Getting started with the HelloWorld application based on the e-Government Framework

Summary

This guide provides a HelloWorld tutorial to quickly work through features of the eGovFrame. It assumes the target reader has basic understanding in java and spring framework.

Follow three steps by step tutorial as below.

1. Install IDE: Configure the IDE for tutorial.
2. Create a Project: Using the sample project, create and run the HelloWorld application.
3. Get into details: Understand the fundamentals of applications based on the eGovFrame by getting into the source code of the project,

The table below lists requirements to build and run the HelloWorld application based on eGovFrame.

Item	Description	Notes
OS	Windows 2000, xp, vista	
JDK	Java SE SDK 5.0 or higher	
IDE	Eclipse 3.6.2	Included in an implementation tool

Step 1. Install the IDE

Install the eGovFrame IDE with an implementation tool and dependent libraries required to follow the HelloWorld tutorial.

Install the IDE

Refer to the installation guide for the implementation tool of the e-Government Framework based on the Eclipse to install the IDE.

Update Plug-Ins

Refer to the update guide for the implementation tool plug-in to use the latest modules.

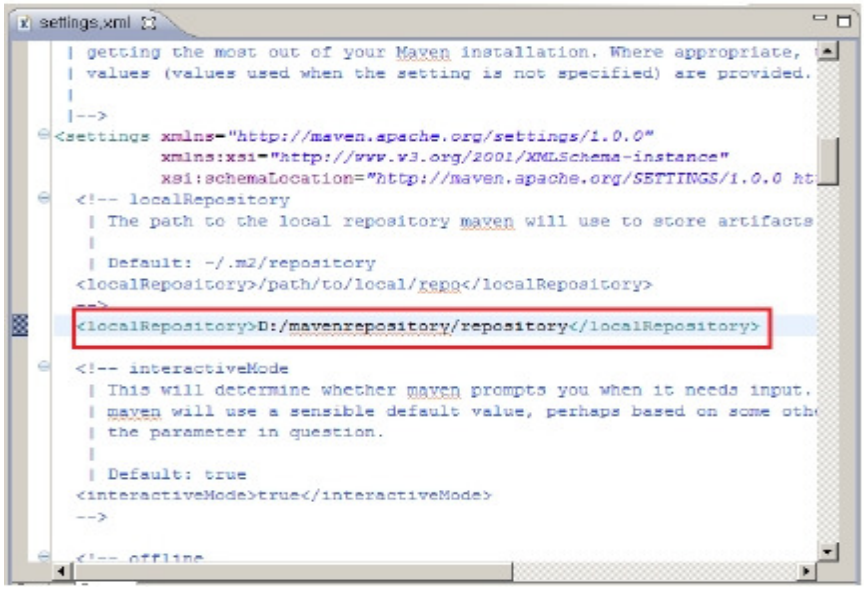
Maven Preferences

The IDE for getting started tutorial requires to manually copy dependent libraries to a Maven local repository as opposed to using Nexus in the runtime environment of real projects. The mavenrepository.zip file is provided to configure a maven local file repository. Unarchive the mavenrepository.zip file to an arbitrary directory and install it.

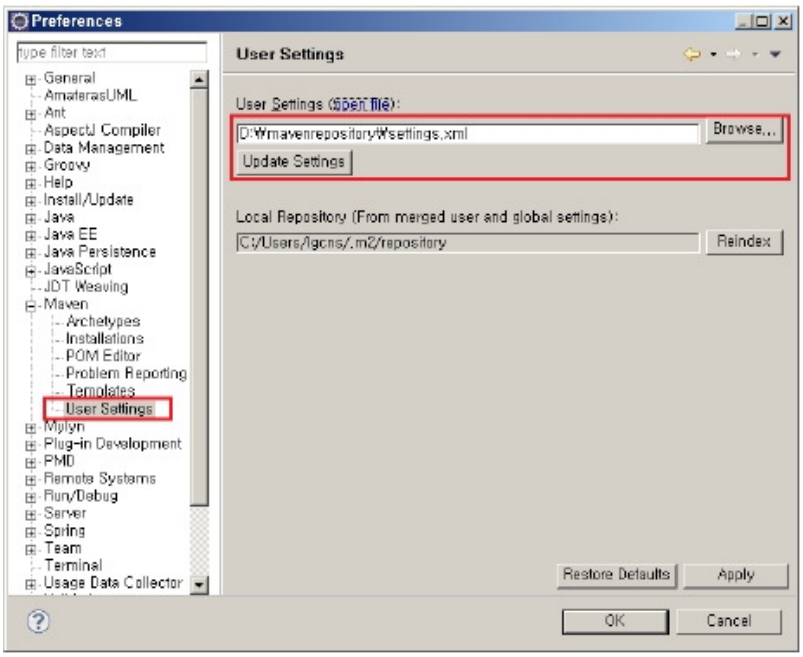
Dependent Library Installation

- Download the mavenrepository.zip file. This file contains a Maven configuration file and dependent libraries.
- Unarchive the downloaded file to an arbitrary directory. (Change the name of the directory to *[MavenRepository Installed Directory]*)
- Use the text editor to modify a localRepository item in *[MavenRepository Installed Directory]*/settings.xml file as follows.

```
<settings xmlns="http://maven.apache.org/settings/1.0.0"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">
    ...
  <localRepository> [MavenRepository Installed Directory]/repository </localRepository>
    ...
</settings>
```



- Launch the implementation tool.
- Display the Preferences window by selecting window>preferences menu of the implementation tool. Specify the User Settings to *[MavenRepository Installed Directory]*/settings.xml in Maven>Installations.
- Make sure the modified Local Repository item in settings.xml is the same as settings. If differs, click the refresh settings button.



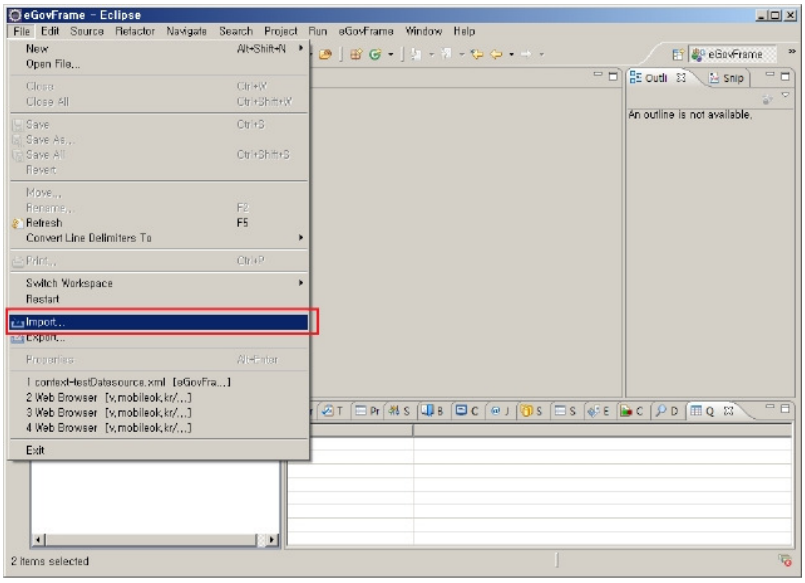
Step 2. Creating and running a project(Core)

Importing a Project

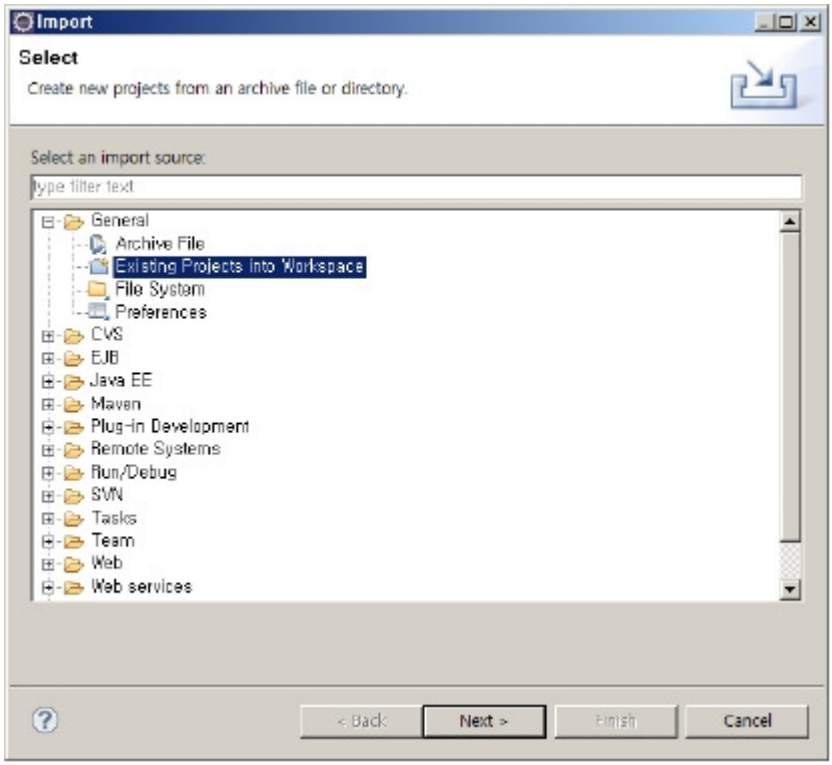
To demonstrate how to create and run a project, this guide uses the HelloWorld project. Follow steps below.

Creating a Project

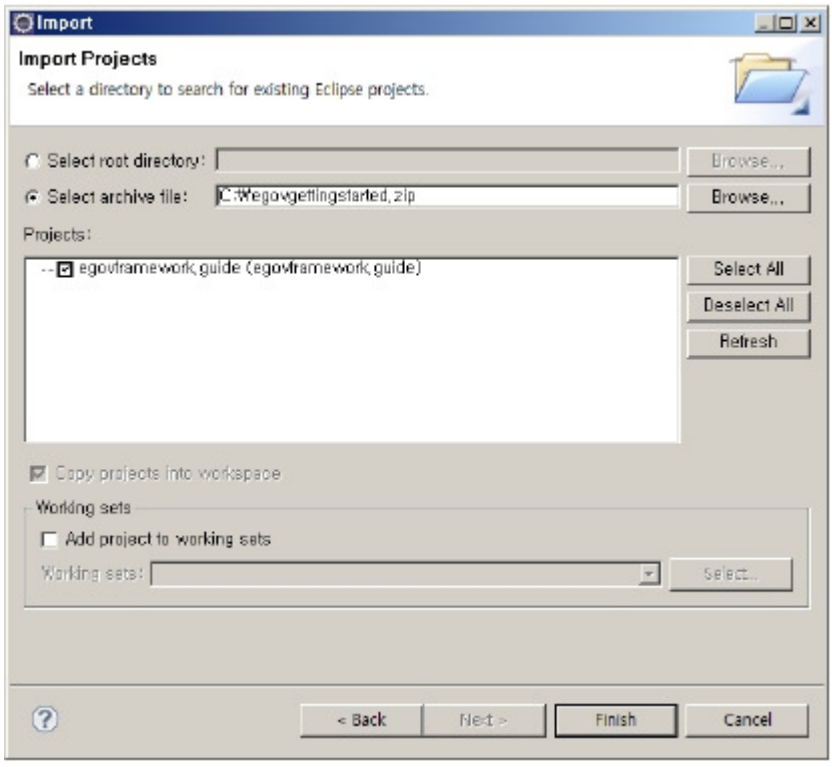
- Download the HelloWorld project file (Unarchiving is not required.).
- Select the File > Import in the implementation tool.



- Choose General>Existing Project into Workspace in an import wizard.
- Click the Next button.



- Choose the select archive file in the Import Projects, and then specify the egovGettingStarted.zip file you just downloaded.
- Click the Finish button.

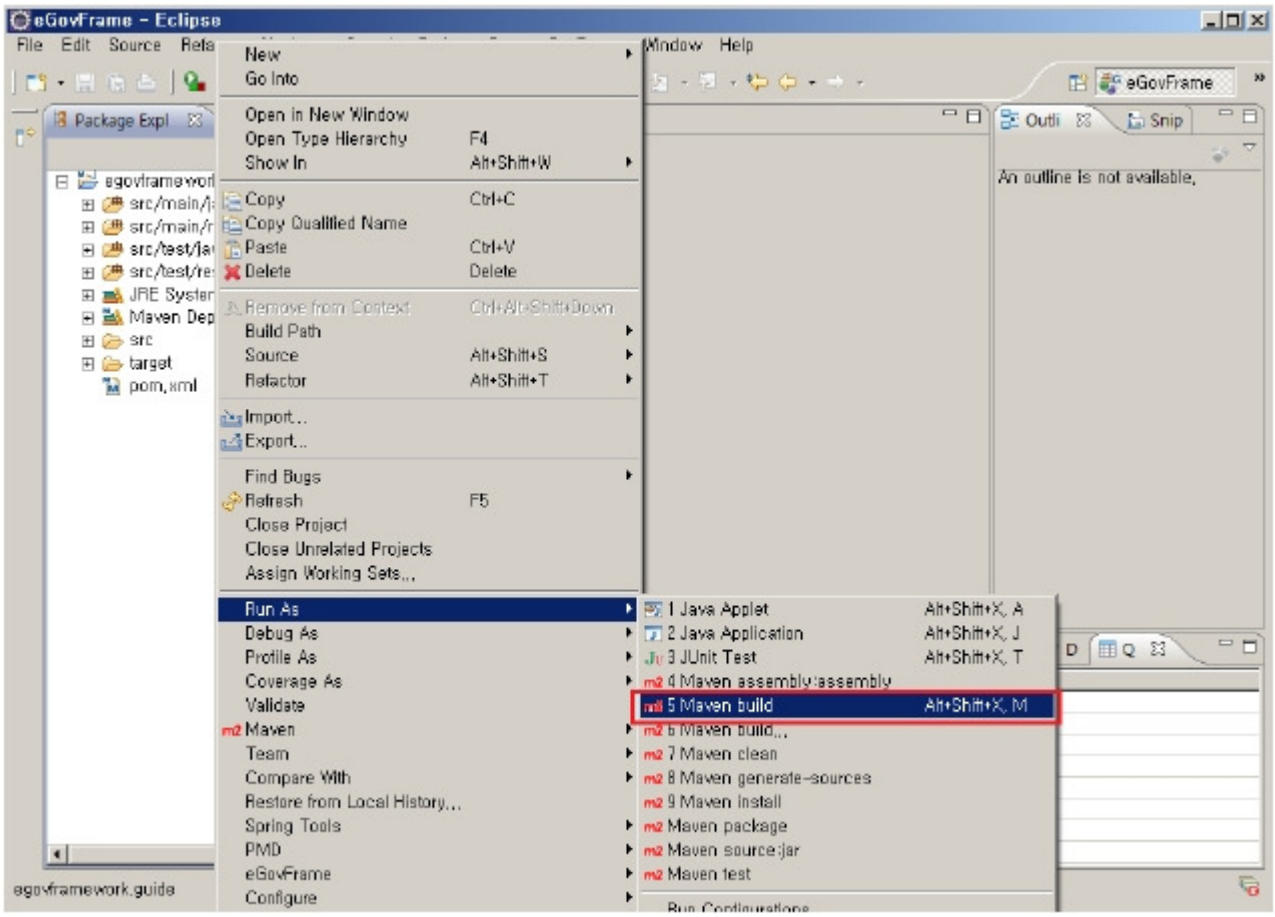


Build using Maven

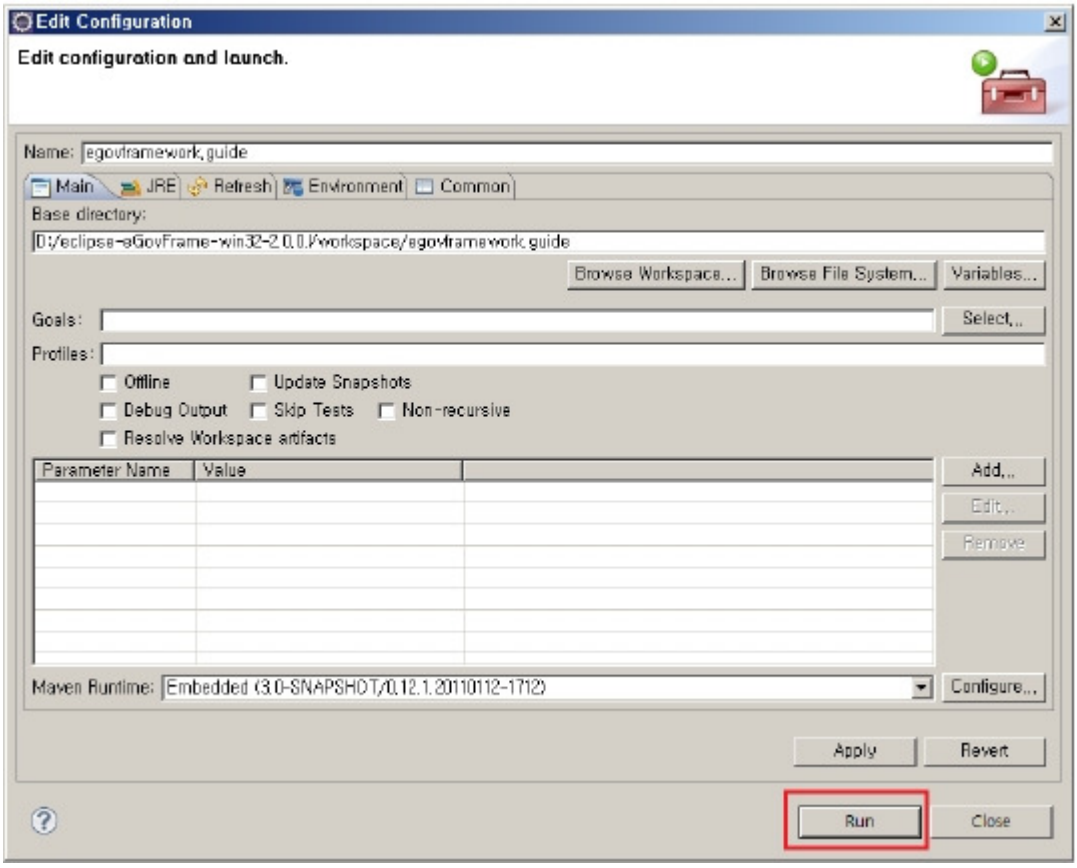
You can compile, test and package using the features of the implementation tool. As well as the Maven-based integrated build is also available. Let’s try the integrated build and see the result.

Project build steps

1. Right click the egovframework.guide in the IDE and select Run As>Maven Build.



- Click the Run button.



- Check the Maven build result on a console window. Maven build performs compile, testing and packaging of a project, and displays the result.

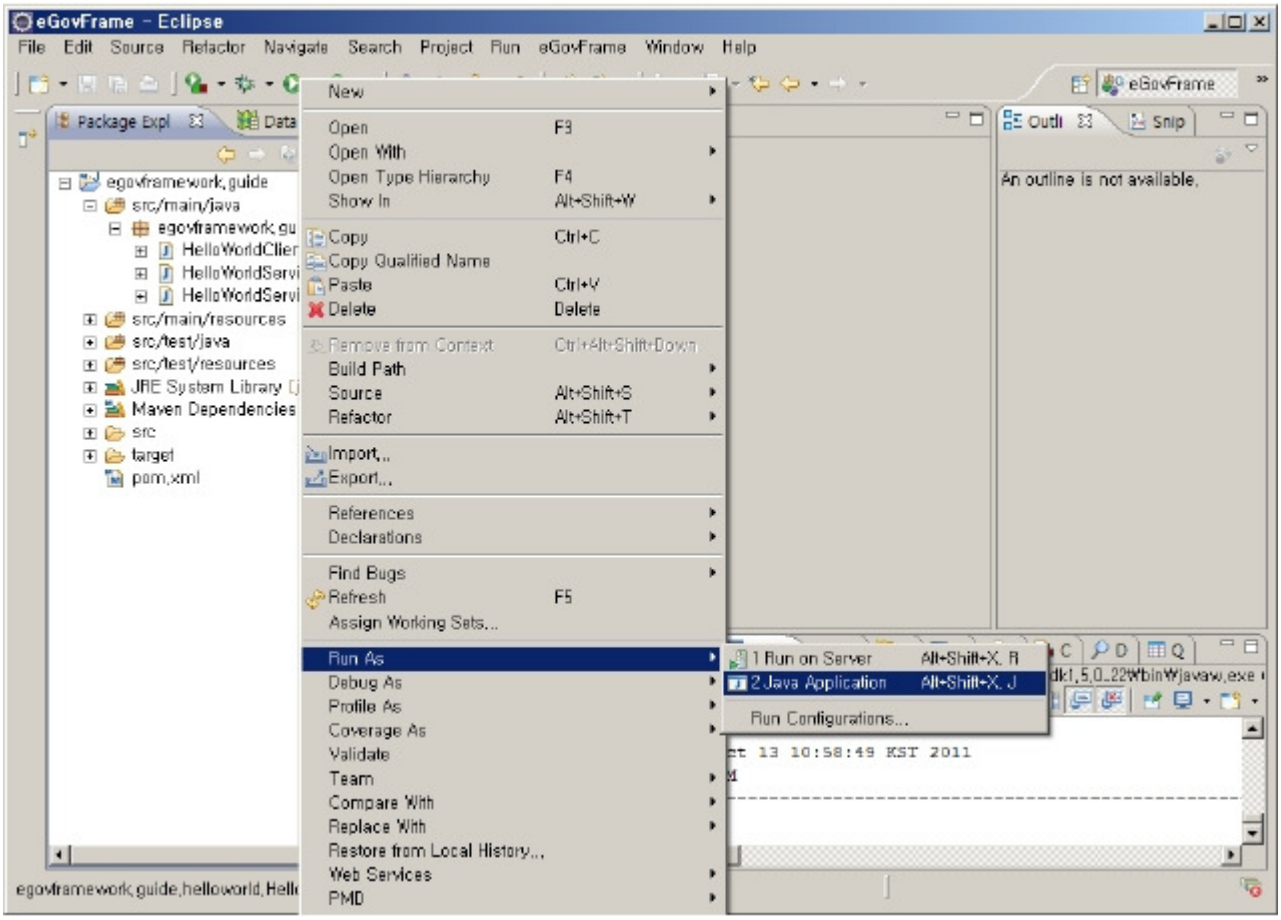
```
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for egovframework:guide:jar:1.0.0
[WARNING] 'dependencies.dependency.(groupId:artifactId:type:classifier)' must be unique: hsqldb:hsqldb:jar -> duplicate declaration of version 1.8.0.10 @ line 282, column 17
[WARNING] 'build.plugins.plugin.version' for org.apache.maven.plugins:maven-surefire-plugin is missing. @ line 396, column 12
[WARNING] 'reporting.plugins.plugin.version' for org.codehaus.mojo:emma-maven-plugin is missing. @ line 448, column 12
[WARNING] 'reporting.plugins.plugin.version' for org.codehaus.mojo:surefire-report-maven-plugin is missing. @ line 453, column 12
[WARNING] 'reporting.plugins.plugin.version' for org.apache.maven.plugins:maven-javadoc-plugin is missing. @ line 478, column 12
[WARNING] 'reporting.plugins.plugin.version' for org.apache.maven.plugins:maven-jxr-plugin is missing. @ line 490, column 12
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
```

```
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building guide 1.0.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.4.3:resources (default-resources) @ guide ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 2 resources
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:compile (default-compile) @ guide ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.4.3:testResources (default-testResources) @ guide ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:2.3.2:testCompile (default-testCompile) @ guide ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.7.1:test (default-test) @ guide ---
[INFO] Surefire report directory: D:\eclipse-eGovFrame-win32-2.0.0\workspace\egovframework.guide\target\surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running egovframework.guide.helloworld.HelloWorldServiceTest
log4j:INFO Using URL [file:/D:/eclipse-eGovFrame-win32-2.0.0r/workspace/egovframework.guide/target/classes/log4j.xml] for automatic log4j configuration of repository named [default].
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.367 sec

Results:

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.3.1:jar (default-jar) @ guide ---
Downloading: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-archiver/1.0/plexus-archiver-1.0.jar
Downloading: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-io/1.0/plexus-io-1.0.jar
Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-io/1.0/plexus-io-1.0.jar (50 KB at 35.4 KB/sec)
Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-archiver/1.0/plexus-archiver-1.0.jar (174 KB at 73.2 KB/sec)
[INFO] Building jar: D:\eclipse-eGovFrame-win32-2.0.0\workspace\egovframework.guide\target\egovframework.guide-coreapp.jar
[INFO]
[INFO] --- maven-install-plugin:2.3.1:install (default-install) @ guide ---
[INFO] Installing D:\eclipse-eGovFrame-win32-2.0.0\workspace\egovframework.guide\target\egovframework.guide-coreapp.jar to C:\repository\egovframework\guide\1.0.0\guide-1.0.0.jar
[INFO] Installing D:\eclipse-eGovFrame-win32-2.0.0\workspace\egovframework.guide\pom.xml to C:\repository\egovframework\guide\1.0.0\guide-1.0.0.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 9.147s
[INFO] Finished at: Thu Oct 13 10:58:49 KST 2011
[INFO] Final Memory: 4M/8M
[INFO] ------------------------------------------------------------------------
```

Run the HelloWorld

Steps to run the HelloWorld

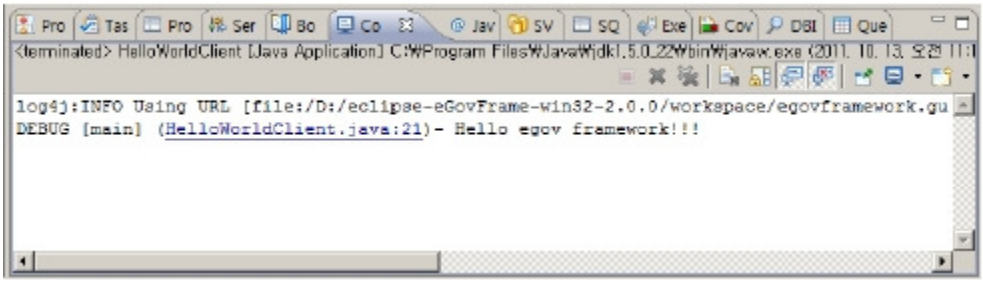
- Right click src/main/java/HelloWorldClient.java, then select Run As>Java Application.



A console window should display a running result as following figure:

.

- A console window should display a running result as following figure:

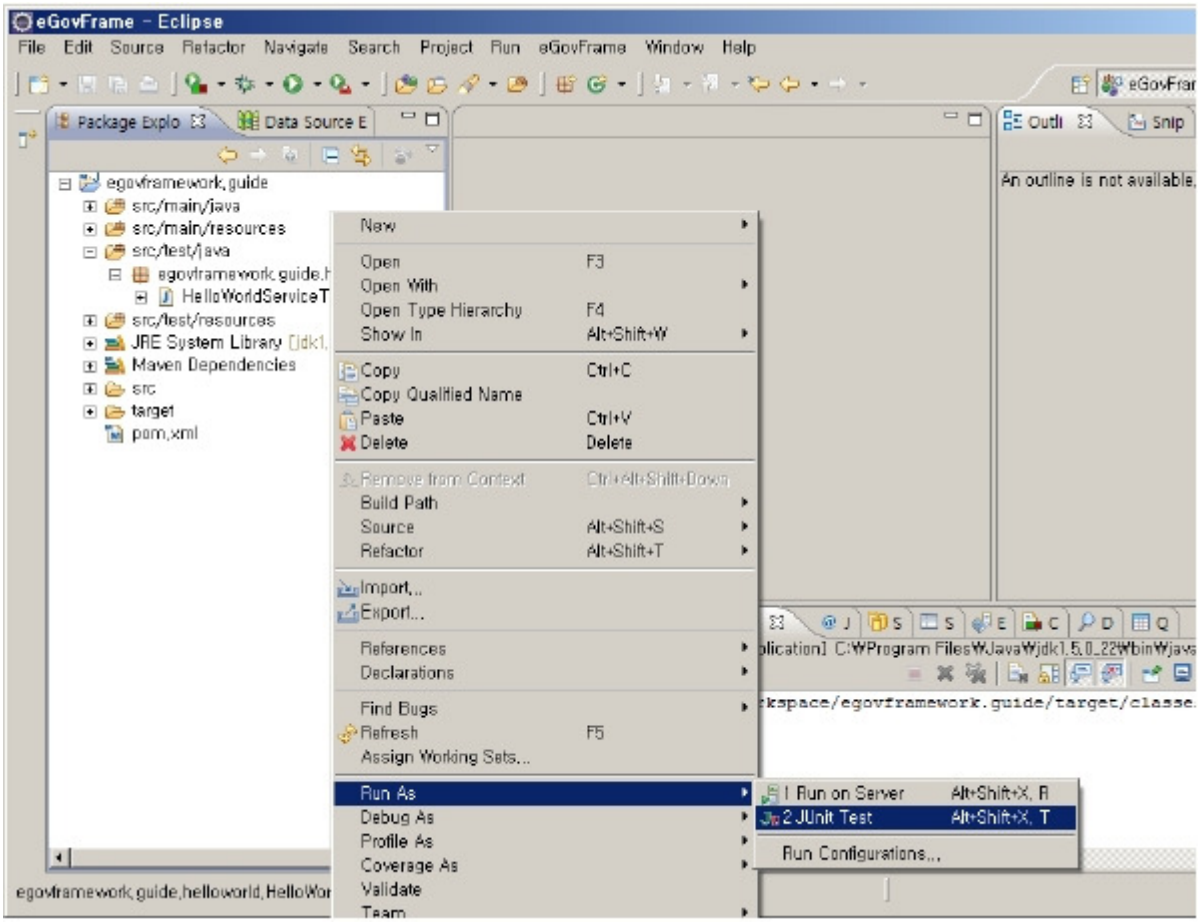


Unit test the HelloWorld

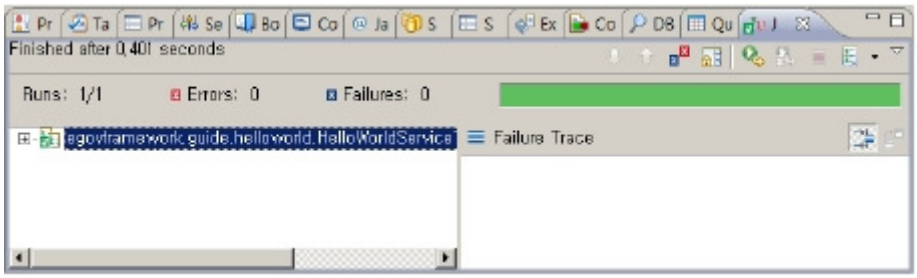
The HelloWorld project includes Test Case based on the JUnit Test Framework. The Test Case has been implemented to test the return value of the sayHello method in a HelloWorldServiceImpl class. Follow steps below to perform tests in the implementation tool.

Steps to test the HelloWorld

- Right click on junit test case for the HelloWorld service (HelloWorldServiceTest,java) in src/test/java, select Run As>Junit test.



- The JUnit result frame should display the test run time and test pass result (testSayHello).



Step 3. Look into the code

Service Interface Class

The following shows the HelloWorld service interface class: HelloWorldService. It declares “String sayHello()”.

```
package egovframework.guide.helloworld;

public interface HelloWorldService {
```

```
public interface HelloWorldService {
	public String sayHello();
}
```

Service Implementation Class

The following code shows the HelloWorld service implementation class: HelloWorldServiceImpl. It implements the HelloWorld Service.

```
package egovframework.guide.helloworld;

public class HelloWorldServiceImpl implements HelloWorldService{

	private String name;

	@Required
	public void setName(String name) {
		this.name = name;
	}

	public String sayHello() {
		return "Hello " + name + "!!!" ;
	}
}
```

Service Property Definition File

The following shows the HelloWorld service property definition file. It defines the name attribute as the helloworld, and name property for the service is declared as “egov framework”

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:context="http://www.springframework.org/schema/context"
	xsi:schemaLocation="
		http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
		http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-2.5.xsd">
		<context:annotation-config/>

		<bean name="helloworld" class="egovframework.guide.helloworld.HelloWorldServiceImpl">
			<property name="name">
				<value>egov framework</value>
			</property>
		</bean>

</beans>
```

Client Class

The following client class runs the HelloWorld Service.

```
package egovframework.guide.helloworld;

import	org.apache.commons.logging.Log;
import	org.apache.commons.logging.LogFactory;
import	org.springframework.context.ApplicationContext;
import	org.springframework.context.support.ClassPathXmlApplicationContext;

public class HelloWorldClient {

	private static Log logger = LogFactory.getLog(HelloWorldClient.class);

	/**
	 * @param args
	 */
	public static void main(String[] args) {
		String configLocation = "context-helloworld.xml";
		ApplicationContext context = new ClassPathXmlApplicationContext(configLocation);
		HelloWorldService helloworld = (HelloWorldSer-vice)context.getBean("helloworld");

		logger.debug(helloworld.sayHello());
	}

}
```

Test Class

The following code shows the unit test class for testing the HelloWorld service class.

```
package egovframework.guide.helloworld;

import static org.junit.Assert.*;

import	org.junit.Before;
import	org.junit.Test;
import	org.springframework.context.ApplicationContext;
import	org.springframework.context.support.ClassPathXmlApplicationContext;

public class HelloWorldServiceTest {

	private ApplicationContext context;

	@Before
	public void setUp() throws Exception {
		String configLocation = "context-helloworld.xml";
		context = new ClassPathXmlApplicationContext(configLocation);
	}

	@Test
	public void testSayHello() {
		HelloWorldService helloworld = (HelloWorldSer-vice)context.getBean("helloworld");
		assertEquals( "Hello egov framework!!!", hel-loworld.sayHello() );
	}

}
```