GWT - APPLICATIONS

Before we start with creating actual *HelloWorld* application using GWT, let us see what are the actual parts of a GWT application. A GWT application consists of following four important parts out of which last part is optional but first three parts are mandatory:

- **Module descriptors**
- **Public resources**
- **Client-side code**
- **Server-side code**

Sample locations of different parts of a typical gwt application **HelloWord** will be as shown below:

Name	Location
Project root	HelloWorld/
Module descriptor	src/com/tutorialspoint/HelloWorld.gwt.xml
Public resources	src/com/tutorialspoint/war/
Client-side code	src/com/tutorialspoint/client/
Server-side code	src/com/tutorialspoint/server/

Module Descriptors

A module descriptor is the configuration file in the form of XML which is used to configure a GWT application. A module descriptor file extension is ***.gwt.xml**, where * is the name of the application and this file should reside in the project's root. Following will be a default module descriptor **HelloWorld.gwt.xml** for a **HelloWorld** application:

```
<?xml version="1.0" encoding="utf-8"?>
<module rename-to='helloworld'>
   <!-- inherit the core web toolkit stuff.                      -->
   <inherits name='com.google.gwt.user.user'/>

   <!-- inherit the default gwt style sheet.                     -->
   <inherits name='com.google.gwt.user.theme.clean.Clean'/>

   <!-- specify the app entry point class.                       -->
   <entry-point class='com.tutorialspoint.client.HelloWorld'/>

   <!-- specify the paths for translatable code                  -->
   <source path='...'/>
   <source path='...'/>

   <!-- specify the paths for static files like html, css etc.   -->
   <public path='...'/>
   <public path='...'/>

   <!-- specify the paths for external javascript files          -->
   <script src="js-url" />
   <script src="js-url" />

   <!-- specify the paths for external style sheet files         -->
   <stylesheet  src="css-url" />
   <stylesheet  src="css-url" />
</module>
```

Following is the brief detail about different parts used in module descriptor.

S.N.	Nodes & Description
1	**<module rename-to="helloworld">** This provides name of the application.
2	**<inherits name="logical-module-name" />**This adds other gwt module in application just like import does in java applications. Any number of modules can be inherited in this manner.
3	**<entry-point />**This specifies the name of class which will start loading the GWT Application. Any number of entry-point classes can be added and they are called sequentially in the order in which they appear in the module file. So when the onModuleLoad of your first entry point finishes, the next entry point is called immediately.
4	**<source path="path" />**This specifies the names of source folders which GWT compiler will search for source compilation.
5	**<public path="path" />**The public path is the place in your project where static resources referenced by your GWT module, such as CSS or images, are stored. The default public path is the public subdirectory underneath where the Module XML File is stored.
6	**<script src="js-url" />**Automatically injects the external JavaScript file located at the location specified by src.
7	**<stylesheet src="css-url" />**Automatically injects the external CSS file located at the location specified by src.

Public resources

These are all files referenced by your GWT module, such as Host HTML page, CSS or images. The location of these resources can be configured using <public path="path" /> element in module configuration file. By default, it is the public subdirectory underneath where the Module XML File is stored.

When you compile your application into JavaScript, all the files that can be found on your public path are copied to the module's output directory.

The most important public resource is host page which is used to invoke actual GWT application. A typical HTML host page for an application might not include any visible HTML body content at all but it is always expected to include GWT application via a <script.../> tag as follows:

```
<html>
<head>
<title>Hello World</title>
   <link rel="stylesheet" href="HelloWorld.css"/>
   <script language="javascript" src="helloworld/helloworld.nocache.js">
   </script>
</head>
<body>

<h1>Hello World</h1>
<p>Welcome to first GWT application</p>

</body>
</html>
```

Following is the sample style sheet which we have included in our host page:

```
body {
   text-align: center;
   font-family: verdana, sans-serif;
}
h1 {
  font-size: 2em;
  font-weight: bold;
  color: #777777;
  margin: 40px 0px 70px;
  text-align: center;
}
```

Client-side code

This is the actual Java code written implementing the business logic of the application and that the GWT compiler translates into JavaScript, which will eventually run inside the browser. The location of these resources can be configured using <source path="path" /> element in module configuration file.

For example **Entry Point** code will be used as client side code and its location will be specified using <source path="path" />. A module **entry-point** is any class that is assignable to **EntryPoint** and that can be constructed without parameters. When a module is loaded, every entry point class is instantiated and its **EntryPoint.onModuleLoad** method gets called. A sample HelloWorld Entry Point class will be as follows:

```
public class HelloWorld implements EntryPoint {
   public void onModuleLoad() {
      Window.alert("Hello, World!");
   }
}
```

Server-side code

This is the server side part of your application and its very much optional. If you are not doing any backend processing with-in your application then you do not need this part, but if there is some processing required at backend and your client-side application interact with the server then you will have to develop these components.

Next chapter will make use of all the above mentioned concepts to create HelloWorld application using Eclipse IDE.