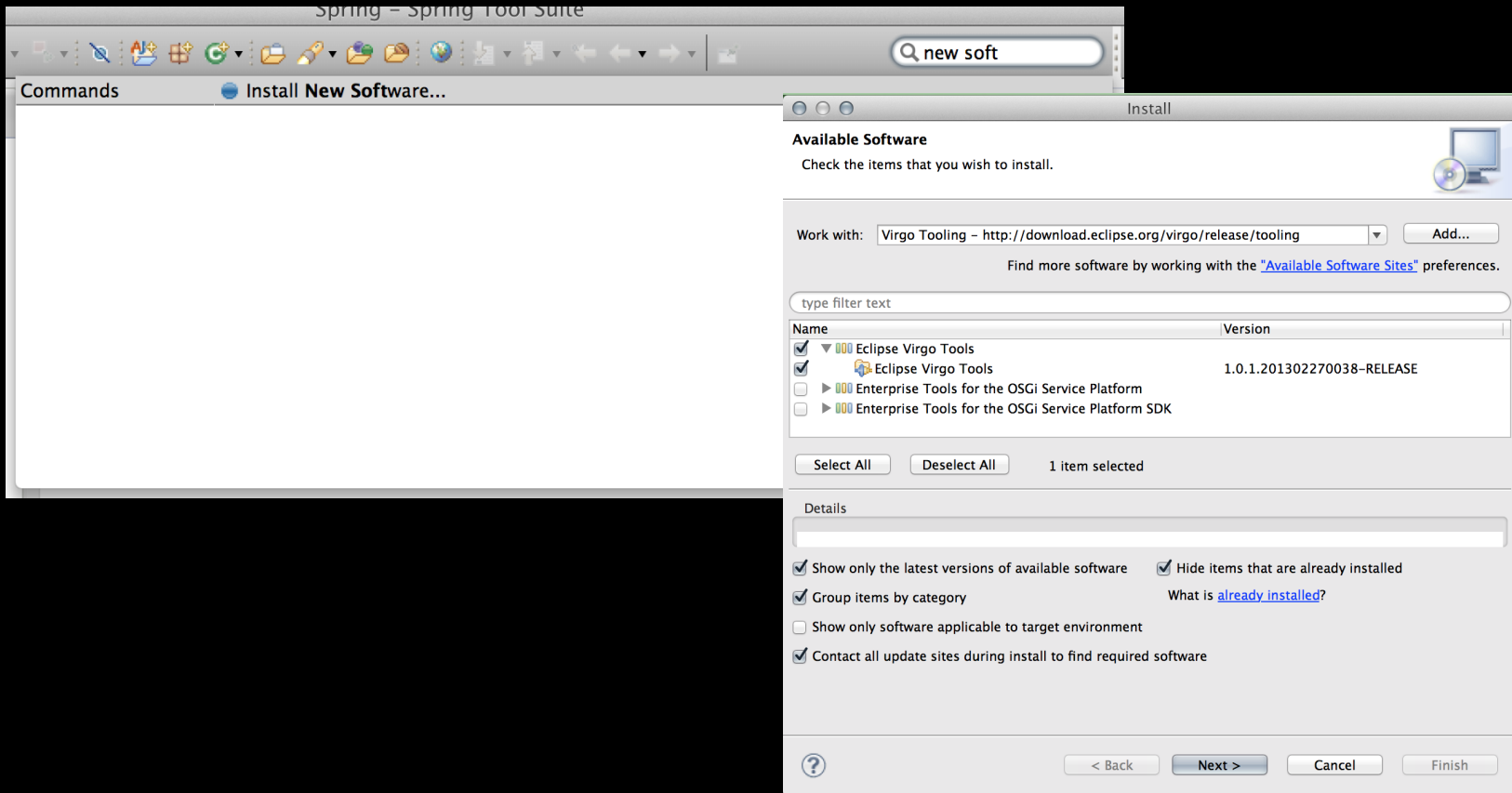
# <ctrl>+<3> software site



**USB stick:**

use `virgo.ide-1.0.1...zip`

# <ctrl>+<3> new software



# <ctrl>+<3> new server

Spring – Dashboard – Spring Tool Suite

new server

New Server – Define a new server

**Define a New Server**  
Choose the type of server to create

[Download additional server adapters](#)

Select the server type:  
virgo

▼ EclipseRT  
Virgo Runtime

Publishes and manages configurations for a Virgo Web Server or Kernel.

Server's host name: localhost

Server name: Virgo Runtime at localhost

**Virgo Web Server**  
Valid installation: Virgo Server for Apache Tomcat v3.6.2.RELEASE.

Name: Virgo Runtime

Server installation directory: jects/virgo-tutorial/tools/virgo-tomcat-server-3.6.2.RELEASE Browse...

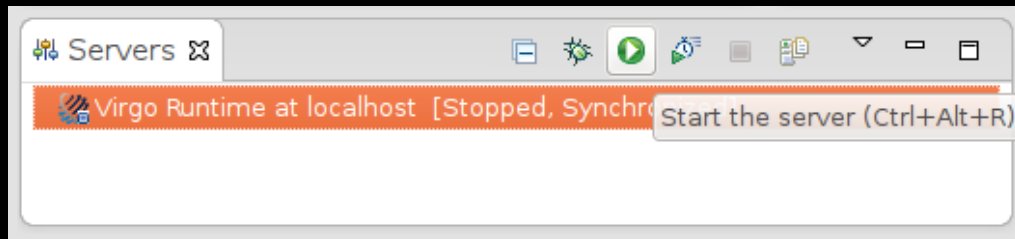
JRE: Workbench default JRE Installed JREs...

Press '⌘3' to show all matches

# Enable OSGi console

```
${VIRGO_HOME}/repository/ext/osgi.console.properties
```

```
telnet.enabled=true  
telnet.port=2501  
telnet.host=localhost  
ssh.enabled=false  
ssh.port=2502  
ssh.host=localhost
```



```
$> telnet localhost 2501  
Trying 127.0.0.1...  
Connected to localhost.  
Escape character is '^]'.  
osgi>
```

# Virgo Alive



Admin Console

localhost:8080

Username: **admin**

Password: **springsource**

# Virgo Admin Console



[Overview](#) [Artifacts](#) [Repositories](#) [Wirings](#) [Dumps](#) [Configurations](#) [Logging](#)

## Host Information

Congratulations on installing Virgo. This is the Web console provided with Virgo to allow management of Virgo instances.

## Documentation

Documentation is available on-line for Virgo:

- [Virgo Documentation](#)
- [Virgo Samples](#)
- [Virgo FAQ](#)

## Host Properties

Name	Value
OSGi Runtime	Apache Tomcat/7.0.35 on Eclipse Virgo 3.6.2.RELEASE
Virtual Machine	1.7.0_51 - Java HotSpot(TM) 64-Bit Server VM 24.51-b03 (Oracle Corporation)
Operating System	Mac OS X 10.9.2 (x86_64)

Region: org.eclipse.virgo.region.user

- + org.eclipse.virgo.web
- + org.eclipse.virgo.apps.repository
- + osgi.console.ssh
- osgi.console.telnet

Property	Value
port	2501
enabled	true
host	localhost
service.pid	osgi.console.telnet



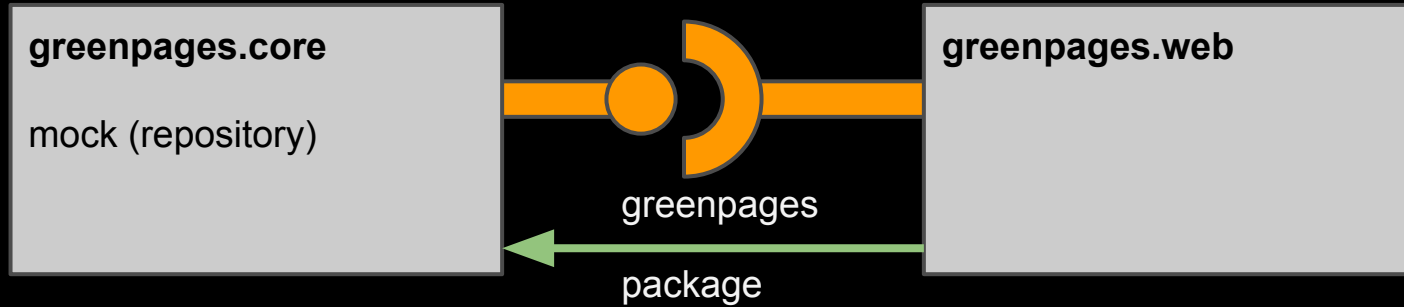
## Virgo Web Server - Greenpages sample

# Virgo demo used in official documentation

“Despite its simplicity, GreenPages is designed to demonstrate many different Virgo features and to act as a template from which other modular applications can be built.”



# Architectural Overview 1/2



OSGi service export

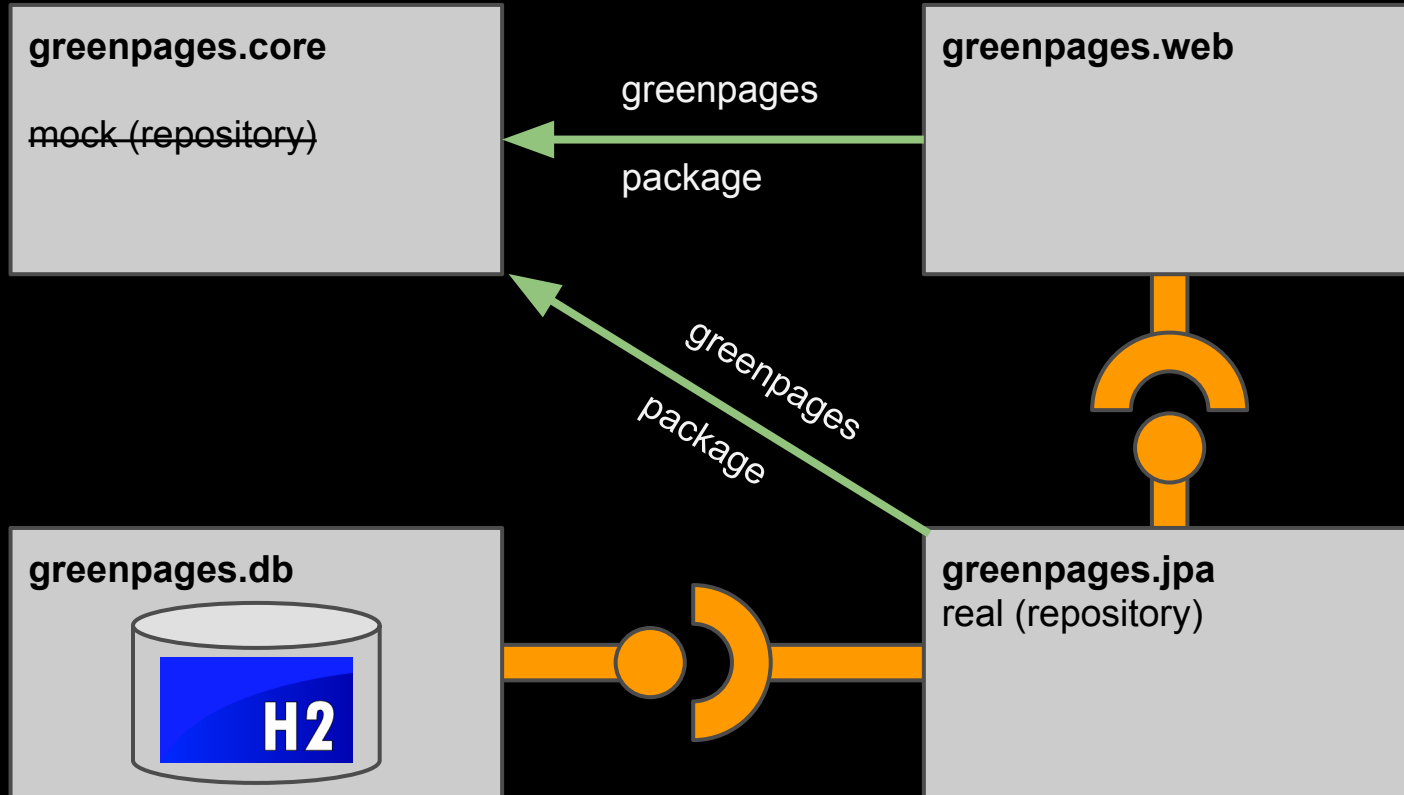


OSGi service import

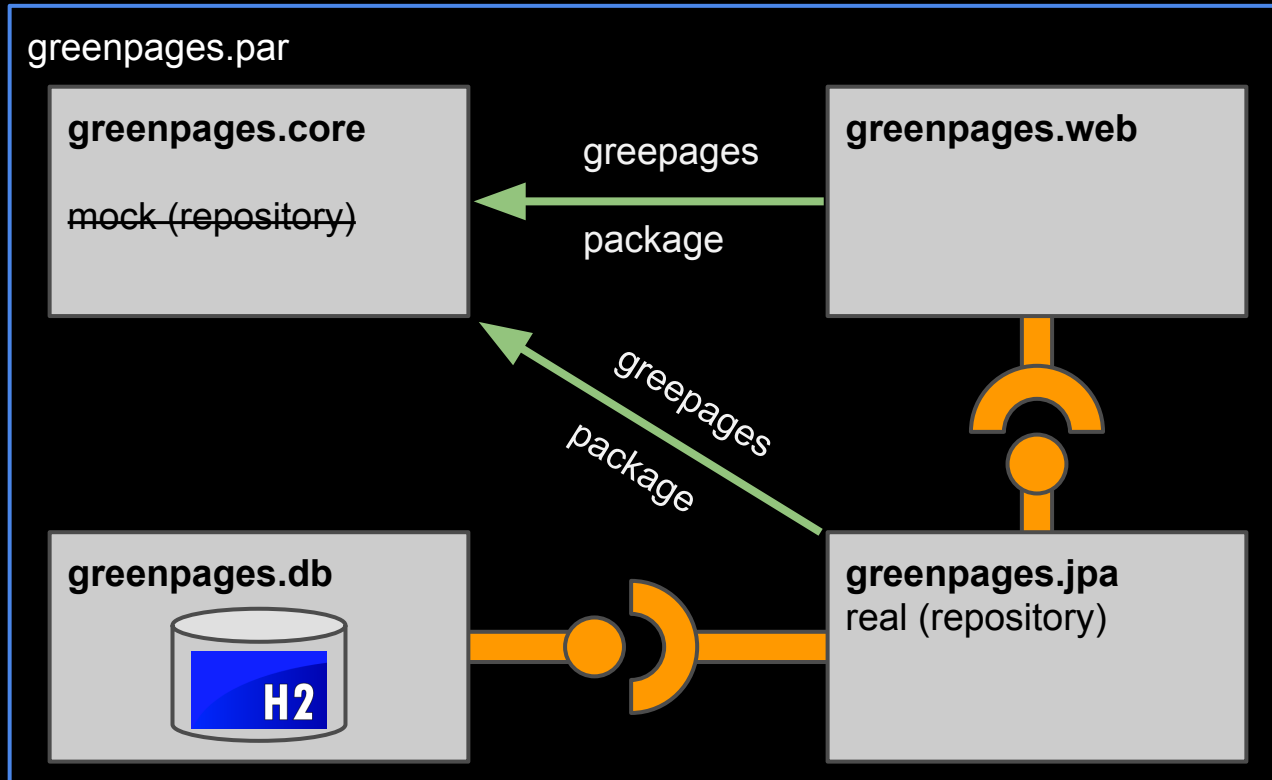


package usage

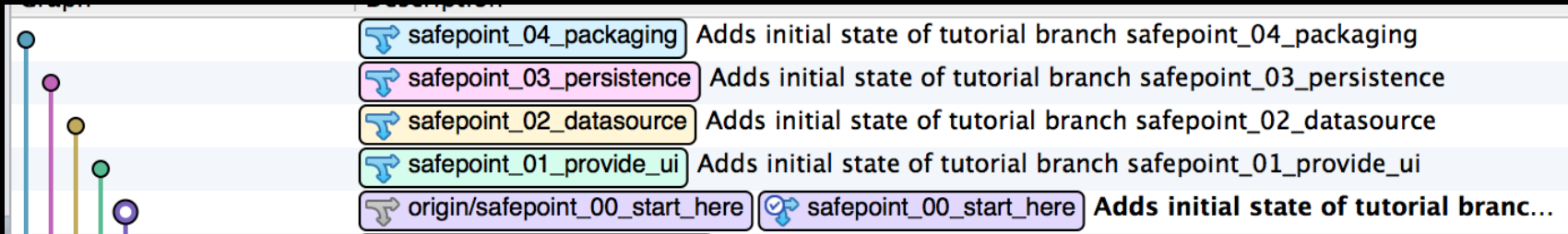
# Architectural Overview 2/2



# Packaging



# Tutorial as Branches



## During the Tutorial YOU do:

1. Try to solve the steps (Hint: Look for TODO x.y)
2. `git diff safepoint_0x_<branch_name>`
3. `git checkout safepoint_0x_<branch_name>`

**Ready, Steady, ...**



**Vir...Go!**

# Kickstart

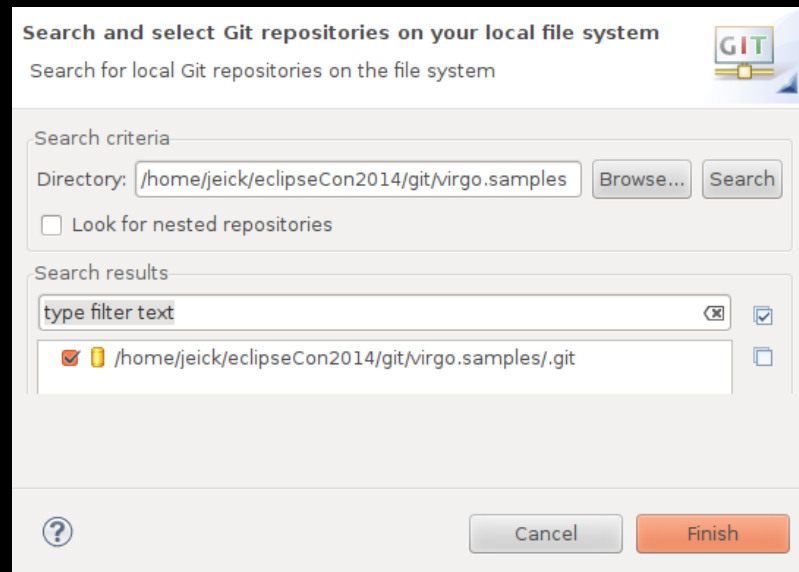
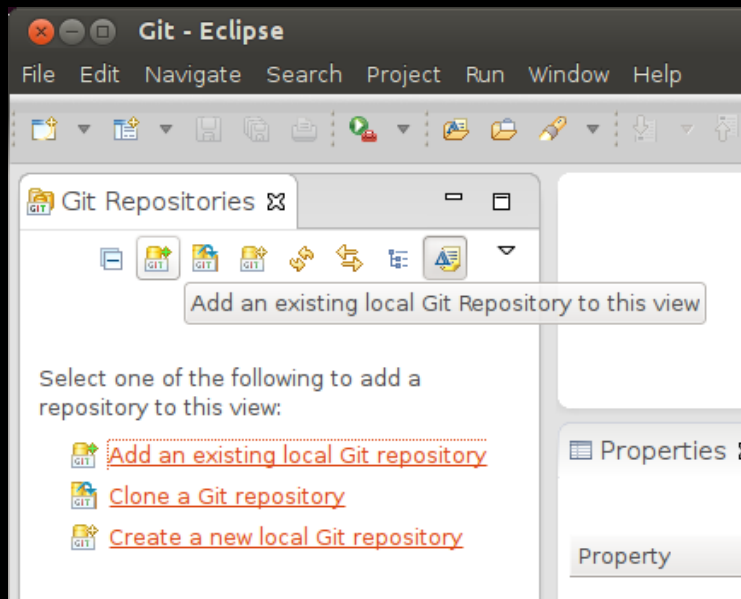


```
git clone https://github.com/eclipse/virgo.  
samples
```

**USB Stick (Get local copy of the Git repository)**

```
unzip /usb/virgo.samples_GITrepo.zip -d <your working dir>/git/
```

# Import Git repo





# Next Goal: Export first OSGi service



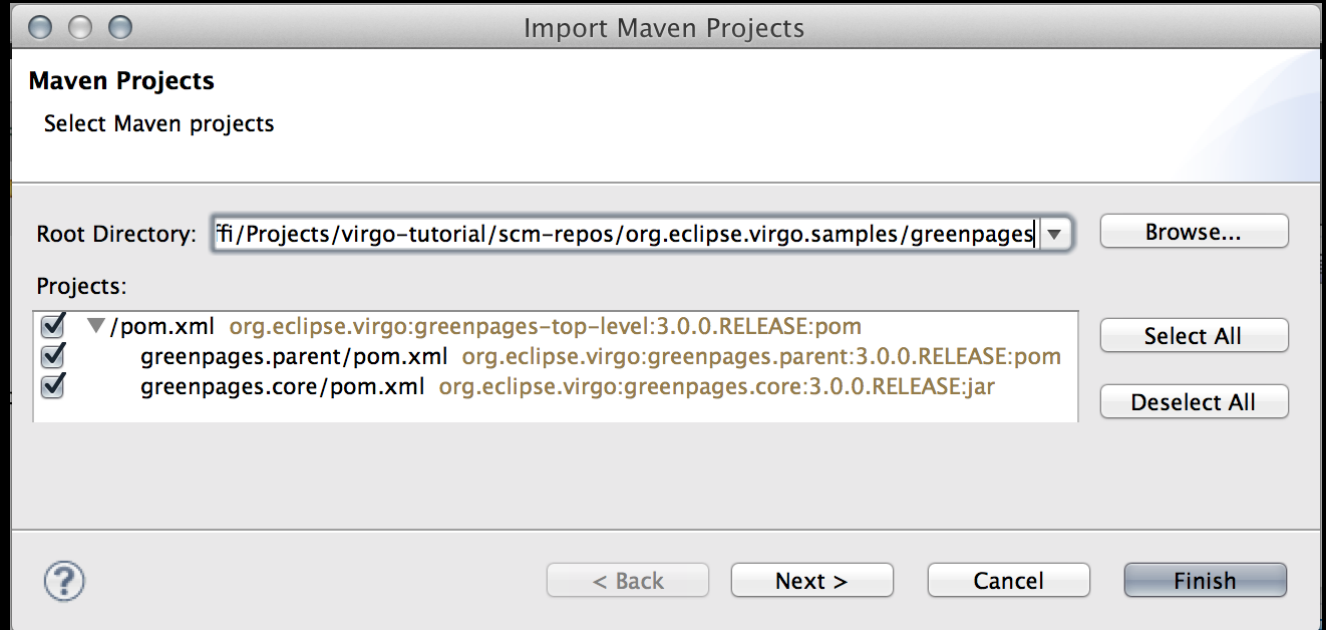
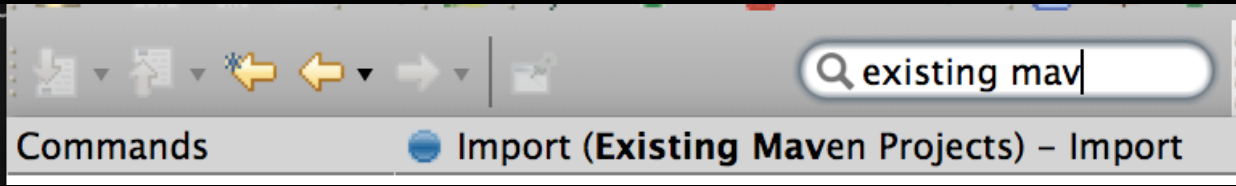
**Idea:** Export mock **OSGi service** to allow other developers on the team to start integrating with your module

# Safepoint 0 - Start Here

```
git checkout safepoint_00_start_here
```

Import... -> Existing Maven Projects

# 0.1 Import existing **Maven** project



## 0.2 Activate the mock OSGi service

Export Spring bean “**directory**” as OSGi service

```
META-INF/spring/osgi-context.xml
```

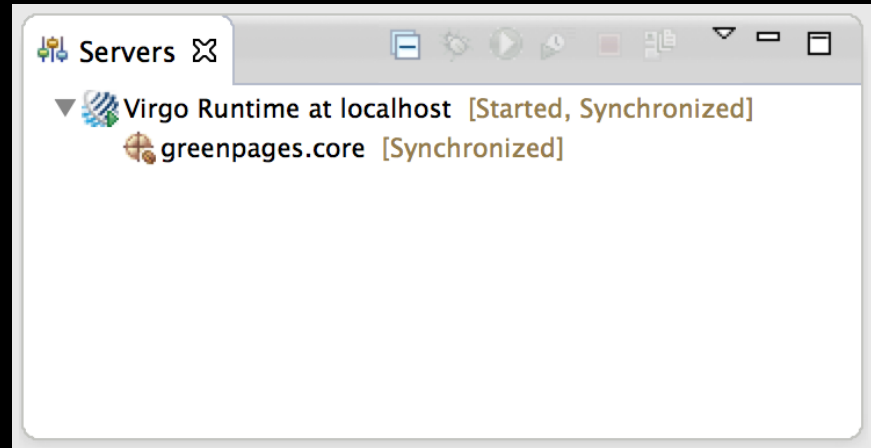
```
<beans xmlns=...>  
  <osgi:service interface="greenpages.Directory"  
    ref="directory"/>  
</beans>
```

## 0.3 Deploy the mock bundle

Add **OSGi Bundle Project Nature**



Deploy the bundle  
(drag'n'drop)



## 0.4 Check if the service is running

```
osgi> services *Directory
{greenpages.Directory}={org.eclipse.gemini.blueprint.bean.name=directory,
org.springframework.osgi.bean.name=directory, osgi.service.blueprint.
compname=directory, Bundle-SymbolicName=greenpages.core, Bundle-Version=3.
0.0.RELEASE, service.id=260}
  "Registered by bundle:" greenpages.core_3.0.0.RELEASE [140]
```

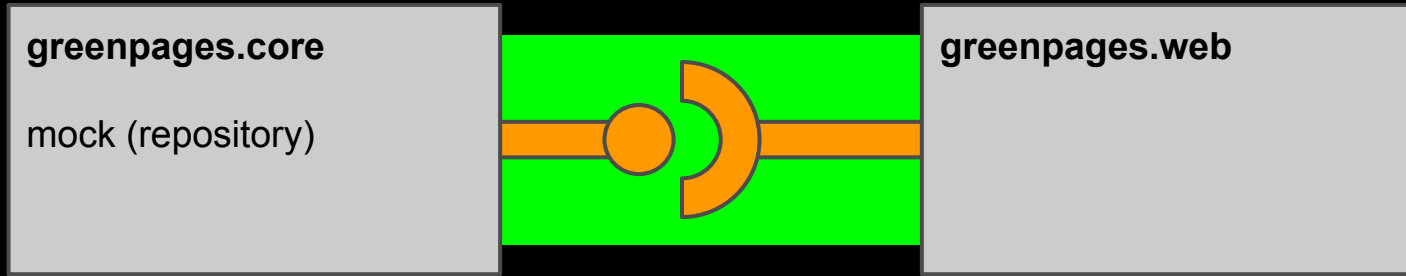
# Safepoint 1 - First OSGi service

```
git diff safepoint_01_provide_ui
```

Prepare for next goal:

```
git checkout safepoint_01_provide_ui  
import -> existing Maven Project greenpages.web
```

# Next Goal: Provide UI



- Wire exported OSGi service Directory
- Make web application available



# 1.0 Ping web application

Set OSGi aware contextClass in `web.xml`

```
<context-param>  
  <param-name>contextClass</param-name>  
  <param-value>org.eclipse.virgo.web.dm.ServerOsgiBundleXmlWebApplicationContext</param-value>  
</context-param>
```

Add OSGi Web-ContextPath in `pom.xml`

```
<Web-ContextPath>greenpages</Web-ContextPath>
```

Visit GreenPages application

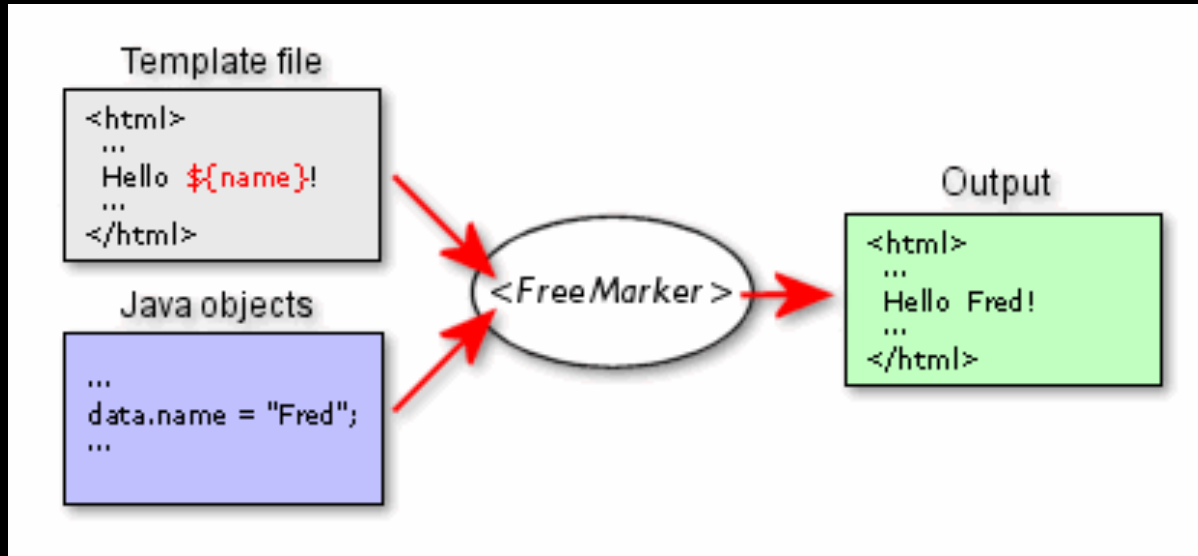
<http://localhost:8080/greenpages/>

# 1.1 First contact



<http://localhost:8080/greenpages/>

# 1.2 Web Presentation Layer



`<FreeMarker>` <http://freemarker.org/>

# 1.3 Import OSGi service “Directory”

Import OSGi service and publish as “**directory**” bean

WEB-INF/applicationContext.xml

```
<beans xmlns=...>
  <osgi:reference interface="greenpages.Directory"
    id="directory"/>
</beans>
```

## 1.4 Let Virgo autowire OSGi Service

**@Autowired** field directory in  
GreenPagesController

## 1.5 Use autowired service

Implement web endpoints:

```
@RequestMapping("/search.htm")
```

```
@RequestMapping("/entry.htm")
```

# Safepoint 2 - Provide UI

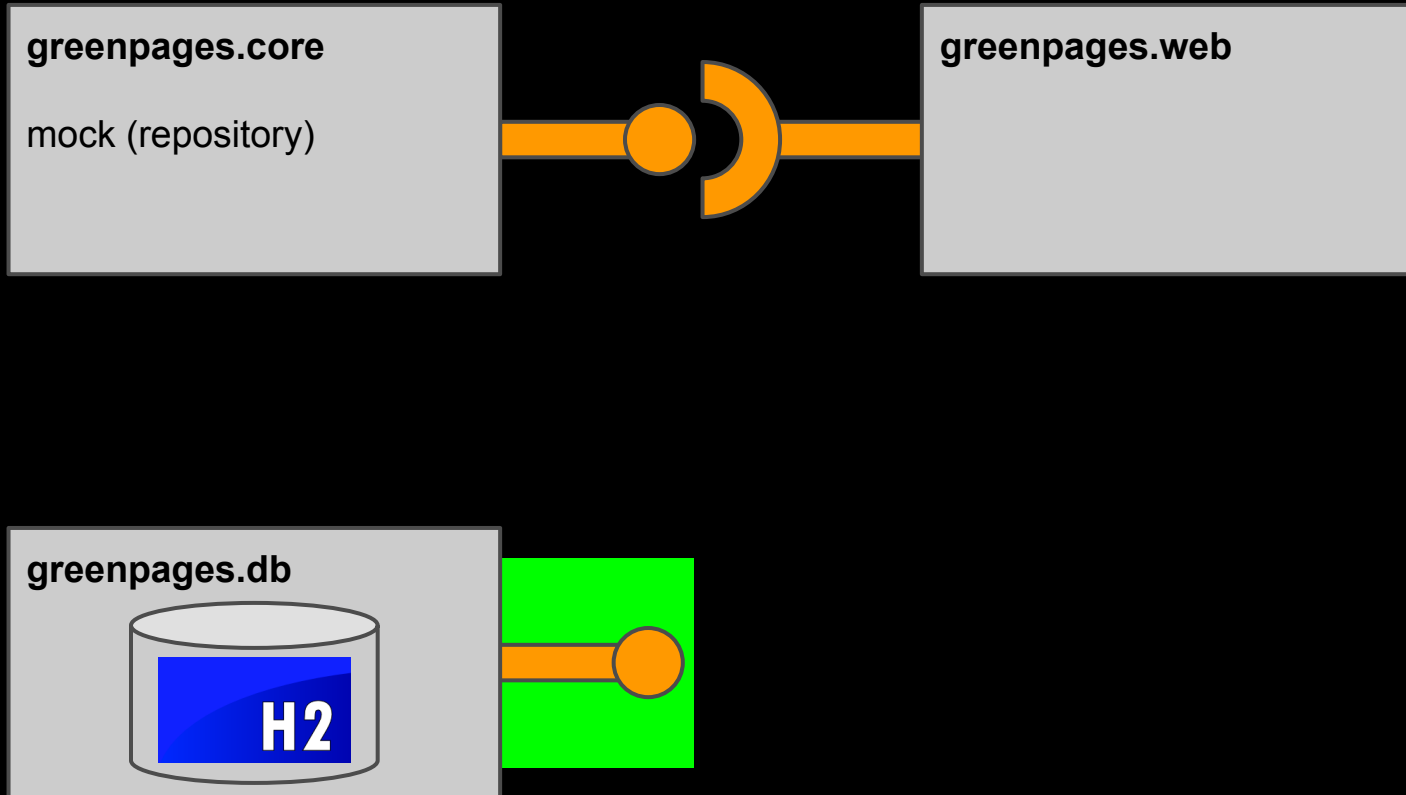
```
git diff safepoint_02_datasource
```

Prepare for next goal:

```
git checkout safepoint_02_datasource
```

```
import -> existing Maven Project greenpages.db
```

# Next Goal: Prepare database





## 2.1 Export javax.sql.DataSource

Export Spring bean “**dataSource**” as OSGi service

```
META-INF/spring/osgi-context.xml
```

```
<beans xmlns=...>  
    <osgi:service ... />  
</beans>
```

## 2.2 Add OSGi Console Commands

Provide “**execute**” and “**queryForInt**” as OSGi commands

```
<osgi:service ref="sqlCommandProvider" auto-export="all-classes">
  <osgi:service-properties>
    <entry key="osgi.command.scope">
      <value>sql</value>
    </entry>
    <entry key="osgi.command.function">
      <array value-type="java.lang.String">
        <value>execute</value>
        <value>queryForInt</value>
      </array>
    </entry>
  </osgi:service-properties>
</osgi:service>
```

## 2.3 Implement SQL commands

Implement SQL commands in  
`SqlCommandProvider`:

```
public void execute(String[] args)
```

```
public int queryForInt(String[] args)
```

## 2.4 Register Spring bean

```
<bean id="sqlCommandProvider"      class="greenpages.  
db.internal.SqlCommandProvider"  
      p:jdbcTemplate-ref="jdbcTemplate" />
```

## 2.5 Insert test data

```
$> telnet localhost 2501
```

```
Trying 127.0.0.1...
```

```
Connected to localhost.
```

```
Escape character is '^]'.
```

```
osgi> sql:execute CREATE TABLE ...
```

```
osgi> sql:execute INSERT INTO LISTING VALUES(...)
```

```
...
```

```
osgi> sql:queryForInt SELECT COUNT(*) FROM LISTING
```

# Safepoint 3 - Provide Datasource

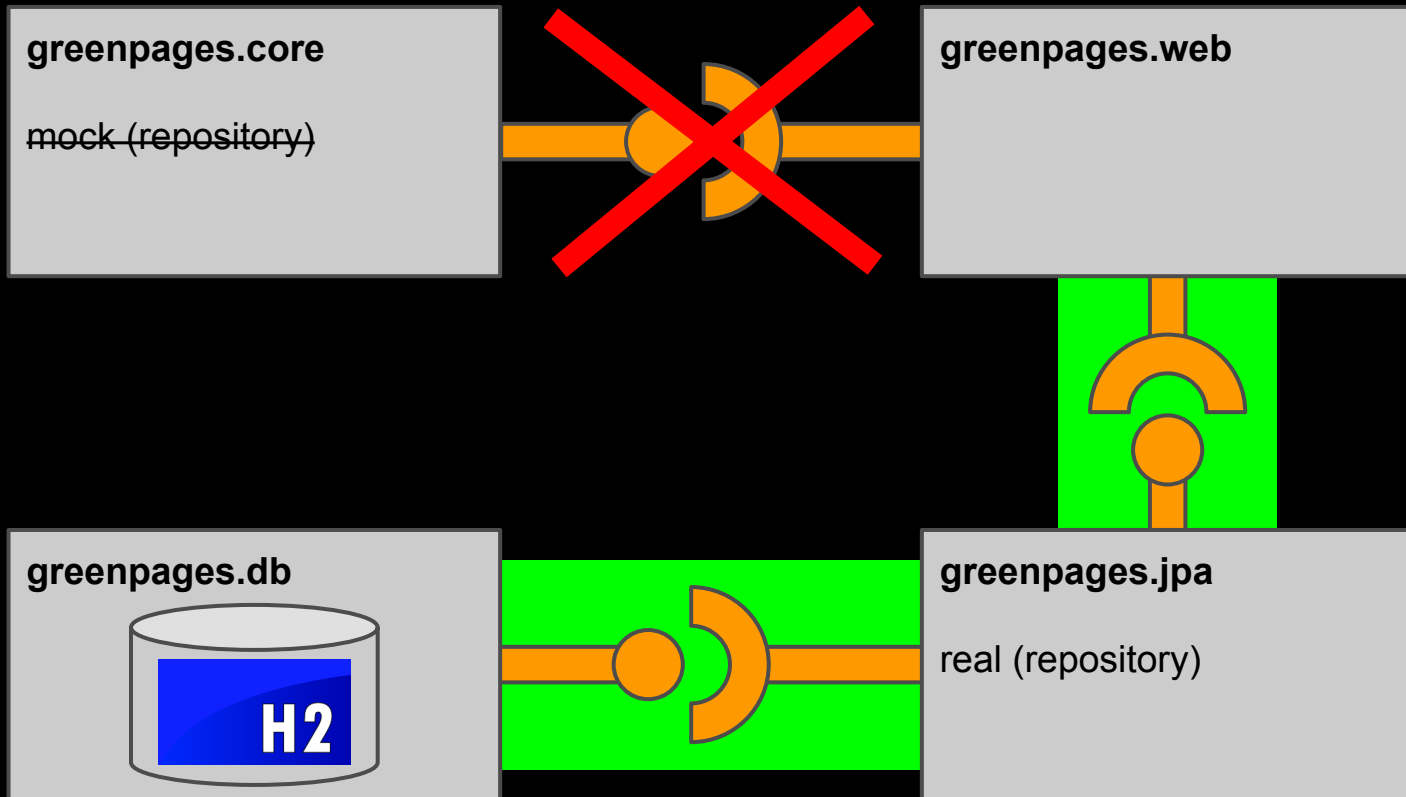
```
git diff safepoint_03_persistence
```

Prepare for next goal:

```
git checkout safepoint_03_persistence
```

import -> existing Maven Project `greenpages.jp`

# Next Goal: Persistence



## 3.1 Rewire OSGi services

```
<beans xmlns=...>  
    <osgi:service ... />  
    <osgi:reference ... />  
</beans>
```



# Safepoint 4 - Persistence

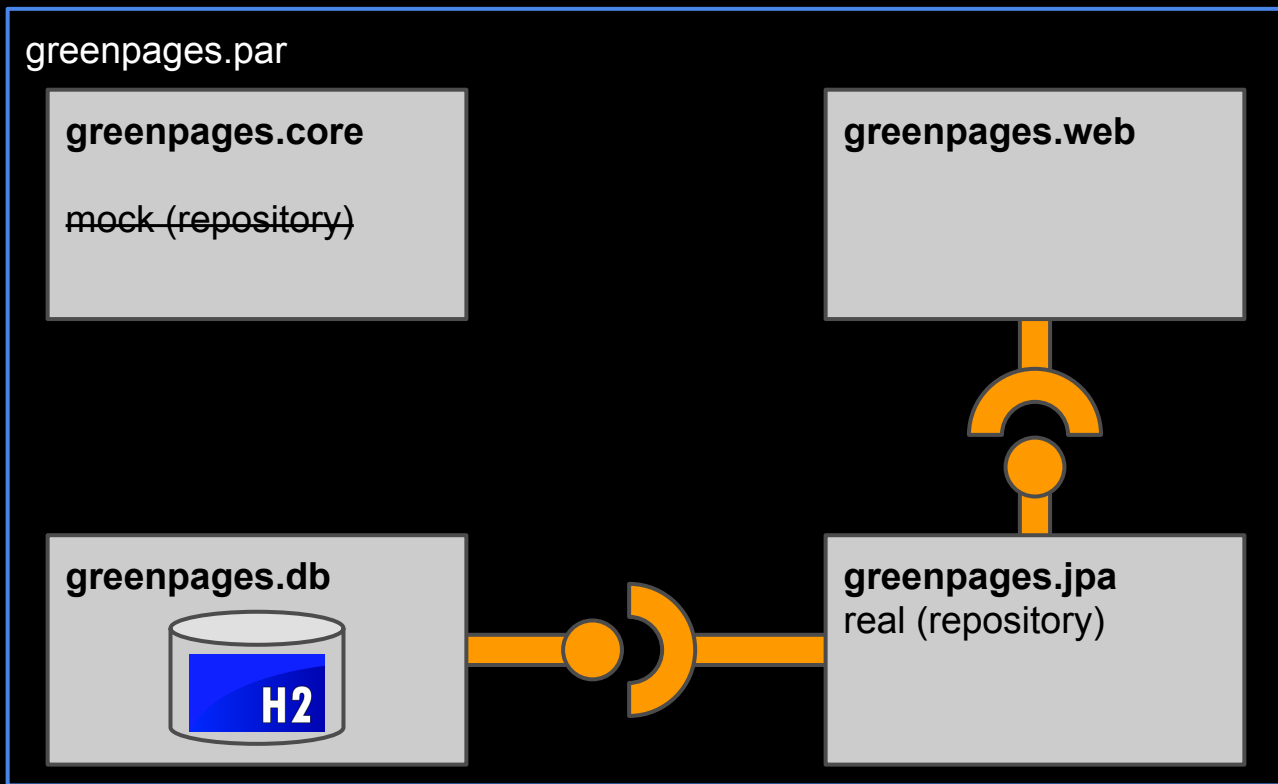
```
git diff safepoint_04_packaging
```

Prepare for next goal:

```
git checkout safepoint_04_packaging
```

```
import -> existing Maven Project greenpages.par
```

# Next Goal: Single Deployment Unit



# Package greenpages

```
mvn package
```

# Deploy greenpages

```
cp greenpages.par ${VIRGO_HOME}/pickup
```

# Inspection via Admin Console

OSGi Explorer

View relationships by: Wirings Services

From bundle [162] greenpages.par-3.0.0.RELEASE-greenpages.db  
3.0.0.RELEASE  
To bundle [163] greenpages.par-3.0.0.RELEASE-greenpages.jpa  
3.0.0.RELEASE

Service(s) between Bundles 162 and 163

Service
[278] javax.sql.DataSource
Published by Bundle 162
Used by Bundle 163
ObjectClass
javax.sql.DataSource

Virgo Admin Console from EclipseRT

© Copyright 2008, 2012 VMware Inc. Licensed under the Eclipse Public License v1.0.

The screenshot displays the Virgo Admin Console interface. On the left, the 'OSGi Explorer' panel lists various bundles, including 'org.eclipse.gemini.management - Fragments', 'org.eclipse.virgo.management.fragment', 'org.eclipse.virgo.management.console', 'org.jolokia.osgi', 'org.eclipse.virgo.apps.splash', 'org.eclipse.virgo.apps.repository-3.6.2.RELEASE', 'org.eclipse.virgo.apps.repository.web', 'org.eclipse.virgo.apps.repository-3.6.2.RELEASE', 'org.eclipse.virgo.apps.repository.core', 'org.eclipse.virgo.apps.repository-3.6.2.RELEASE-synthetic.context', 'com.springsource.freemarker', 'com.springsource.jdbc.datasource.impl', 'org.springframework', 'org.springframework', 'org.springframework', 'org.springframework', 'org.springframework', 'com.springsource', 'greenpages.par-3.0.0.RELEASE-com.springsource.org.h2', 'greenpages.par-3.0.0.RELEASE-greenpages.web', 'greenpages.par-3.0.0.RELEASE-greenpages.db', 'greenpages.par-3.0.0.RELEASE-greenpages.jpa', 'greenpages.par-3.0.0.RELEASE-greenpages.core', and 'greenpages.par-3.0.0.RELEASE-synthetic.context'. A pop-up window titled 'Service(s) between Bundles 162 and 163' shows a table with the following data: [278] javax.sql.DataSource, Published by Bundle 162, Used by Bundle 163, ObjectClass, and javax.sql.DataSource. On the right, a bundle relationship diagram shows bundles [0] org.eclipse.osgi, [76] org.eclipse.gemini.blueprint.extender, [162] greenpages.par-3.0.0.RELEASE-greenpages.db, [163] greenpages.par-3.0.0.RELEASE-greenpages.jpa, [66] org.eclipse.virgo.shell.command, and [161] greenpages.par-3.0.0.RELEASE-greenpages.web. The diagram indicates that bundle [163] is the central node, with arrows pointing to it from bundles [0], [76], and [162]. Bundle [163] then has arrows pointing to bundles [66] and [161]. A tooltip for bundle [163] shows the relationship between bundle [162] and bundle [163].

# Congratulations, you made it!



**Thank you!**

# Who has/can load a class?

```
osgi> clhas greenpages.Directory
```

```
Bundles containing [greenpages/Directory.class]:
```

```
136      greenpages.core  
        /greenpages/Directory.class
```

```
osgi> clload greenpages.Directory
```

```
Successfully loaded [greenpages.Directory] from:
```

```
136      greenpages.core  
137      greenpages.web  
        [provided by 136 greenpages.core]  
133      osgi.enterprise  
        [provided by 136 greenpages.core]
```



# miscellaneous

**Clean start** `${VIRGO_HOME}/bin/startup.sh -clean`

**Debug Virgo** `${VIRGO_HOME}/bin/startup.sh -debug -suspend`

**Virgo logs** `${VIRGO_HOME}/serviceability/logs/`

## ▼ Server Startup Configuration

Specify startup options. Changing a setting requires a server restart.

☐ Tail application trace files into Console view

☒ Start server with `-clean` option

`-XX:MaxPermSize=`

# Evaluate This Session

- 1 Sign-in: [www.eclipsecon.org](http://www.eclipsecon.org)
- 2 Select session from schedule
- 3 Evaluate:   