Google code

## Google App Engine

# Creating a Project

App Engine Java applications use the Java Servlet standard for interacting with the web server environment. An application's files, including compiled classes, JARs, static files and configuration files, are arranged in a directory structure using the WAR standard layout for Java web applications. You can use any development process you like to develop web servlets and produce a WAR directory. (WAR archive files are not yet supported by the SDK.)

## The Project Directory

For this tutorial, we will use a single directory named `Guestbook/` for all project files. A subdirectory named `src/` contains the Java source code, and a subdirectory named `war/` contains the complete application arranged in the WAR format. Our build process compiles the Java source files and puts the compiled classes in the appropriate location in `war/`.

The complete project directory looks like this:

```
Guestbook/
  src/
    ...Java source code...
    META-INF/
      ...other configuration...
  war/
    ...JSPs, images, data files...
    WEB-INF/
      ...app configuration...
      lib/
        ...JARs for libraries...
      classes/
        ...compiled classes...
```

If you are using Eclipse, create a new project by clicking the New Web Application Project button in the toolbar:  Give the project a "Project name" of `Guestbook` and a "Package" of `guestbook`. Uncheck "Use Google Web Toolkit," and ensure "Use Google App Engine" is checked. See Using the Google Plugin for Eclipse for more information. The wizard creates the directory structure, and the files described below.

If you are not using Eclipse, create the directory structure described above. As you read each of the files described in this section, create the files using the given locations and names.

You can also copy the new project template included with the SDK, in the `appengine-java-sdk/demos/new_project_template/` directory.

## The Servlet Class

App Engine Java applications use the Java Servlet API to interact with the web server. An HTTP servlet is an application class that can process and respond to web requests. This class extends either the javax.servlet.GenericServlet class or the javax.servlet.http.HttpServlet class.

Our guest book project begins with one servlet class, a simple servlet that displays a message.

If you are not using the Eclipse plugin, create the directories for the path `src/guestbook/`, then create the servlet class file described below.

In the directory `src/guestbook/`, a file named `GuestbookServlet.java` has the following contents:

```
package guestbook;

import java.io.IOException;
import javax.servlet.http.*;

public class GuestbookServlet extends HttpServlet {
    public void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws IOException {
        resp.setContentType("text/plain");
```