

### **Ex no 3: Installing and Running the Google App Engine on Windows**

#### **Aim:**

To perform the installation of the Google App Engine Software Development Kit (SDK) on a Microsoft Windows and running a simple “hello world” application.

#### **Procedure:**

- The App Engine SDK allows you to run Google App Engine Applications on your local computer. It simulates the run--time environment of the Google App Engine infrastructure.

#### **Step1: To install python**

##### **Pre--Requisites: Python 2.5.4**

If you don't already have Python 2.5.4 installed in your computer, download and Install Python 2.5.4 from:

<http://www.python.org/download/releases/2.5.4/>

#### **Step 2: to install Google App Engine**

##### **Download and Install**

You can download the Google App Engine SDK by going to:

<http://code.google.com/appengine/downloads.html> and download the appropriate installpackage.

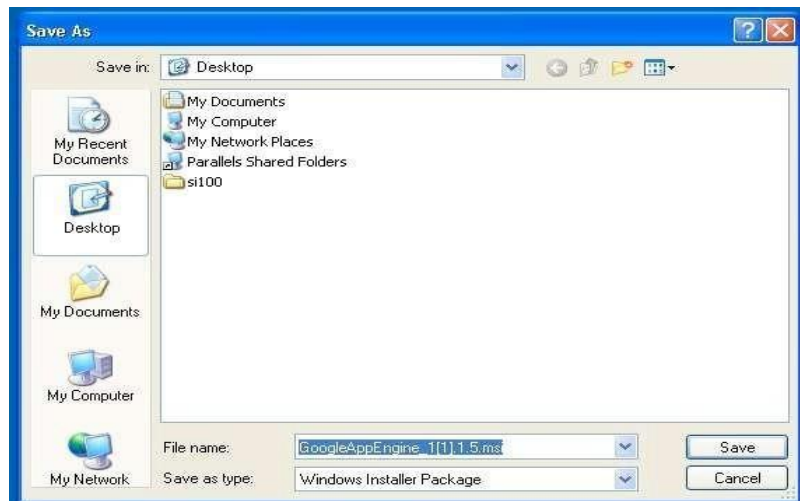
## Download the Google App Engine SDK

Before downloading, please read the [Terms](#) that govern your use of the App Engine SDK.

Please note: The App Engine SDK is under **active development**, please keep this in mind as you explore its capabilities. See the [SDK Release Notes](#) for the information on the most recent changes to the App Engine SDK. If you discover any issues, please feel free to notify us via our [Issue Tracker](#).

Platform	Version	Package	Size	SHA1 Checksum
Windows	1.1.5 - 10/03/08	<a href="#">GoogleAppEngine_1.1.5.msi</a>	2.5 MB	e974312b4aefc0b3873ff0d93eb4c525d5e88c30
Mac OS X	1.1.5 - 10/03/08	<a href="#">GoogleAppEngineLauncher-1.1.5.dmg</a>	3.6 MB	f62208ac01c1b3e39796e58100d5f1b2f052d3e7
Linux/Other Platforms	1.1.5 - 10/03/08	<a href="#">google_appengine_1.1.5.zip</a>	2.6 MB	cbb9ce817bdabf1c4f181d9544864e55ee253de1

Download the Windows installer – the simplest thing is to download it to your Desktop or another folder that you remember.



Double Click on the **GoogleApplicationEngine** installer.



Click through the installation wizard, and it should install the App Engine. If you do not have Python 2.5, it will install Python 2.5 as well.

Once the install is complete you can discard the downloaded installer

After installation Google app engine looks that,



### **Step 3: Making of the FirstApplication**

Now we need to create a simple application. We could use the “+” option to have the launcher make us an application – but instead we will do it by hand to get a better sense of what is going on.

Make a folder for your Google App Engine applications. I am going to make the Folder on my Desktop called “**apps**” – the path to this folder is:

**C:\Documents and Settings\csev\Desktop\apps**

And then make a sub--folder in within apps called “**ae--01--trivial**”--the path to this folder would be:

C:\ Documents and Settings \csev\Desktop\apps\ae--01--trivial

Using a text editor such as JEdit ([www.jedit.org](http://www.jedit.org)), create a file called **app.yaml** in the **ae--01--trivial** folder with the following contents:

```
application: ae-01-  
trivial version: 1  
runtime: python  
api_version: 1  
handlers:  
- url: /.  
  script: index.py
```

Then create a file in the **ae--01--trivial** folder called **index.py** with three lines in it:

```
print 'Content-  
Type: text/plain' print ' '  
print 'Hello there Chuck'
```

Step 4: Run the program

```
Administrator: Google Cloud SDK Shell - dev_appserver.py app.yaml
Welcome to the Google Cloud SDK! Run "gcloud -h" to get the list of available commands.
---
C:\Program Files (x86)\Google\Cloud SDK>dev_appserver.py app.yaml

Updates are available for some Cloud SDK components. To install them,
please run:
$ gcloud components update

This action requires the installation of components: [app-engine-
python, cloud-datastore-emulator]

Your current Cloud SDK version is: 316.0.0
Installing components from version: 316.0.0

These components will be installed.


| Name                         | Version | Size     |
|------------------------------|---------|----------|
| Cloud Datastore Emulator     | 2.1.0   | 18.4 MiB |
| gRPC python library          | 1.20.0  |          |
| gRPC python library          | 1.20.0  | 1.5 MiB  |
| gcloud app Python Extensions | 1.9.91  | 6.1 MiB  |


For the latest full release notes, please visit:
https://cloud.google.com/sdk/release\_notes
Do you want to continue (Y/n)? y

Creating update staging area
```

### Output:

Once you have selected your application and press **Run**. After a few moments your application will start and the launcher will show a little green icon next to your application. Then press **Browse** to open a browser pointing at your application which is running at **<http://localhost:8080/>**

Paste **http://localhost:8080** into your browser and you should see your application as follows:



**Result:**

Thus the python application program was executed.

**Ex no 4: Use GAE launcher to launch the web applications on Windows**

**Aim:**

To deploy the GAE launcher on a Microsoft Windows and running a simple “hello world” application.

## **Procedure:**

### **Step 1. Download the basic housekeeping stuff**

No matter what platform you build products on, there is always some housekeeping stuff you need to put in place before you can hit the ground running. And deploying apps within the Google App Engine is no exception.

1. Download [Python 2.7](#)

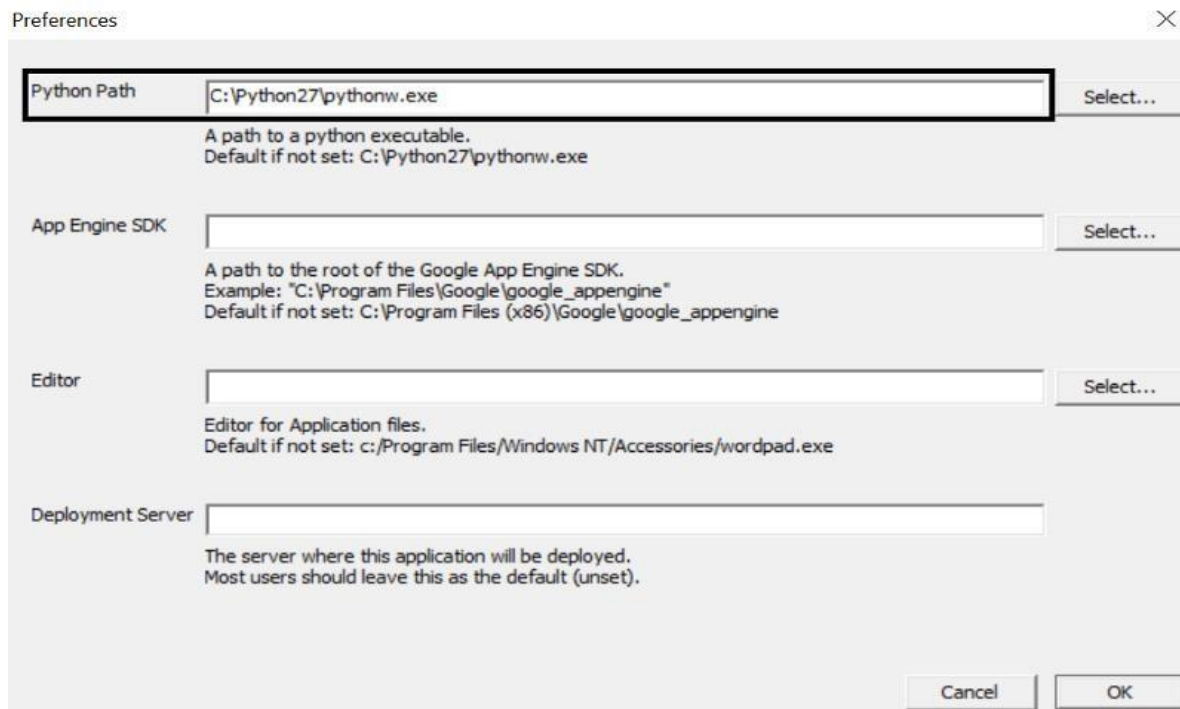
As of when this article was written, the Google App Engine [standard environment supports Python only upto version 2.7](#). However, it is only a matter of time before support for Python 3.x is added. You can check the App Engine docs for the latest info.

2. Download [Google Cloud SDK](#)

This will allow you to fork apps onto your local machine, make changes (edit and develop the app), and deploy your app back to the cloud.

3. Set the Python path in the Google App Engine launcher

After downloading the SDK, launch the App Engine launcher, go to Edit -> Preferences and make sure you set the path for where you installed Python in step 1 above.



Set the Python path in Google App Engine launcher

That's all you need. Your local machine should now be ready to build webapps.

## Step 2. App Engine sign-up

This is often the most confusing part of the entire setup. Things you should know when you sign-up:

1. Currently, App Engine offers a free trial for one year.
2. The trial includes \$300 of credit that can be used during the one year trial period.
3. You will need to add a credit card to sign-up (for verification purposes).
4. You will not be charged during the sign-up process.
5. You will not be charged during the trial period as long as you do not cross the credit limit offered.

Here are the steps you need to follow to sign-up:

1. Go to the [Google Cloud](#) landing page
2. Follow the sign-up process and go to your App Engine dashboard



Most of the hard work is complete after a successful sign-up.

### Step 3. Create a new project

The next step is to create a new Python project that you can work on. Follow the screenshots below to create a new project.

Launch the new project wizard.

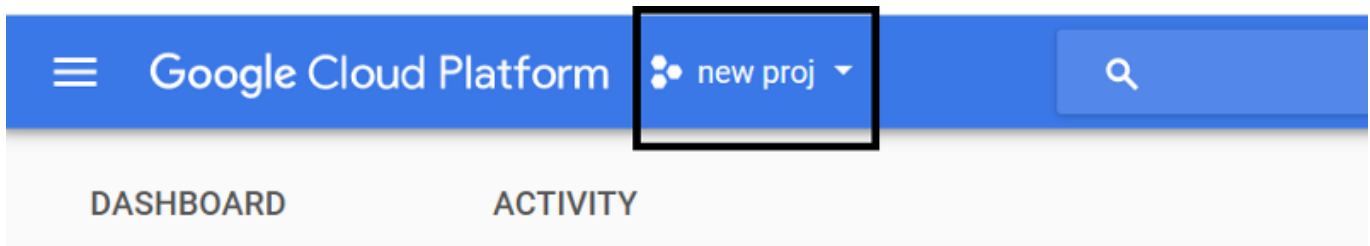


Image courtesy.

<https://console.cloud.google.com/home>

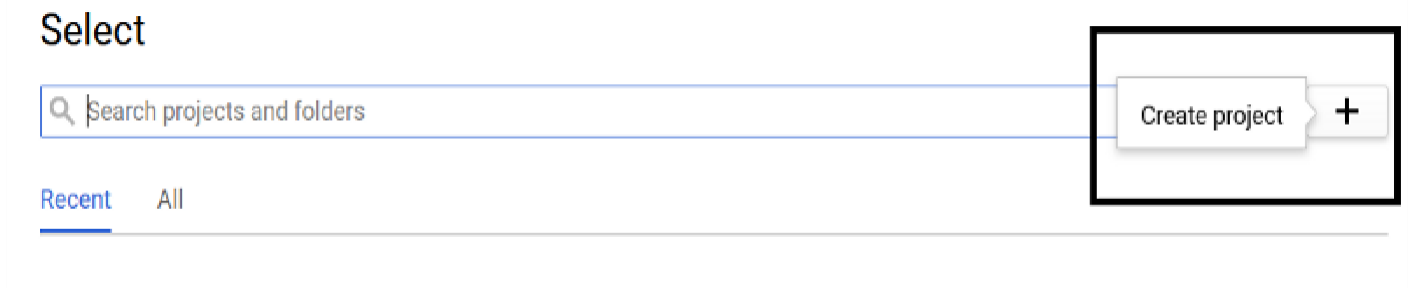




Image courtesy <https://console.cloud.google.com/home>


Give your app a name and make a note of your project ID.

## New Project

 You have 24 projects remaining in your quota. [Learn more.](#)

Project name 

myHelloWorld

Your project ID will be myhelloworld-201222  [Edit](#)

Create

Cancel

Image courtesy.

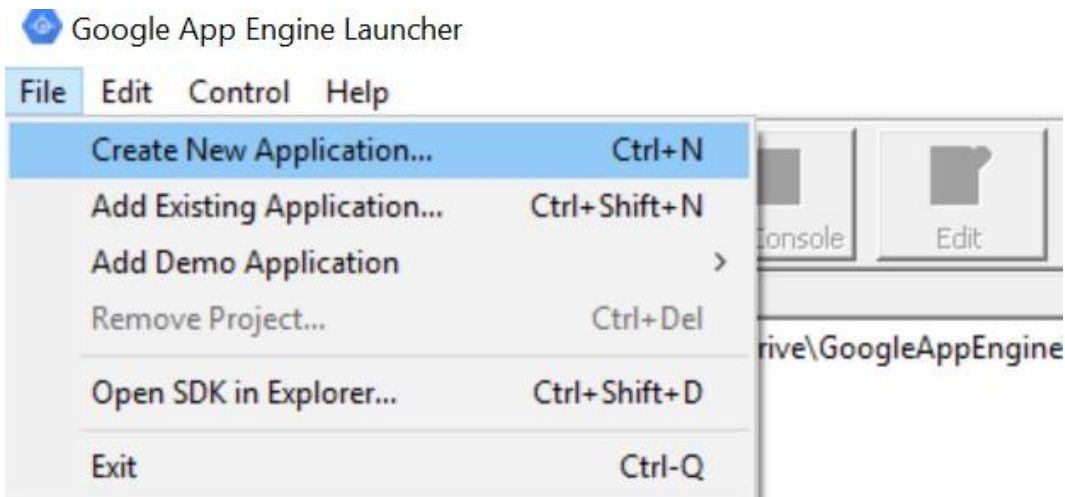
<https://console.cloud.google.com/home>

Hit the create button and Google should take a few minutes to set up all that is necessary for your newly created app.

### Step 4. Fork the app to develop it locally

The next step in the process is to fork the app on your local machine. This will allow you to make changes to the app locally and deploy it whenever you wish to.

Go to Google App Engine launcher and create a new application.



Enter the project ID of your newly created app. Also, provide the folder (local destination) where you wish to store the app locally. Make sure you select the Python 2.7 as your runtime engine.

#### Add New Application



Application Settings

Application Name: myhelloworld-201222

Parent Directory:

Runtime: Python 2.7 ▼

Port: 9080

Admin Port: 8001

Hit the create button, and you should see your app listed on the window that follows. You should also check that you now see some files in your local storage (the directory you chose in the screenshot above) after this step.

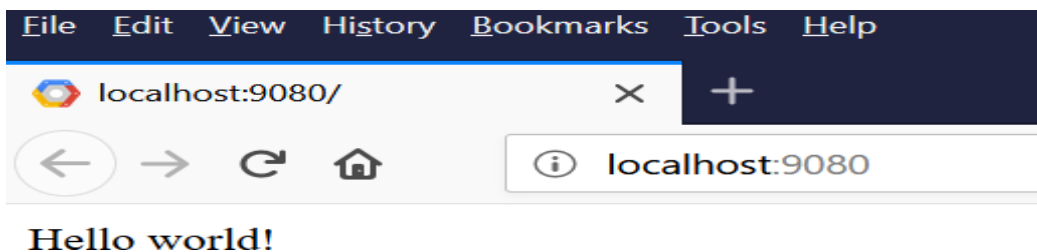
## Step 5. Run the app locally

Before you go ahead and make some changes to the app, it is important to check whether or not you have executed all the above steps correctly. This can be done by simply running the app locally.

Select the app and hit the run button on the window.



Wait for a few seconds until you can hit the **Browse** button. Once the **Browse** button becomes clickable, click it. This should take you to the browser, and you should see the hello world text appear in your browser window. Alternatively, you can manually go to the browser and use the port specified to access the app.



As long as you see the above screen, you are all set.

## Step 6. Understand the app structure

It is finally time to look at the lines of code which are running this webapp. Open your app folder in the text editor of your choice. I recommend [Sublime text](#) or [VS Code](#). However, feel free to choose the one you prefer.

Here is a description of the various files.

### **app.yaml**

This file is a basic markup file that stores information (some metadata) about the app. It is important to note the following crucial parts of the file.

1. **application**

This is the project ID which you should never change. This is the unique identifier for the app

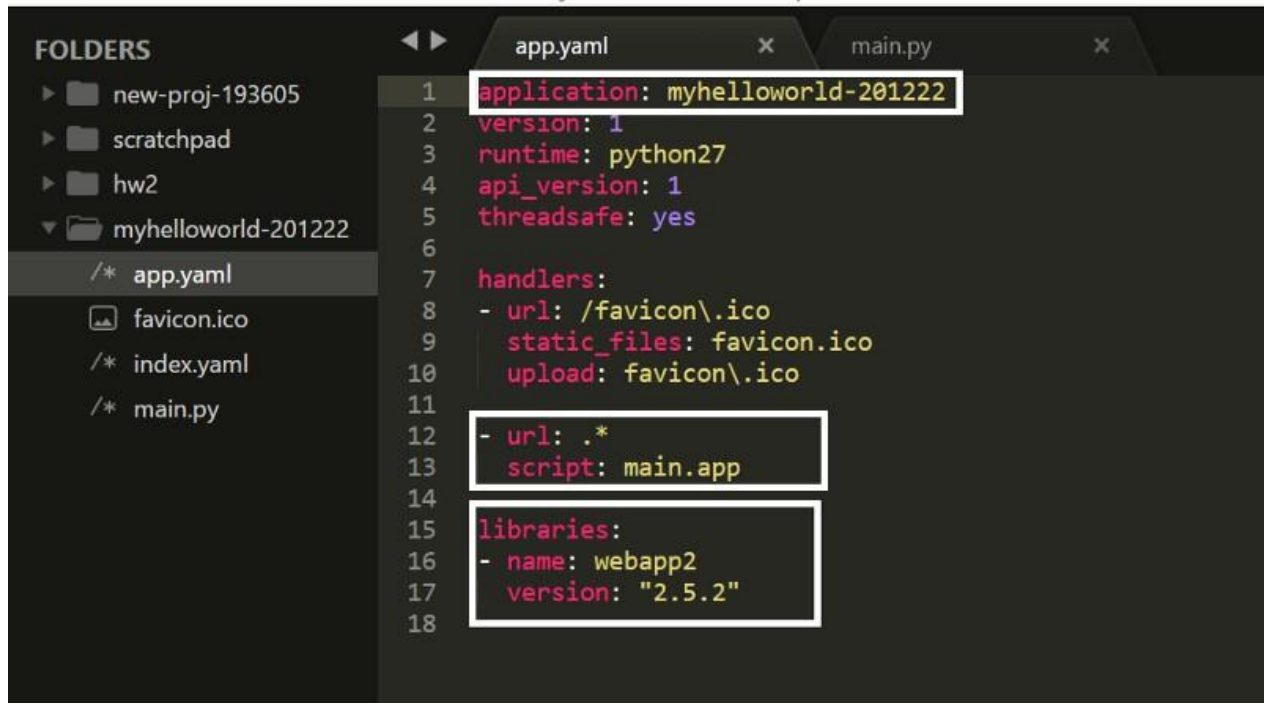
2. **url -> script**

This is the homepage for the app. In other words, this file will be rendered in your browser when you launch the app

3. **libraries**

This is where you can include external libraries to use within the webapp

File Edit Selection Find View Goto Tools Project Preferences Help



The screenshot shows an IDE window with a menu bar at the top: File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, Help. On the left is a 'FOLDERS' sidebar showing a tree structure: new-proj-193605, scratchpad, hw2, and myhelloworld-201222. The 'myhelloworld-201222' folder is expanded, showing files: app.yaml, favicon.ico, index.yaml, and main.py. The 'app.yaml' file is selected and its content is displayed in the main editor. The code is as follows:

```
1 application: myhelloworld-201222
2 version: 1
3 runtime: python27
4 api_version: 1
5 threadsafe: yes
6
7 handlers:
8 - url: /favicon\.ico
9   static_files: favicon.ico
10  upload: favicon\.ico
11
12 - url: .*
13   script: main.app
14
15 libraries:
16 - name: webapp2
17   version: "2.5.2"
18
```

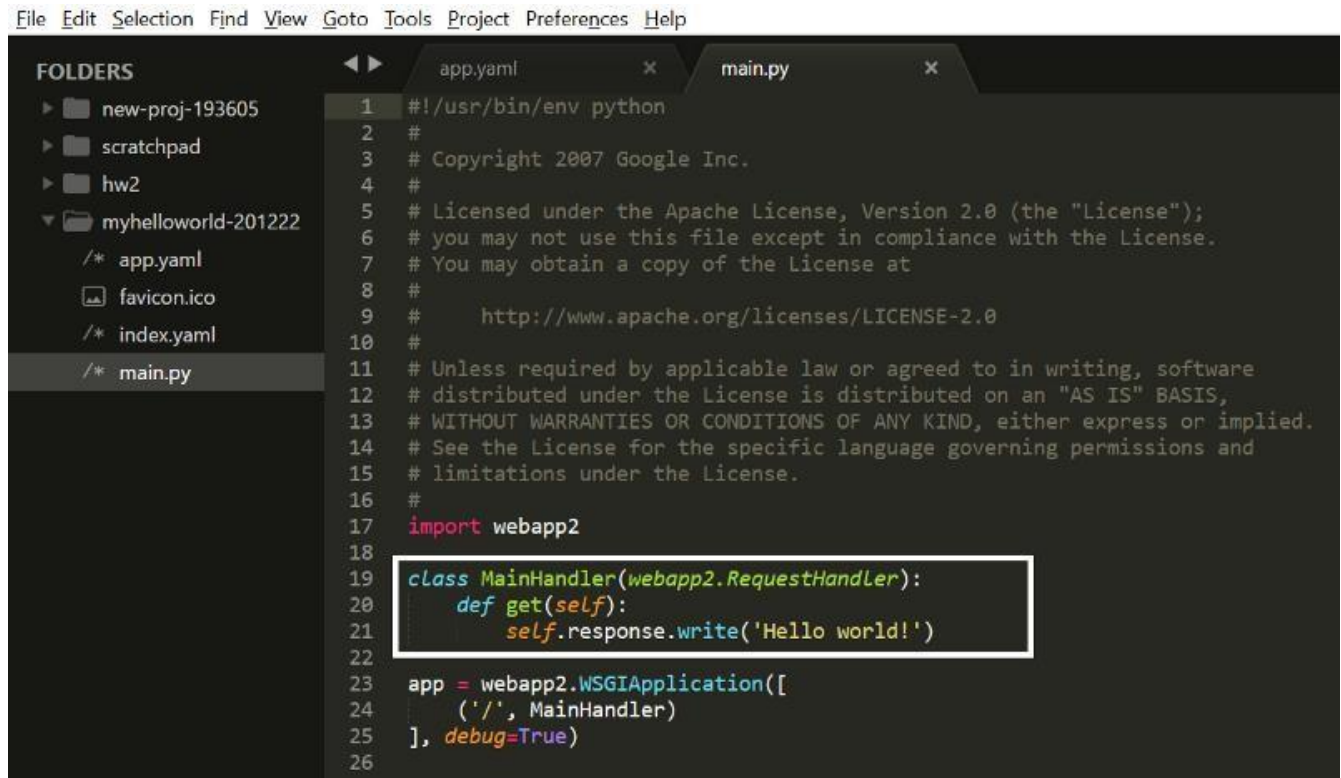
Three white rectangular boxes highlight specific parts of the code: the first box highlights line 1, the second box highlights lines 12-13, and the third box highlights lines 15-17.

app.yaml

file in the webapp folder

### main.py program

This is the homepage of the app (as discussed above). Note that the hello world text in the browser window (step 5) is due to the code you see highlighted below.



```
File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS
  ▶ new-proj-193605
  ▶ scratchpad
  ▶ hw2
  ▼ myhelloworld-201222
    /* app.yaml
    favicon.ico
    /* index.yaml
    /* main.py

1  #!/usr/bin/env python
2  #
3  # Copyright 2007 Google Inc.
4  #
5  # Licensed under the Apache License, Version 2.0 (the "License");
6  # you may not use this file except in compliance with the License.
7  # You may obtain a copy of the License at
8  #
9  #     http://www.apache.org/licenses/LICENSE-2.0
10 #
11 # Unless required by applicable law or agreed to in writing, software
12 # distributed under the License is distributed on an "AS IS" BASIS,
13 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14 # See the License for the specific language governing permissions and
15 # limitations under the License.
16 #
17 import webapp2
18
19 class MainHandler(webapp2.RequestHandler):
20     def get(self):
21         self.response.write('Hello world!')
22
23 app = webapp2.WSGIApplication([
24     ('/', MainHandler)
25 ], debug=True)
26
```

**main.py file in the webapp folder**

### Step 7. Make your changes and deploy the new app

No hello world app is ever complete without the developer changing the hello world text to something else just to make sure that everything happening behind the scenes is working as it should.

Go ahead and change the text in the above screenshot to something else.

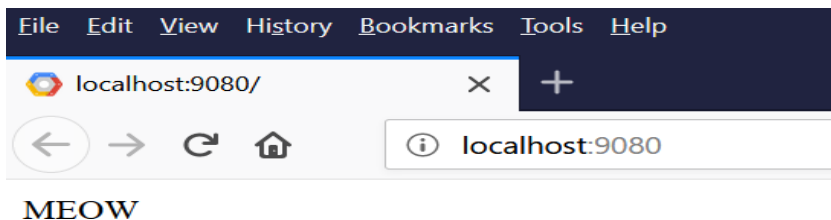
```

17  import webapp2
18
19  class MainHandler(webapp2.RequestHandler):
20      def get(self):
21          self.response.write('MEOW')
22
23  app = webapp2.WSGIApplication([
24      ('/', MainHandler)
25  ], debug=True)
26

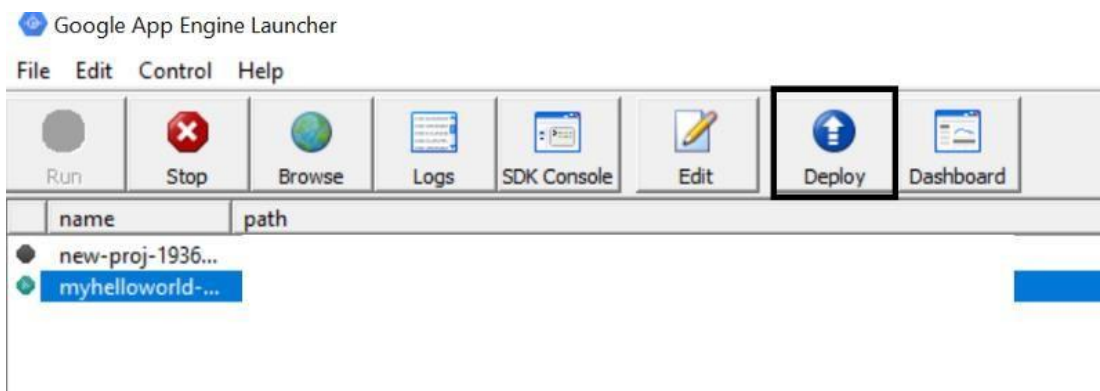
```

### Output:

Save the changes, go to the browser and refresh the page. You should see the page with the text “MEOW” displayed.



Finally, it is time to deploy your changes to the cloud to make them globally accessible via a URL. Go to the App Engine launcher, select the app, and hit the **Deploy** button.

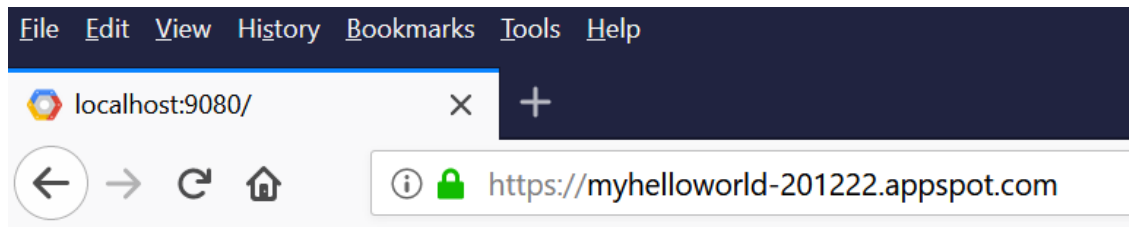




This will ensure your app gets deployed onto Google Cloud. To check whether or not everything worked just fine, go to the URL below:

**`https://<yourProjectID>.appspot.com/`**

You should see the exact same window as above, expect now, it is a URL that is globally accessible.



**Result:**

Thus the python web application is executed with the google app engine .