

Table of Contents

- 전자정부 표준프레임워크 기반의 시작하기(Getting Started)
 - 개요
 - Step 1. 개발환경 설치
 - 개발환경설치
 - 플러그인 업데이트
 - Maven 환경설정
 - Step 2. 프로젝트 생성(Core) 및 실행
 - 프로젝트 Import
 - Maven을 이용한 빌드
 - HelloWorld 실행
 - HelloWorld 테스트 실행
 - Step 3. 자세히 들여다 보기
 - 서비스 인터페이스 클래스
 - 서비스 구현 클래스
 - 서비스 속성 정의 파일
 - 클라이언트 클래스
 - 테스트 클래스

개요

본 가이드는 전자정부 표준프레임워크 기반의 단순한 응용프로그램(HelloWorld)를 직접 실행해 봄으로써 빠른 시간 내에 전자정부 프레임워크의 기본 기능을 파악하기 위하여 제공한다. 본 가이드의 사용자는 java 및 spring framework에 대한 기본적인 지식이 있다는 것을 가정하였다.

아래의 3가지 단계에 따라 순서대로 따라하기 방식으로 진행된다.

1. 개발환경 설치 : 실행을 위한 개발환경을 구축한다.
2. 프로젝트 생성 : 제공한 샘플 프로젝트를 이용하여 HelloWorld 응용 어플리케이션을 생성하고 실행해 본다.
3. 자세히 들여다 보기 : 생성/실행한 프로젝트의 내부 소스코드를 학습하여 전자정부 표준프레임워크 기반의 응용 어플리케이션 구현의 원리를 이해한다.

전자정부 표준프레임워크 기반의 Hello World 응용 프로그램 개발 및 실행을 위한 구현도구의 환경정보는 다음과 같다.

항목	설명	비고
OS	Windows 2000, xp, vista	
JDK	Java SE SDK 5.0 이상	
IDE	Eclipse 3.6.2	구현도구에 포함

Step 1. 개발환경 설치

전자정부 표준프레임워크에서 제공하는 구현도구(implementation tool) 및 HelloWorld 응용 어플리케이션을 위한 종속라이브러리를 이용하여 실행에 필요한 개발환경을 설치한다.


개발환경설치

먼저 eclipse 기반의 전자정부 표준프레임워크의 [구현도구\(implementation tool\)](#) 설치를 참조하여 설치한다.


플러그인 업데이트

설치한 구현도구의 플러그인이 최신 모듈을 사용할 수 있도록 [구현도구\(implementation tool\)](#) [플러그인 업데이트](#)를 참조하여 업데이트를 수행한다.

Maven 환경설정

시작하기 개발환경은 실제 프로젝트 수행환경과 달리 Nexus를 이용하지 않고, 종속 라이브러리를 Maven 로컬 파일저장소에 수동으로 복사하여 개발환경을 구축한다. Maven의 로컬 파일 저장소를 설정하기 위하여 제공한  mavenrepository.zip 파일을 임의의 디렉토리에 압축 해제하여 설치한다.

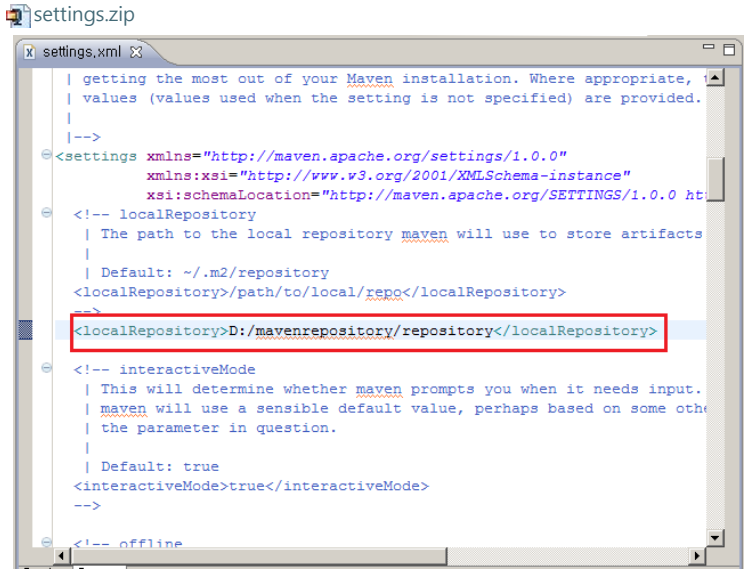
종속라이브러리 설치순서

- Maven 설정파일 및 종속라이브러리를 포함한  mavenrepository.zip 를 다운로드 한다.
- 다운로드 받은 파일은 임의의 디렉토리에서 압축해제한다.(압축해제한 디렉토리는 [\[MavenRepository 설치디렉토리\]](#) 로 명명한다.)
- 텍스트 에디터를 이용하여 [\[MavenRepository 설치디렉토리\]/settings.xml](#) 파일의 localRepository 항목의 값을 다음과 같이 수정한다.

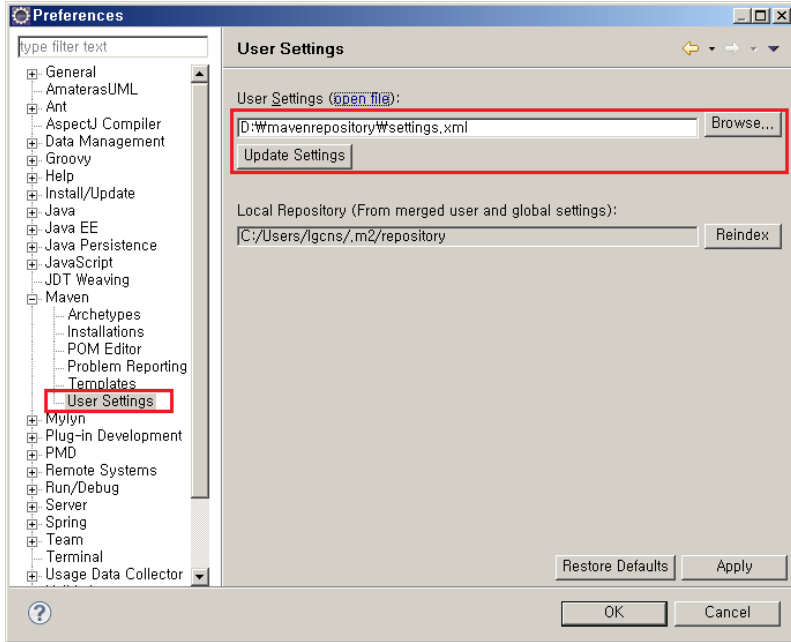
```
<settings xmlns="http://maven.apache.org/settings/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd">

  <localRepository> [MavenRepository 설치디렉토리]/repository </localRepository>

</settings>
```



- 구현도구를 실행한다.
- 구현도구의 window>preferences 메뉴를 선택하여 설정화면을 표시한다. 설정화면에서 Maven>Installtions 의 User Settings 항목을 [MavenRepository 설치 디렉토리]/settings.xml 파일로 지정한다.
- Local Repository 항목에 settings.xml 파일을 수정한 내용과 일치하는 것을 확인한다. 만약 설정한 내용과 다른 경우, refresh settings 버튼을 클릭한다.



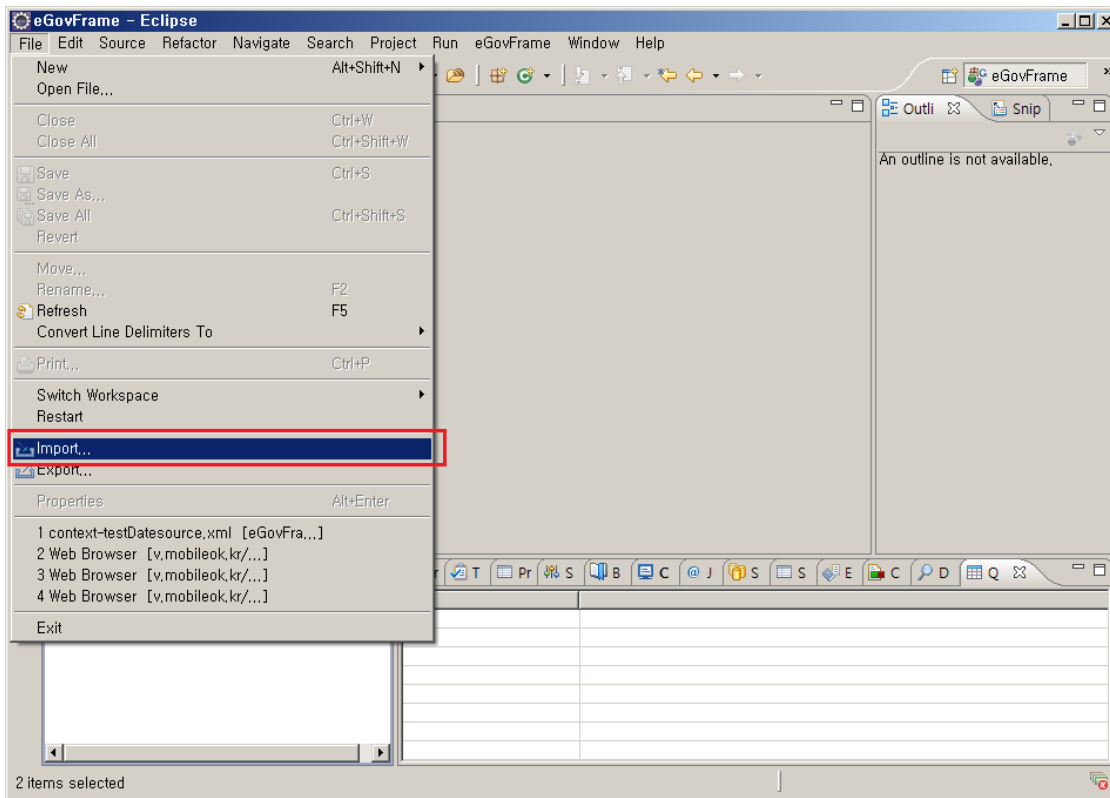
Step 2. 프로젝트 생성(Core) 및 실행

프로젝트 Import

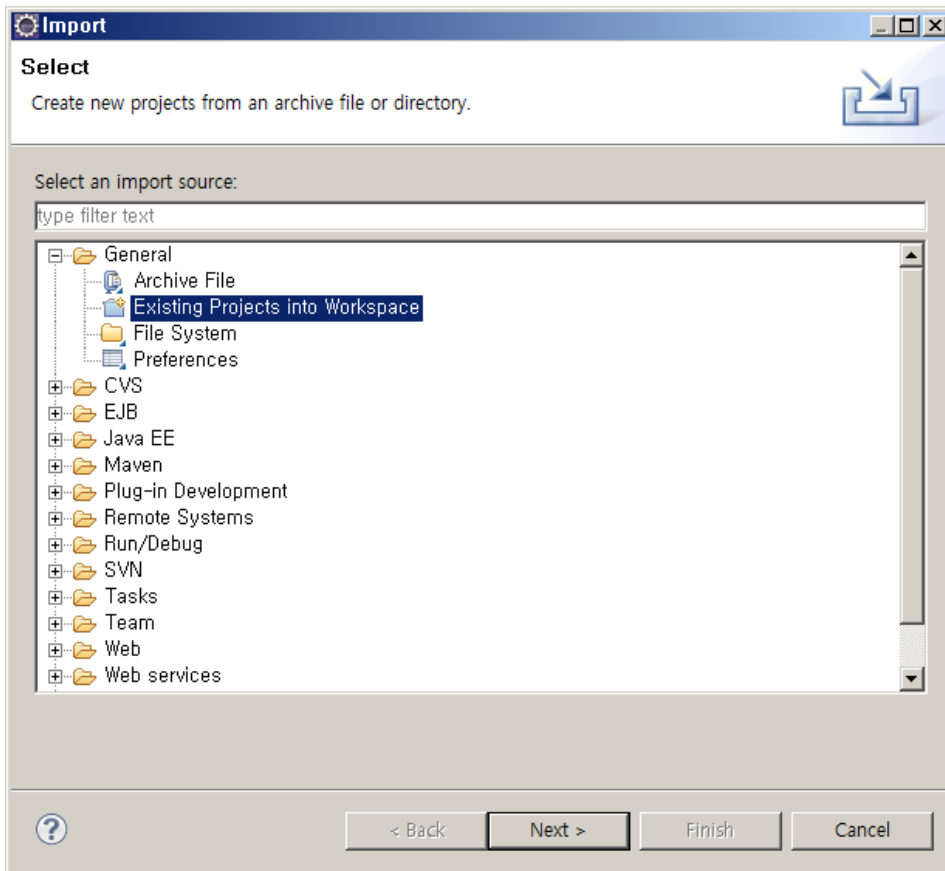
프로젝트 생성 및 실행을 위하여 본 가이드는 HelloWorld 프로젝트 파일을 제공한다. 아래의 순서에 따라 프로젝트를 생성한다.

프로젝트 생성순서

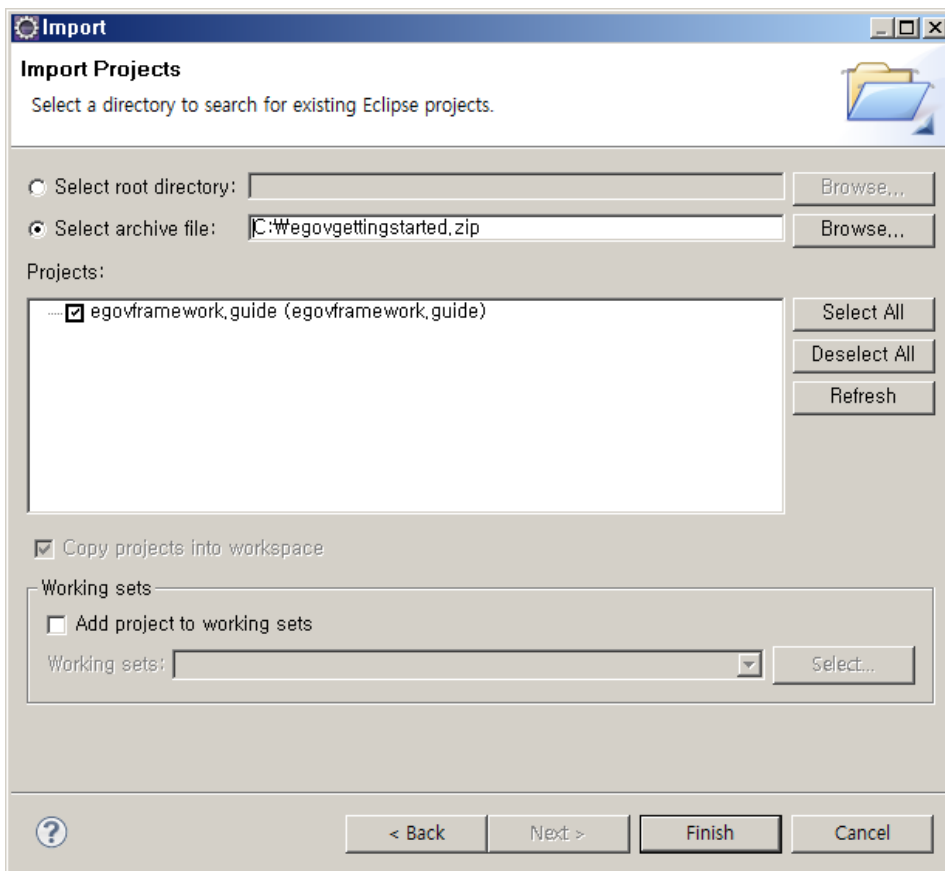
- HelloWorld 프로젝트 파일을 다운로드 받아서 임의의 디렉토리에 저장한다. (다운로드 받은 파일의 압축을 해제할 필요는 없다.)
- 구현도구에서 File>Import.. 메뉴를 선택한다.



- Import wizard에서 General>Existing Projects into Workspace 를 선택한다.
- next 버튼을 클릭한다.



- Import Projects에서 select archive file 항목을 선택하고 제공한 egovGettingStarted.zip 파일을 지정한다.
- Finish 버튼을 클릭한다.

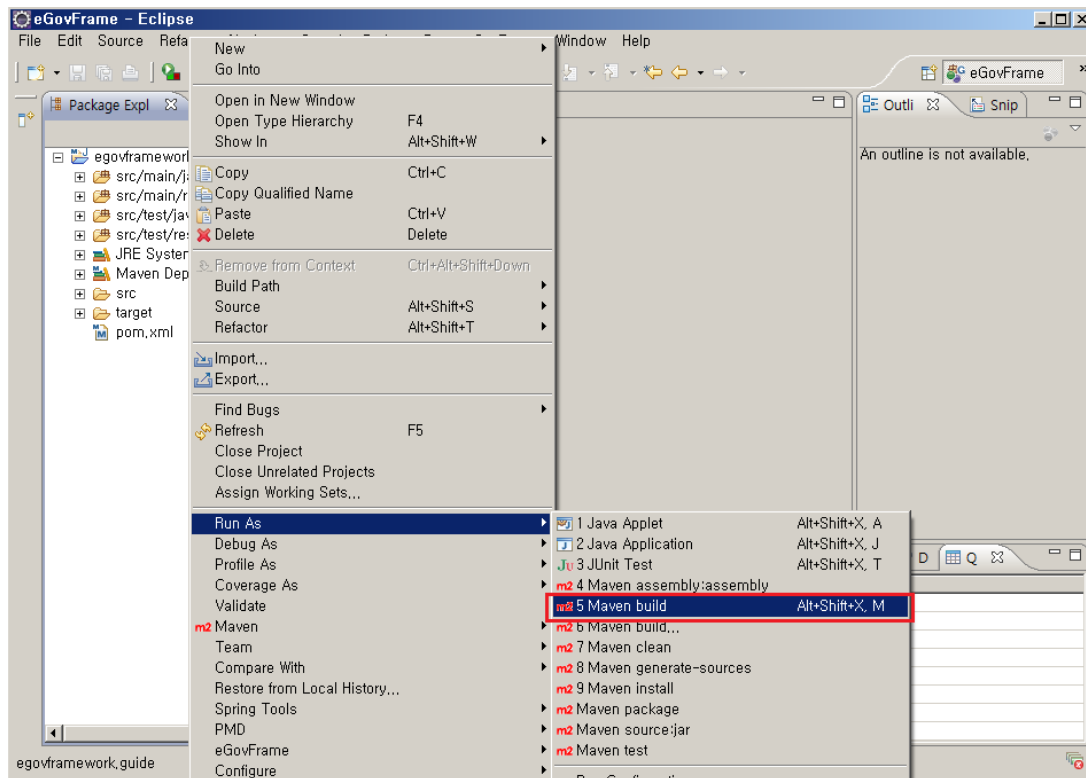


Maven을 이용한 빌드

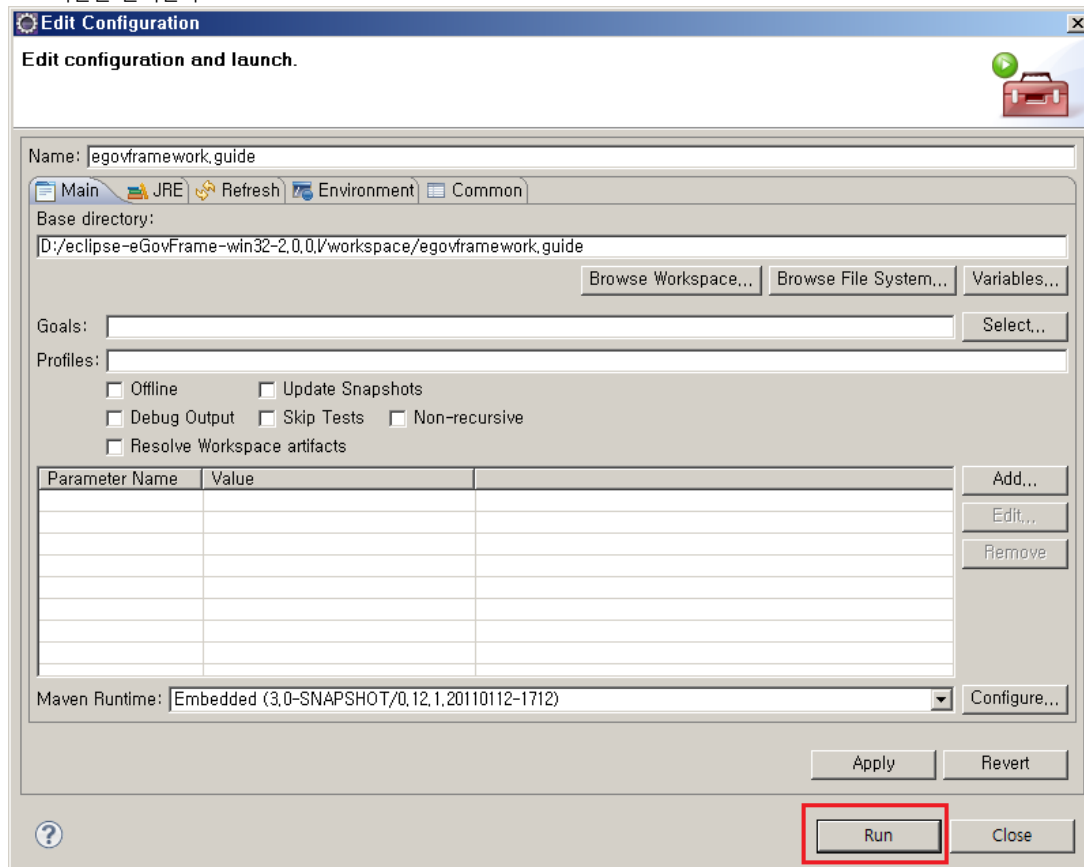
구현도구의 기능을 이용하여 컴파일, 테스트, 패키징 기능을 각각 수행할 수 있는 기능을 제공하고 있으나 Maven기반의 통합 빌드의 기능을 사용할 수도 있다. 여기서는 통합빌드를 수행해 보고 성공적인 빌드 결과를 확인한다.

프로젝트 빌드순서

- 개발환경에서 egovframework.guide 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 Run As>Maven Build를 클릭한다.



- Run 버튼을 클릭한다.



- 콘솔창에서 Maven 빌드의 결과를 확인한다. 메이븐 빌드는 프로젝트의 컴파일, 테스트, 패키징을 모두 수행하고 그 결과를 제공한다.

```
[INFO] Scanning for projects...
[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] It is highly recommended to fix these problems because they threaten the stability of your build.
[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

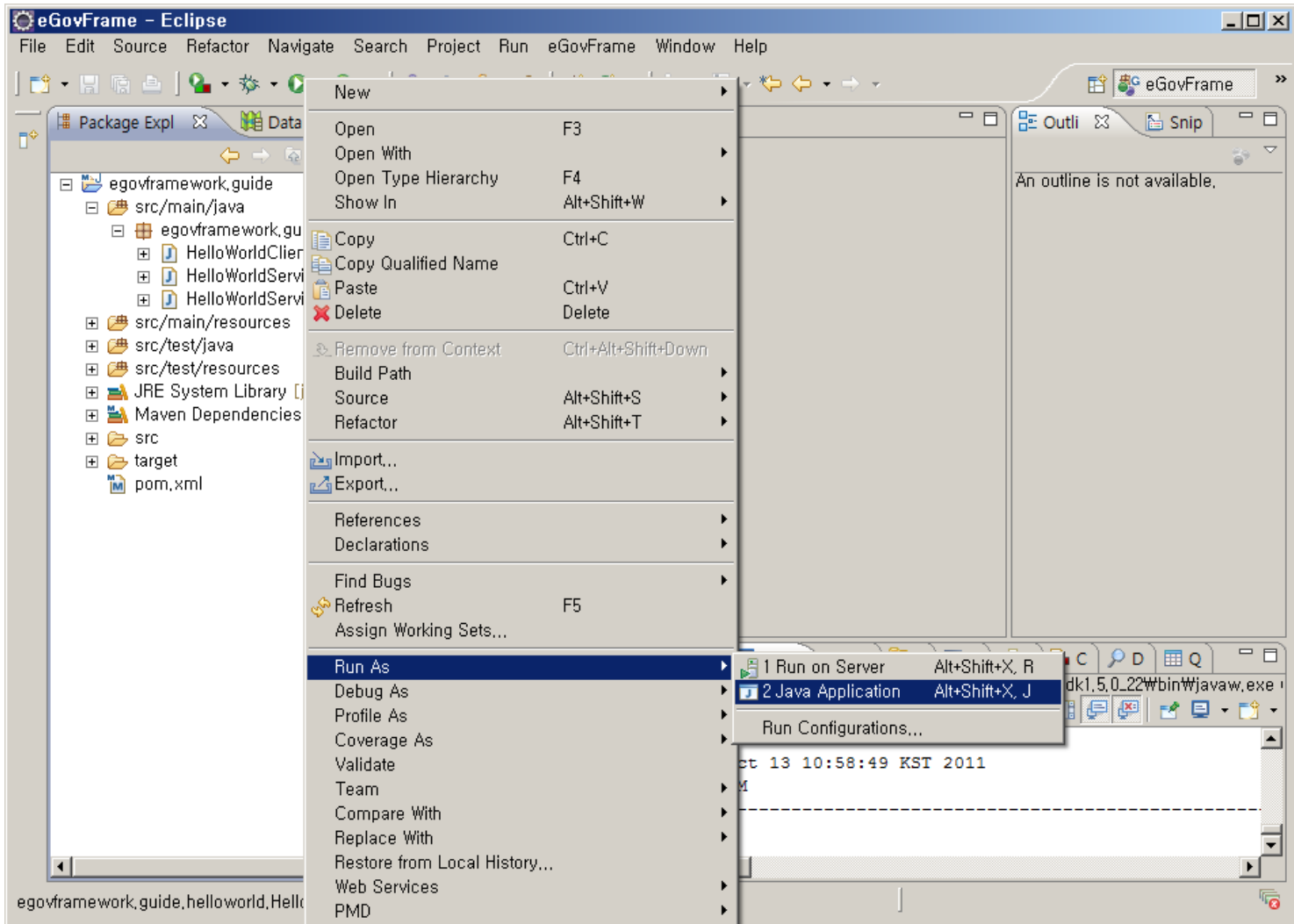
Results :

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

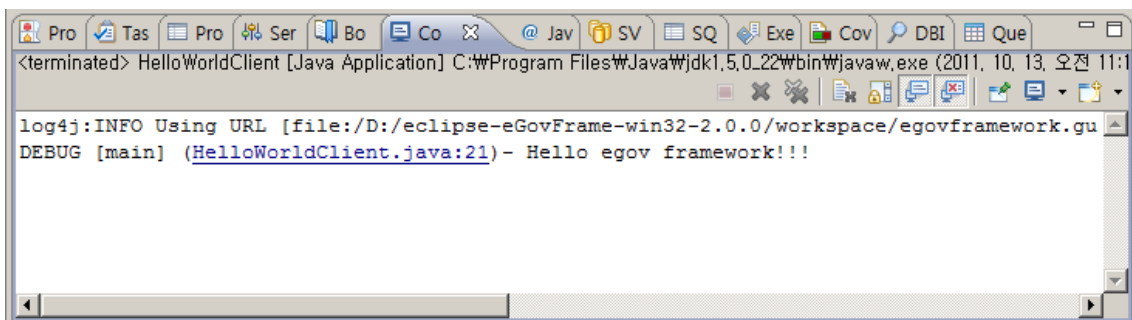
[INFO]
[INFO] --- maven-jar-plugin:2.3.1:jar (default-jar) @ guide ---
[INFO] Downloading: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-archiver/1.0/plexus-archiver-1.0.jar
[INFO] Downloading: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-io/1.0/plexus-io-1.0.jar
[INFO] Downloaded: http://repo1.maven.org/maven2/org/codehaus/plexus/plexus-io/1.0/plexus-io-1.0.jar (50 KB at 35.4 KB/sec)
[INFO] 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] -----
[INFO] Total time: 9.147s
[INFO] Finished at: Thu Oct 13 10:58:49 KST 2011
[INFO] Final Memory: 4M/8M
[INFO] -----
```

HelloWorld 실행순서

- 프로젝트의 src/main/java에서 HelloWorldClient.java를 마우스 오른쪽 버튼으로 클릭하고 Run As>Java Application 을 클릭한다.



- Console창에서 다음과 같은 실행 결과를 확인 할 수 있다.

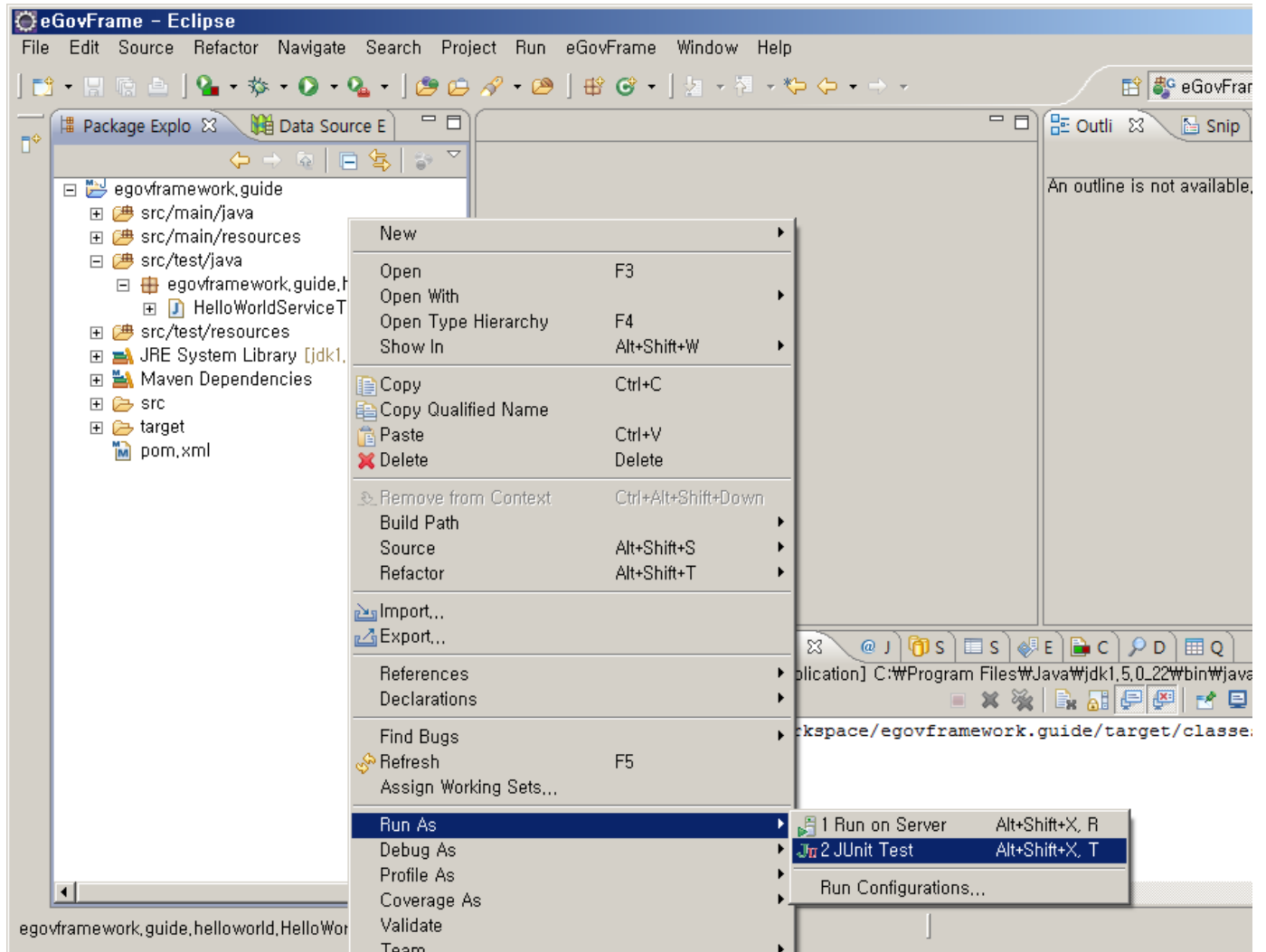


HelloWorld 테스트 실행

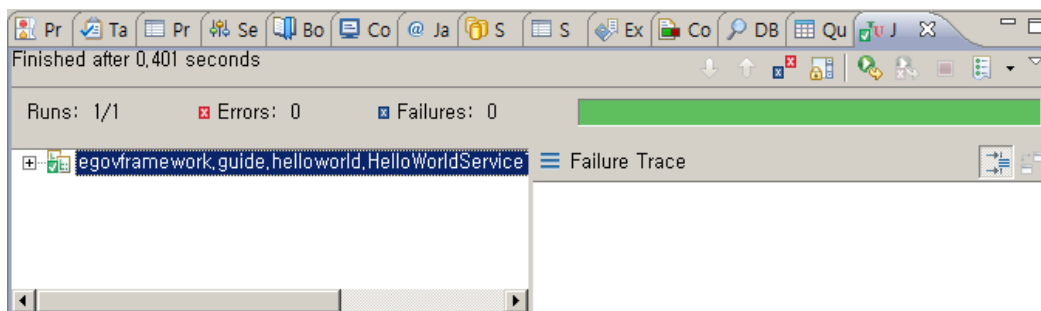
HelloWorld 프로젝트는 JUnit Test Framework 기반의 Test Case를 포함하고 있다. Test Case는 HelloWorldServiceImpl 클래스의 sayHello 메서드의 실행결과에 대한 결과값을 확인하도록 구현되어 있다. 제공한 구현도구에서 아래의 순서로 테스트를 수행한다.

HelloWorld 테스트 실행순서

- 프로젝트의 src/test/java에서 HelloWorld 서비스의 junit test case(HelloWorldServiceTest.java)를 마우스 오른쪽 버튼으로 클릭하고 Run As>JUnit test 을 클릭한다.



- JUnit 결과창에서 다음과 같이 테스트(testSayHello)의 수행시간 및 성공여부를 확인 할 수 있다.



Step 3. 자세히 들여다 보기

서비스 인터페이스 클래스

다음은 HelloWorld 서비스의 인터페이스 클래스인 HelloWorldService이다. "String sayHello()" 메서드를 선언한다.

```
package egovframework.guide.helloworld;

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

서비스 구현 클래스

다음은 HelloWorld 서비스의 구현 클래스인 HelloWorldServiceImpl이다. 서비스 인터페이스 클래스인 HelloWorldService를 구현하고 있다.

```
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 + "!!!" ;
    }
}
```

서비스 속성 정의 파일

다음은 HelloWorld 서비스의 속성정의 파일 이다. helloworld를 name 속성으로 서비스 구현클래스를 정의하고 있으며 해당 서비스의 name 속성을 "egov framework"로 선언하였다.

```
<?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>
```

클라이언트 클래스

다음은 HelloWorld 서비스를 실행하기 위한 클라이언트 클래스 이다.

```
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 = (HelloWorldService)context.getBean("helloworld");

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

테스트 클래스

다음은 HelloWorld 서비스 클래스를 테스트하기 위한 단위 테스트 클래스이다.

```
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 = (HelloWorldService)context.getBean("helloworld");
        assertEquals("Hello egov framework!!!", helloworld.sayHello());
    }
}
```