To Create an OSGi Project Using the Eclipse IDE

1. Open eclipse.
2. Go to File -> New -> Other.. -> Plug-in Development
   Once there select Plug-in Project and click Next > (see Fig. 1)

   ![Select a wizard](image)

   Fig. 1 – Plug-in Project

3. Select a name for your project and change the Target Platform to “an OSGi framework: “ to and choose standard from the dropdown menu, then click Next > (see Fig. 2)
Fig. 2 – Plug-in Project details

4. In the next screen, under Options uncheck the “Generate an activator, a Java class”, and click Next (see Fig. 3). This is needed because it is recommended to use a declarative service.
5. On the next screen simply click Finish (see Fig. 4).

6. **Note:** when using declarative service, you need to create a component.xml in OSGI-INF folder.
7.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0"
  activate="startup" deactivate="shutdown" modified="modif"
  name="org.knopflerfish.test1">
  <implementation class="org.knopflerfish.test1" />
  <reference bind="setQuote" cardinality="1..1"
    interface="de.vogella.osgi.quote.IQuoteService"
    name="IQuoteService"
    policy="static"
    bind = "setQuote" 
    unbind="unsetQuote" />
</scr:component>
```
We hope you use the series of method names: "startup shutdown modif".
And “setXXX and unsetXX”

Component.java  example

```java
package org.kf.test.dateAsayconsumer;

import java.util.Date;

import org.kf.test.ds.seed.DateService;
```
import org.kf.test.say.Say;

public class Component {

    DateService dateService;
    Say say;
    private void startup() {
        log("startup()");
        //TODO:
        log("----startup()---------");
    }

    private void shutdown() {
        log("shutdown()");
        //TODO:
        log("----shutdown()--------");
    }

    private void modif() {
        log("modif()");
        //TODO:
        log("----modif()--------");
    }

    /**
     * Called by the Declarative Service component finds a registered
     * DateService as specified in the component.xml
     */
    public synchronized void setDateService(DateService dateService) {
        log("setDateService.DateConsumer");
    }
}
this.dateService = dateService;

log("---Component.setDateService().DateConsumer");
}

/**
 * Called by the Declarative Service component notices an unregistered
 * DateService as specified in the component.xml
 */

public synchronized void unsetDateService() {
    log("unsetDateService.org.kf.test.Dateconsumer");
    this.dateService= null;
    log("---unset");
}

private void log(String message) {
    System.out.println(message);
}

Install an OSGi environment:

1. eclipse-jee-juno-SR2-win32.zip
2. knopflerfish_osgi_sdk_4.0.0.beta-1.jar
3. we do not need Knopflerfish Eclipse Plugin.

Export a bundle

1. Export
2.  
3.  next  

<table>
<thead>
<tr>
<th>Destination</th>
<th>Options</th>
<th>JAR Signing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Export source: Generate source bundles
- Package plug-ins as individual JAR archives
- Qualifier replacement (default value is today's date): 201308012021
- Save as Ant script: 
- Allow for binary cycles in target platform
- Use class files compiled in the workspace

4. Then drag bundles into knopflerfish framework and start them.

5. such icon Circle means it is a component generated from declarative services.

**Bundle a standard library jar for OSGi lib.**

jdbc.Nosql  javex.swing.Sw...
1. File -> New -> Other.. ->
2. Add External..
3. Next

4. 

5. It adds export-package automatically.
6. Then export this bundle and use it in knopflerfish framework.
Some tips:

1. Component class and clazzImpl are located into different packages.
2. Declarative service is strongly recommended. Both listener and tracker are deprecated.
3. Bundles can run in eclipse and knopflerfish framework when using Activator. While declarative service bundles can only run in knopflerfish framework.
4. Lazy start is not recommended. But it works.
5. Check "This component is enabled when started." in component.xml
6. Please use clazzImpl implement clazz, use @Override. clazz is in org.kf.test.clazz package, and clazzImpl is in org.kf.test.clazz.impl package, and provide service and reference using "INTERFACE" rather than "CLASS".
7. In file Service-Component: OSGI-INF/component.xml activate="activate" deactivate="deactivate" modified="modified";  bind="setXxxx" ;  unbind="unsetXxxx"
8. SWT is recommended when using OSGI. Using swing may always have problems when close the swing framework.
Some learning materials:

1. Knopflerfish Manuals and Documentation
   http://www.knopflerfish.org/documentation.html
   please read OSGi R4 v4.3 or v5.
5. OSGi Modularity - Tutorial
   http://www.vogella.com/articles/OSGi/article.html
6. OSGi in Action: Creating Modular Applications in Java
### Possible bugs and solutions:

<table>
<thead>
<tr>
<th>Possible bugs and solutions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>to solve &quot;Got java.net.BindException: Address already in use: JVM_Bind when invoking &quot;private void OSGI.Component.startup() throws java.lang.reflect.InvocationTargetException,java.lang.InterruptedException,java.io.IOException&quot; in: class OSGI.Component&quot;</td>
</tr>
<tr>
<td>1.kill the process javaw.exe</td>
</tr>
<tr>
<td>2.and move that 3 bundles to another folder.</td>
</tr>
<tr>
<td>3.then restart knopflerfish framework, reload bundles and run.</td>
</tr>
<tr>
<td>4.so do not have to restart system.</td>
</tr>
<tr>
<td>A severe fault of Knopflerfish in eclipse. It can not bundle the OSGI-INF/component.xml into jar automatically.</td>
</tr>
<tr>
<td>Results in &quot; Resource not found: OSGI-INF/component.xml &quot; !!!</td>
</tr>
<tr>
<td>Solution: New a Component Definition by eclipse, do not copy component.xml into project OSGI-INF folder.</td>
</tr>
<tr>
<td>Please always clean, restart framework, delete and re-new the plug-in project when it is weird to debug.</td>
</tr>
</tbody>
</table>