Design Patterns and Frameworks Dr.-Ing. Max Leuthäuser INF 2081 http://st.inf.tu-dresden.de/teaching/dpf Exercise Sheet No. 12 Software Technology Group Institute for SMT Department of Computer Science Technische Universität Dresden 01062 Dresden

OSGi and Design Patterns

Task 12.1: The OSGi Framework

Download Eclipse Classic 3.7.1 from http://www.eclipse.org and install the ObjectTeams Development Tooling (OTDT) and the ObjectTeams Equinox Integration (OT/Equinox) via the update manager of Eclipse.

Object Teams is part of the Indigo simultaneous release. This means, no further URL must be configured for installing the OTDT and OT/Equinox, simply select the Indigo - http://download.eclipse.org/releases/indigo software site and open the *Programming Languages* category.

To improve your understanding on OSGi in Eclipse read the tutorial of Lars Vogel¹.

1a)

Now create a new OT Plug-in Project dpf.osgi for the OSGi framework Equinox. Make sure you add org.eclipse.core.runtime to the plug-in dependencies.

Create the following plugin.xml in the plug-in's root folder.

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.7"?>
<plugin>
  <extension
    id="app"
    point="org.eclipse.core.runtime.applications">
    <application</pre>
      cardinality="singleton-global"
      thread="any"
      visible="true">
      <run
        class="dpf.osgi.Apparat">
      </run>
    </application>
  </extension>
</plugin>
```

Implement an application using the following code.

```
import org.eclipse.equinox.app.IApplication;
import org.eclipse.equinox.app.IApplicationContext;
public class Apparat implements IApplication {
    @Override
    public Object start(IApplicationContext context) throws Exception {
        Person hans = new Person("Hans");
        Person karl = new Person("Karl");
```

¹http://www.vogella.de/articles/OSGi/article.html

```
hans.identify();
karl.identify();
return null;
}
@Override
public void stop() { }
}
```

Now create a second OT Plug-in Project dpf.osgi.base without generating an Activator and which only consists of the Person class outlined below.

```
public class Person {
   public String name;
   public Person(String name) {
     this.name = name;
   }
   public String getName() {
     return name;
   }
   public void setName(String name) {
     this.name = name;
   }
   public void identify() {
     System.out.println(name);
   }
}
```

Now create a new OSGi run configuration, deactivate all bundles, enable the dpf.osgi.* bundles and click Add Required Bundles. Then click Validate Bundles to ensure that no bundles are missing or conflicting.

Enable the Enable OT/Equinox setting.

ame: 🛛	dpf.osgi les 🛛 (M= Arguments) 🔄 Settings) 🖶 Tracing 🛛 🌄 Environment 🕅 📰	Common		
Framew	vork: Equinox 🗘 Default Start level: 4 🗘 Defa	ult Auto-Start:	true	
type f	îlter text			Select All
Bun	dles	Start Level	Auto-Start	
	♦ Workspace			Deselect All
	dpf.osgi (1.0.0.qualifier)	default	default	
	↓ dpf.osgi.base (1.0.0.qualifier)	default	default	Add Working Set
	↓ dpf.osgi.ext (1.0.0.qualifier)	default	default	
	Target Platform			Add Required Bundles
	org.eclipse.core.contenttype (3.4.1.R35x_v20090826-0451)	default	default	
	org.eclipse.core.jobs (3.4.100.v20090429–1800)	default	default	Restore Defaults
	org.eclipse.core.runtime (3.5.0.v20090525)	default	true	(nestore benauts
	org.eclipse.equinox.app (1.2.0.v20090520-1800)	default	default	
	org.eclipse.equinox.common (3.5.1.R35x_v20090807-1100)	2	true	
	> org.eclipse.equinox.preferences (3.2.300.v20090520-1800)	default	default	
	org.eclipse.equinox.registry (3.4.100.v20090520-1800)	default	default	
	org.eclipse.osgi (3.5.1.R35x_v20090827)	-1	true	
	Image: Provide the second state of the seco	default	default	
	org.objectteams.otequinox.hook (1.3.2.200909280002)	default	false	
	org.objectteams.otequinox.runtime (1.3.2.200909280002)	default	false	
	🐌 org.objectweb.asm (3.1.0.v200803061910)	default	default	Only show selected bundles
				15 out of 20 selected
🗹 Incl	ude optional dependencies when computing required bundles			
🗹 Add	new workspace bundles to this launch configuration automatically			
🗌 Vali	date bundles automatically prior to launching			Validate Bundles

Run the application. The commands help, apps and startApp on the OSGi console should get you started.

Task 12.2: ObjectTeams and OT/Equinox

2a)

Try out and understand the ObjectTeams Observer pattern example which can be found under File \Rightarrow New. . . \Rightarrow Examples. . .

● ○ ○ New Example	
Select a wizard Implementation and application of the standard Observer pattern with ObjectTeams/Java.	2
Wizards:	
type filter text	0
ATM Example Ilight Bonus Example Observer Pattern Example Order System Example Stop Watch Example	
? < Back Next > Cancel Finish	

The ObjectTeams Language Reference² provides detailed explanation on roles in ObjectTeams.

```
2b)
```

In this subtask we are going to extend the example of Task 1. Create an ObjectTeams Plug-in Project dpf.osgi.ext (without generating an Activator).

Add org.eclipse.objectteams.otequinox to the dependencies via dpf.osgi.ext's MANIFEST.MF.

Import the exported package of dpf.osgi.base at dpf.osgi.ext and the packages of dpf.osgi.base and dpf.osgi.ext at dpf.osgi.

Now integrate the Team below in the new plug-in and change the implementation of IApplication in the dpf.osgi plug-in.

```
public team class University {
    public void register(Person as Student ersti, int matrikel) {
        ersti.matrikel = matrikel;
    }
    public class Student playedBy Person {
        studentIdentify <- before identify;
        public int matrikel;
        public void studentIdentify() {
            System.out.println("Matrikel: "+matrikel);
        }
    }
}</pre>
```

 $^{^{2}}$ http://wiki.eclipse.org/OTJ

```
import org.eclipse.equinox.app.IApplication;
import org.eclipse.equinox.app.IApplicationContext;
public class Apparat implements IApplication {
 @Override
 public Object start(IApplicationContext context) throws Exception {
   Person hans = new Person("Hans");
   Person karl = new Person("Karl");
   System.out.println("--No context--");
   hans.identify();
   karl.identify();
   University u = new University();
   u.activate();
   u.register(hans, 123);
   u.register(karl, 345);
   System.out.println("--Uni active-----");
   hans.identify();
   karl.identify();
   u.deactivate();
   System.out.println("--Uni inactive---");
   hans.identify();
   karl.identify();
   return null;
 }
 @Override
 public void stop() { }
}
```

To start this application some changes need to be integrated in the Eclipse run configuration.

First, register the extension point org.eclipse.objectteams.otequinox.aspectBindings in the plugin.xml.³ It should look like this:

2c)

Extend the previous solutions with the role **professor**. A professor can be identified through his group membership.

 $^{^{3}} http://wiki.eclipse.org/OTEquinox/Aspect_Binding$

```
---Uni inactive---

...

---Uni active-----

Matrikel: 123

Hans

Matrikel: 345

Anja

Peter is professor of the database group

----Uni inactive---

...
```

2d)

Extend the solution with lectures, which are held by professors and attended by students. Implement a method which returns all lectures attended by a student.