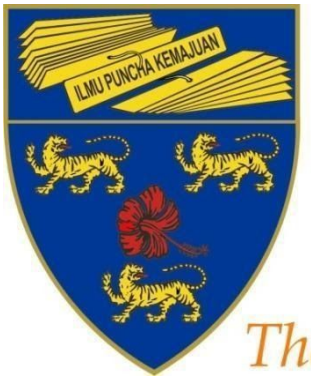


UNIVERSITY OF MALAYA

The Leader in Research & Innovation

WIF3006
COMPONENT BASED SOFTWARE ENGINEERING
SEMESTER I 2020/2021

Lecturer : Dr. Mumtaz Begum binti Peer Mustafa

Report on Lab Practises on OSGi

Group Members	Matric Number
Muhammad Fariz Rizal bin Shaharum	17127981/1
Muhammad Amin bin Mohd Din	17184491/1
Nur Arina binti Mohd Nor	17076071/1
Nur Fatihah Atikah binti Mohd Rashdan	17100287/1
Nur Sabrina binti Mohd Marzuki	17133400/1
Nur Awatif binti Wahadi	17169707/1

1.1 Source Code

I. AgeCalculatorApi (IAgeCalculator.java)

```java
package codabook.agecalculator.osgi.ifce;

import java.util.Calendar;

/*****************************************************************************************

public interface IAgeCalculator {
    public int calculateAge(Calendar dateOfBirth);
}
```

II. AgeCalculatorClient (AgeCalculatorOSGiClient.java)

```java
package codabook.agecalculator.osgi.client;

import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.Scanner;

import org.osgi.framework.ServiceReference;
import org.osgi.service.component.ComponentContext;

import codabook.agecalculator.osgi.ifce.IAgeCalculator;

/**********************************************************************************************/

public class AgeCalculatorOSGiClient {

    ComponentContext context;
    ServiceReference reference;
    IAgeCalculator ageCalculator;

    public void activate(ComponentContext context) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("What is your year of birth?");
        int year = scanner.nextInt();

        System.out.println("What is your month of birth (1-12)?");
        int month = scanner.nextInt();

        System.out.println("What is your date of birth (1-31)?");
        int date = scanner.nextInt();

        Calendar dateOfBirth = new GregorianCalendar();
        dateOfBirth.clear();
        dateOfBirth.set(year, month - 1, date);

        if (reference != null) {
            ageCalculator = (IAgeCalculator) context.locateService(
                    "IAgeCalculator", reference);

            int age = ageCalculator.calculateAge(dateOfBirth);

            System.out.println("Your age is " + age);

        }

    }

    public void gotService(ServiceReference reference) {
        System.out.println("Bind Service");
        this.reference = reference;
    }

    public void lostService(ServiceReference reference) {
        System.out.println("unbind Service");
        this.reference = null;
    }

}
```

III. AgeCalculatorImpl (AgeCalculatorImpl.java)

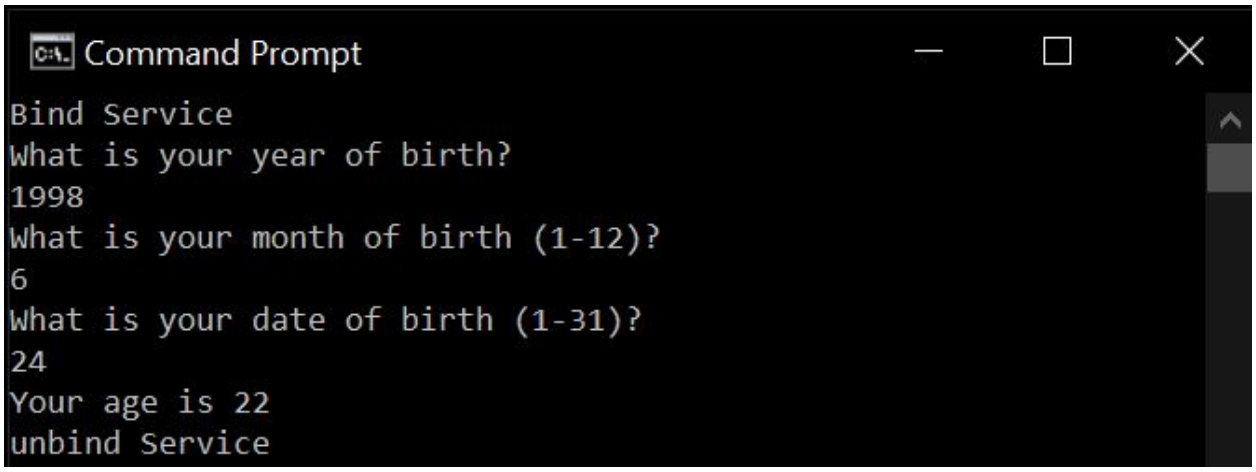
```java
package codabook.agecalculator.osgi.impl;

import java.util.Calendar;
import java.util.GregorianCalendar;

import codabook.agecalculator.osgi.ifce.IAgeCalculator;

/**********************************************************************************************/

public class AgeCalculatorImpl implements IAgeCalculator {

    public int calculateAge(Calendar dateOfBirth) {
        Calendar rightNow = new GregorianCalendar();

        int currentYear = rightNow.get(Calendar.YEAR);
        int currentMonth = rightNow.get(Calendar.MONTH);
        int currentDate = rightNow.get(Calendar.DATE);

        int birthYear = dateOfBirth.get(Calendar.YEAR);
        int birthMonth = dateOfBirth.get(Calendar.MONTH);
        int birthDate = dateOfBirth.get(Calendar.DATE);

        int age = 0;

        boolean isCurrentYearBdayPassed = (currentMonth > birthMonth)
                || ((currentMonth == birthMonth) && (currentDate >= birthDate));

        if (isCurrentYearBdayPassed) {
            age = currentYear - birthYear;
        } else {
            age = currentYear - 1 - birthYear;
        }

        return age;
    }

}
```

IV. Launcher (Launcher.java)

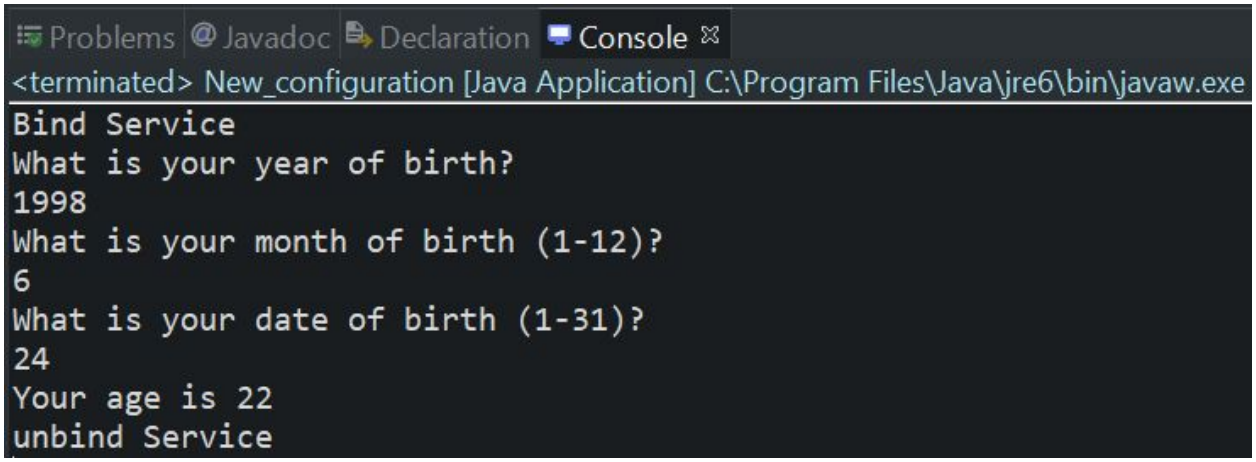
```java
package codabook.osgi;

import java.io.File;
import java.lang.reflect.Method;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import java.util.ServiceLoader;

import org.eclipse.osgi.framework.internal.core.Constants;
import org.osgi.framework.Bundle;
import org.osgi.framework.BundleContext;
import org.osgi.framework.BundleException;
import org.osgi.framework.launch.Framework;
import org.osgi.framework.launch.FrameworkFactory;
/*********************************************************************************************/
public class Launcher {

    public static void main(String[] args) {
        try {
            FrameworkFactory fwFactory = ServiceLoader
                    .load(FrameworkFactory.class).iterator().next();

            Map<String, String> configMap = new HashMap<String, String>();

            configMap.put("org.osgi.framework.storage.clean", "onFirstInit");

            Framework framework = fwFactory.newFramework(configMap);
            framework.init();
            BundleContext bndlCtxt = framework.getBundleContext();
            Bundle mainBundle = null;

            List<Bundle> bundleList = new LinkedList<Bundle>();

            File folder = new File(".");
            for (File file : folder.listFiles()) {
                if (file.getName().endsWith(".jar")&&!file.getName().contains("org.eclipse.osgi_")) {
                    Bundle bundle = bndlCtxt.installBundle(file.toURI()
                            .toString());
                    bundleList.add(bundle);

                    if (bundle.getHeaders().get("Main-Class") != null) {
                        mainBundle = bundle;
                    }
                }
            }

            framework.start();

            for (Bundle bundle : bundleList) {
                if (bundle.getHeaders().get(Constants.FRAGMENT_HOST) == null)
                    try {
                        bundle.start();
                    } catch (BundleException be) {
                        be.printStackTrace();
                    }
            }

            framework.stop();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}
```

1.2 Output

I. JAR File



II. Eclipse IDE



1.3 Discussion

- **OSGi Component Model**

1. Open Service Gateway Initiative or known as OSGi framework is a standard-based platform whose specifications are provided by OSGi Alliance.
2. Two parts of OSGi specification:
    a. OSGi framework
    b. OSGi standard service
3. The framework of OGSi is the OSGi runtime environment which provides all the functionality as per specification.
4. In the OSGi framework, applications are being deployed and executed.
5. Besides that, API for the development of components are being provided by the OSGi framework.
6. OSGi standard services is a reusable service which provides a part of development platform implementation.
7. OSGi framework has three conceptual layers

Layer	Responsibility
Module	Packaging and sharing the code
Life cycle	Managing the life cycle of a deployed module during runtime
Service	Dynamic service publication, searching and binding

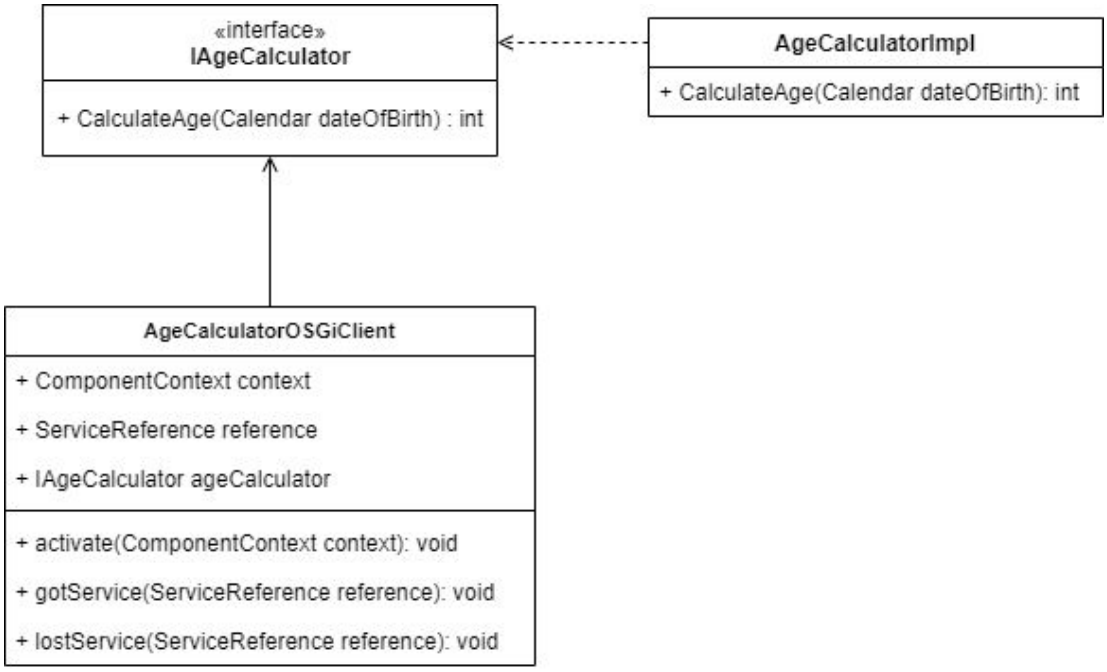
- **OSGi Bundle**

1. A deployment module in the form of a JAR file that contains manifest information where metadata about the bundle is stored unlike a regular JAR file where it only has class files and resource files.
2. Manifest file has specific OSGi information such as module name, version number and number of dependencies.
3. Bundles explicitly defines what portion of its code is externally visible, thus, a bundle must always declare its external dependencies that it has

- **OSGi Service Registry**

1. OSGi service registry uses service-oriented programming where it provides service publication, service discovery and service binding.
2. A service is defined by a JAVA interface.
3. A service can only be used once it found its service provider and bind with it.
4. Services can appear and go dynamically in a runtime application, just like bundles.

OSGi AgeCalculator Component Model



I. IAgeCalculator

- IAgeCalculator interface provides service binding for AgeCalculator application and is defined by the interface bundle *codabook.agecalculator.osgi.ifce*
- This section of the code will be used by AgeCalculatorImpl to implement the exposed services, in this case, the method calculateAge(Calendar dateOfBirth).
- It defines its dependencies and exports *codabook.agecalculator.osgi.ifce* according to MANIFEST.MF as follows :-

```
1 Manifest-Version: 1.0
2 Bundle-ManifestVersion: 2
3 Bundle-Name: Ifce
4 Bundle-SymbolicName: AgeCalculatorApi
5 Bundle-Version: 1.0.0.qualifier
6 Bundle-ActivationPolicy: lazy
7 Bundle-RequiredExecutionEnvironment: JavaSE-1.6
8 Import-Package: org.osgi.framework;version="1.3.0"
9 Export-Package: codabook.agecalculator.osgi.ifce
10
```

MANIFEST.MF

- IAgeCalculator is used by AgeCalculatorClient to invoke the calculateAge() method in AgeCalculatorImpl where it takes the calendar parameter.

II. AgeCalculatorImpl

- AgeCalculatorImpl implements IAgeCalculator interface and its provided service by importing the interface bundle from IAgeCalculator.
- component.xml defined AgeCalculatorImpl component and the service it required and where it is provided, in this case, IAgeCalculator.

AgeCalculatorImpl/META-I

Node	Content
?-? xml	version="1.0" encoding="UTF-8"
scr:component	
xmlns:scr	http://www.osgi.org/xmlns/scr/v1.1.0
name	codabook.agecalculator.osgi.impl
implementation	
class	codabook.agecalculator.osgi.impl.AgeCalculatorImpl
service	
provide	
interface	codabook.agecalculator.osgi.ifce.IAgeCalculator

component.xml

- The dependencies of AgeCalculatorImpl are defined as below :-

III. AgeCalculatorOSGiClient

- AgeCalculatorOSGiClient component will consume the services which provided by the IAgeCalculator component.
- Three methods are defined:

activate	Fragment that declares the API services
	Invoke when the component is activated
	To ***locate the IAgeCalculator***, componentContext is used with injected service reference
gotService	An user-defined method which being mentioned in component.xml
	When the service object is ***binded***, this method is invoked with the service reference (using dependency injection)
lostService	An user-defined method which being mentioned in component.xml
	When the service object is ***unbinded***, this method is invoked with the service reference

```
30      public void activate(ComponentContext context) {
58
59      public void gotService(ServiceReference reference) {
60          System.out.println("Bind Service");
61          this.reference = reference;
62      }
63
64      public void lostService(ServiceReference reference) {
65          System.out.println("unbind Service");
66          this.reference = null;
67      }
68
69  }
```

Methods in AgeCalculatorOSGiClient component

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0"
3   name="codabook.agecalculator.osgi.client">
4       <implementation class="codabook.agecalculator.osgi.client.
5       AgeCalculatorOSGiClient"/>
6       <reference bind="gotService" cardinality="1..1"
7       interface="codabook.agecalculator.osgi.ifce.IAgeCalculator"
8       name="IAgeCalculator" policy="dynamic" unbind="lostService"/>
9   </scr:component>
```

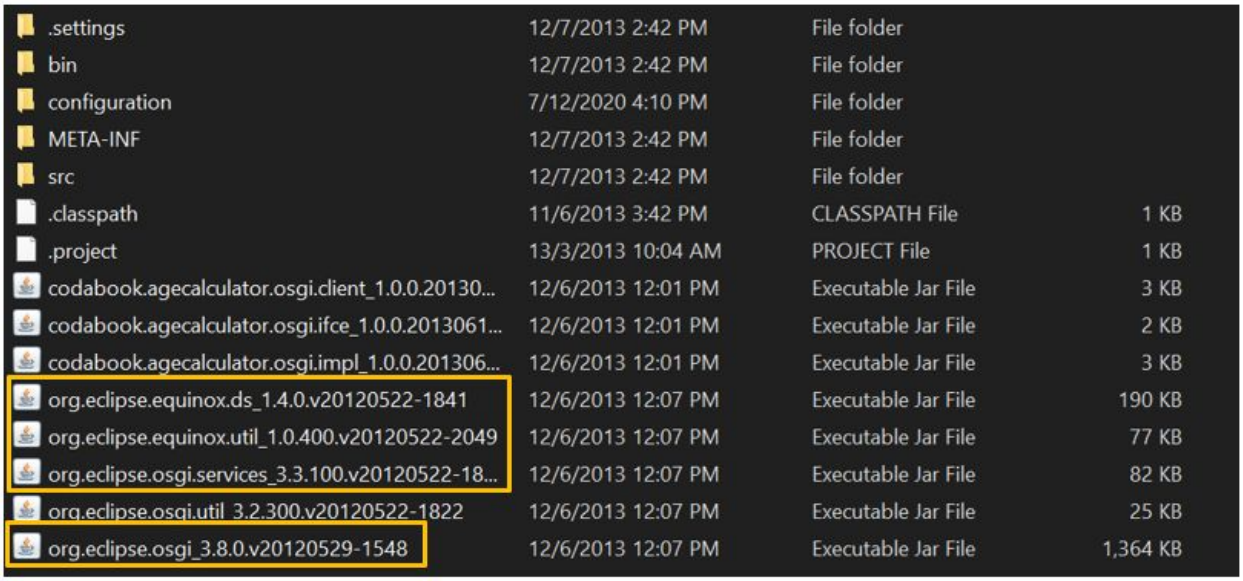
component.xml

1.4 Installation Guide

I. JAR Files

1. Save these listed files in the ‘AgeCalculator’ folder.
    - 'org.eclipse.equinox.ds_1.4.0.v20120522-1841.jar'
    - 'org.eclipse.equinox.util_1.0.400.v20120522-2049.jar'
    - 'org.eclipse.osgi_3.8.0.v20120830-144521.jar'
    - 'org.eclipse.osgi.services_3.3.100.v20120522-1822.jar'

2. The listed files can be found in the ‘section_3_5_src\Launcher’ folder.



3. To run the project, the steps can be followed from the README file.

II. SRC Files

1. **Requirement :** Eclipse IDE, Java SE 1.6
2. Open Eclipse IDE
3. Import > General > Existing Projects into Workspace
4. Select ‘section_3_5_src’
   Import all of these folders :-
    - AgeCalculatorAPI
    - AgeCalculatorImpl
    - AgeCalculatorClient
    - Launcher
5. Go to Window > Preferences > Build Path > User Libraries > New
6. Name the library ‘Equinox’
7. Go to ‘Add external JAR’ > Go to plugins folder
   Usually, it would be under eclipse folder (for eg: C:\Users\<User>\eclipse\plugins)
8. Find a file named ‘org.eclipse.osgi_3.8.0.v20120830-144521.jar’, click ‘Open’

9. Click on ‘Apply and Close’
10. Right click at on ‘AgeCalculatorAPI’ > Build Path > Configure build path > Add library > Choose ‘Equinox’
11. Go to ‘Add External JAR’ > Go to section_3_5_src folder > Launcher > Select all the JAR Files. Click on ‘Apply and Close’.
12. Repeat Step 10 and 11 with the rest of the folders.
13. Right click on ‘Launcher’ > Run > Run as Java Application > Select org.osgi.Launcher