Installing and Running the Google Application Engine
On Windows XP
University of Michigan – Informatics

This document describes the installation of the Google Application Engine Software Development Kit (SDK) on a Macintosh and running a simple "hello world" application.

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

Download and Install

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

```
http://code.google.com/appengine/downloads.html
```

and download the appropriate install package.

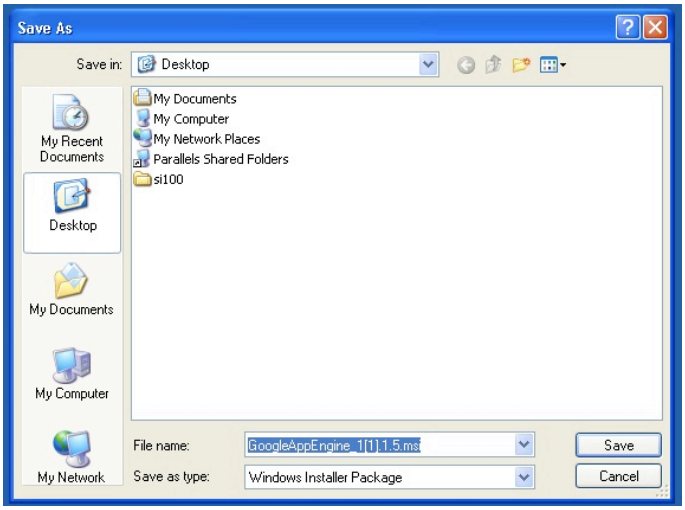
Download the Google App Engine SDK

Before downloading, please read the Terms that govern your use of the App Engine SDK.

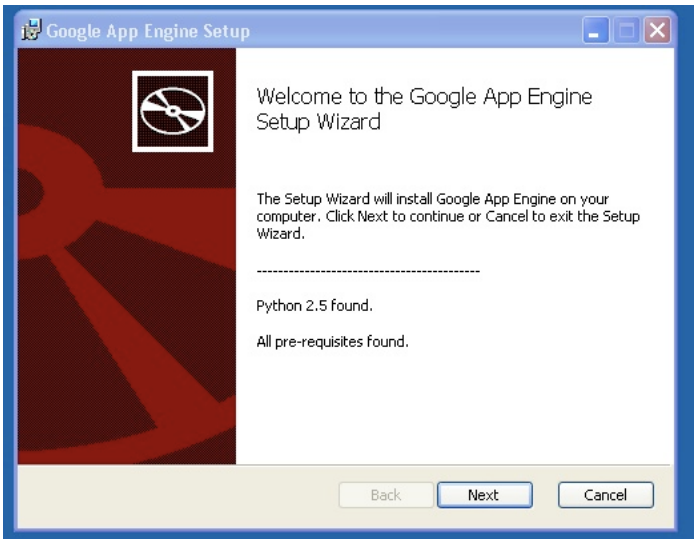
Please note: The App Engine SDK is under **active development**, please keep this in mind as you explore its capabilities. See the SDK Release Notes for the information on the most recent changes to the App Engine SDK. If you discover any issues, please feel free to notify us via our Issue Tracker.

Platform	Version	Package	Size	SHA1 Checksum
Windows	1.1.5 - 10/03/08	GoogleAppEngine_1.1.5.msi	2.5 MB	e974312b4aefc0b3873ff0d93eb4c525d5e88c30
Mac OS X	1.1.5 - 10/03/08	GoogleAppEngineLauncher-1.1.5.dmg	3.6 MB	f62208ac01c1b3e39796e58100d5f1b2f052d3e7
Linux/Other Platforms	1.1.5 - 10/03/08	google_appengine_1.1.5.zip	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

Making your First Application

Now you 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 Application 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), create a file called **app.yaml** in the **ae-01-trivial** folder with the following contents:

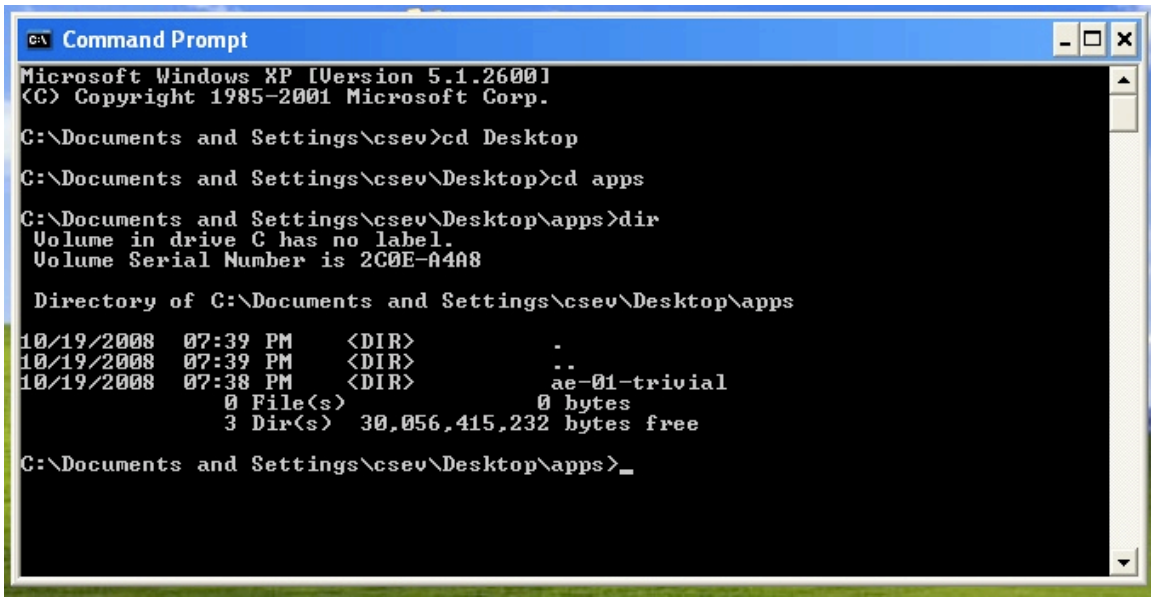
```
application: ae-01-trivial
version: 1
runtime: python
api_version: 1

handlers:
- url: /.*
  script: index.py
```

Note: Please do not copy and paste these lines into your text editor – you might end up with strange characters – simply type them into your editor.
Then create a file in the **ae-01-trivial** folder called **index.py** with a single line in it:

```
print "Hello there Chuck"
```

Then start the **Command Prompt** program by pessing the Windows icon in the lower left and typing "**Command**" in the "**Start Search**" box – when it highlights to the **Command Prompt** – press **Enter** to start the **Command Prompt**.

Then use the **cd** command to navigate into the **apps** directory.



Then start the Google Application Engine Web Server and run your application using the command:

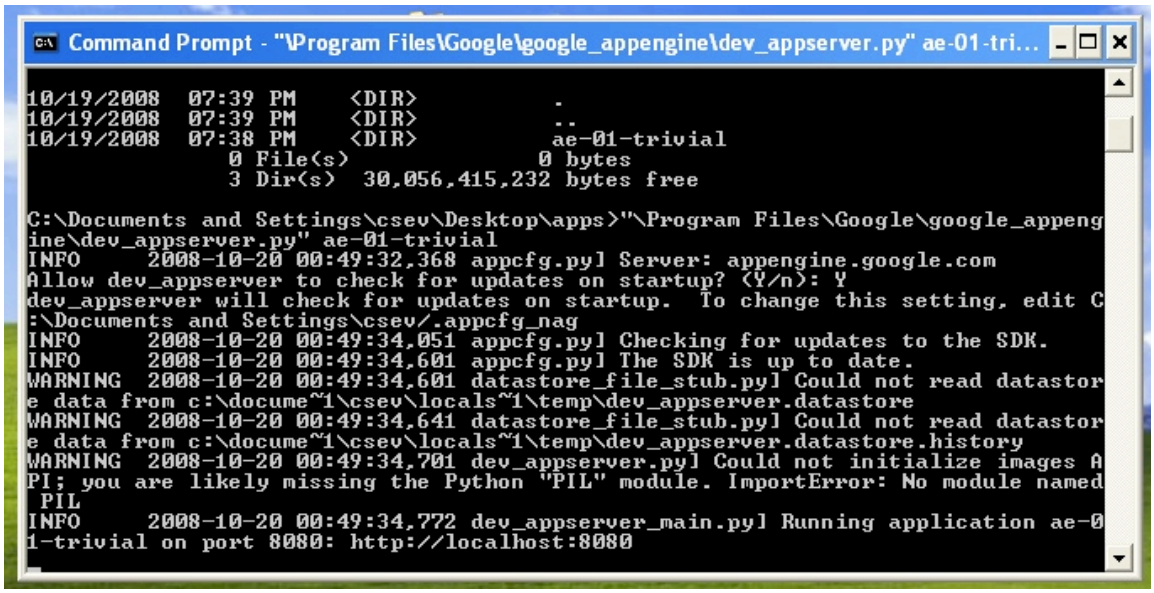
```
\Program Files\Google\google_appengine\dev_appserver.py ae-01-trivial
```

Hint: You can save some typing by using the following sequence – pressing **TAB** as indicated –they are just shown to make it easier.

\Pro**TAB**\Go**TAB**\go**TAB**\de**TAB**

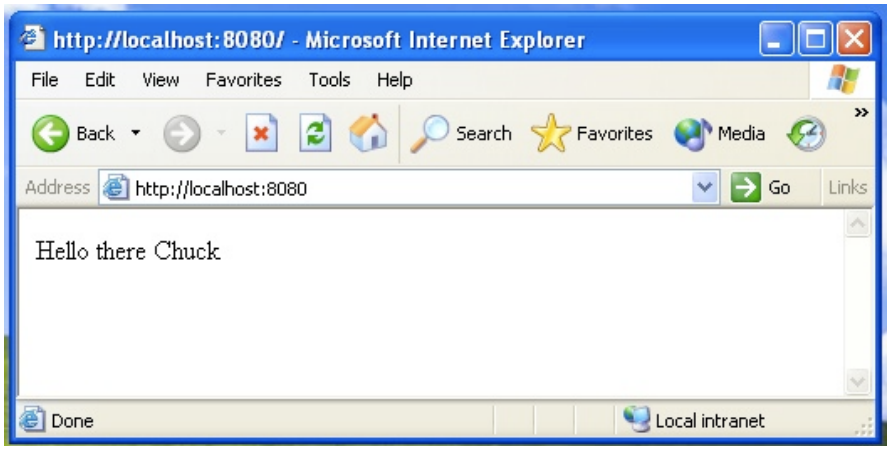
When you type part of a file name or folder name and press TAB, Windows completes the name for you to save you some tying. While this seems complex – it will soon become second nature.

Once the Application Engine starts with your application, you will be asked if you want the AppEngine to check for updates (Y is OK here) and then after a few messages – the server will start up as shown:



The last line tells you which port your application is running on and what URL you should use to access your application – in this case our application is at **http://localhost:8080**

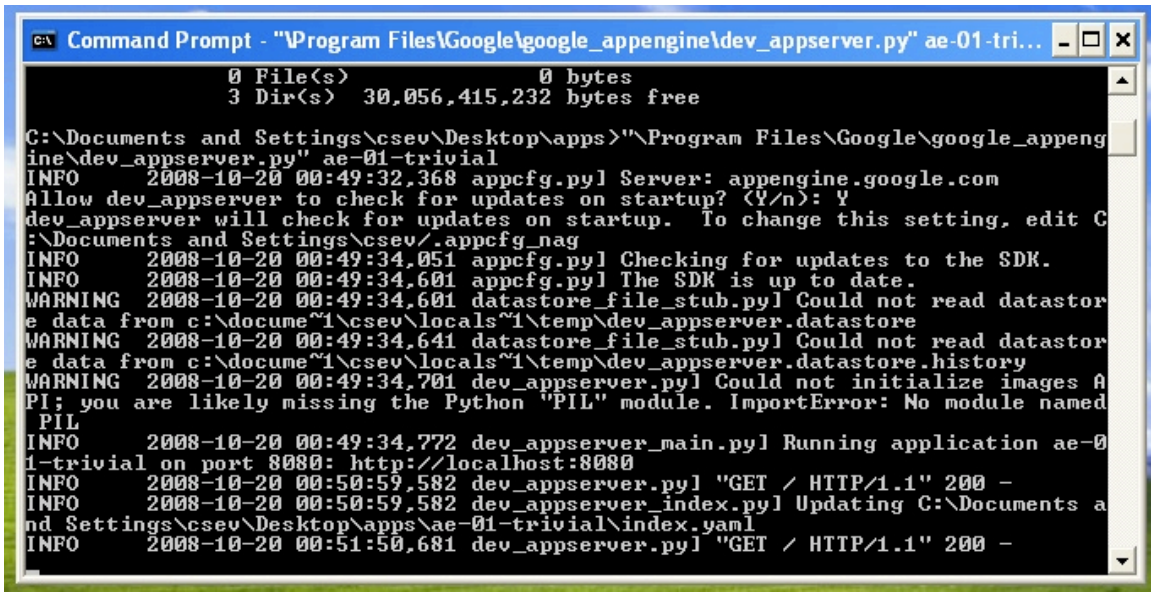
Paste **http://localhost:8080** into your browser and you should see your application as follows:



Just for fun, edit the **index.py** to change the name "Chuck" to your own name and press Refresh in the browser to verify your updates.

Watching the Log

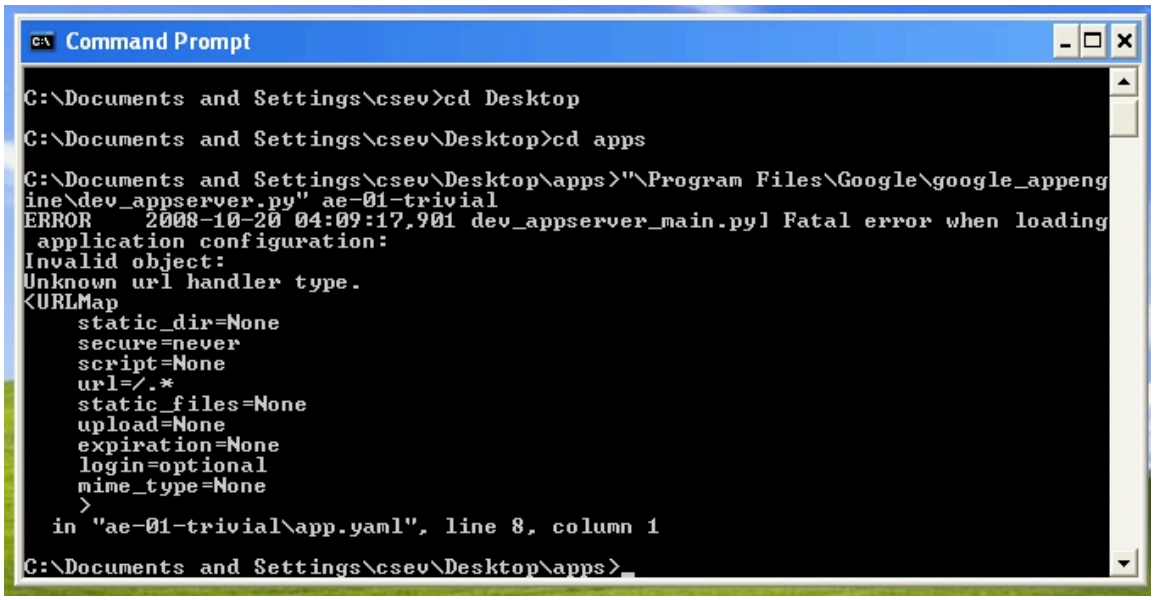
You can watch the internal log of the actions that the web server is performing in the same window as you started the application server:



Each time you press **Refresh** – you can see it retrieving the output with a **GET** request.

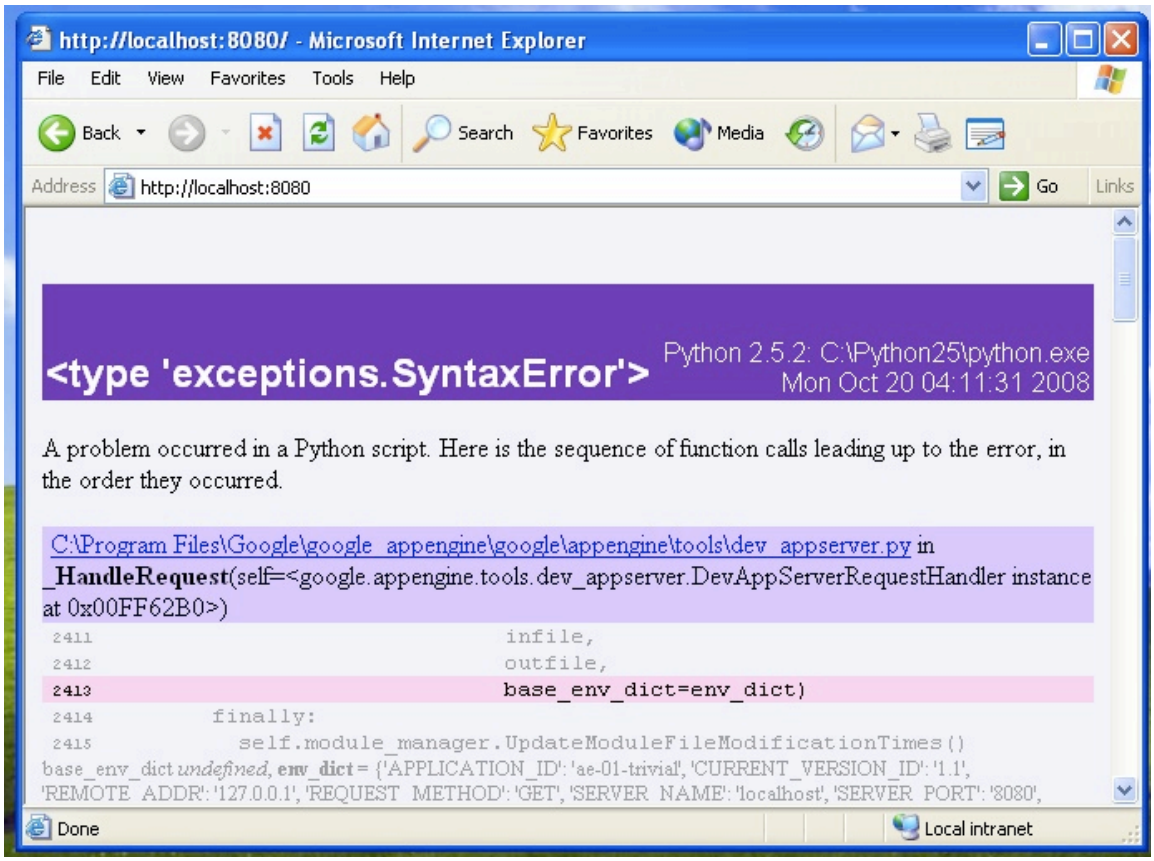
Dealing With Errors

With two files to edit, there are two general categories of errors that you may encounter. If you make a mistake on the **app.yaml** file, the AppEngine will not start and you will see an error like the following in the command line:

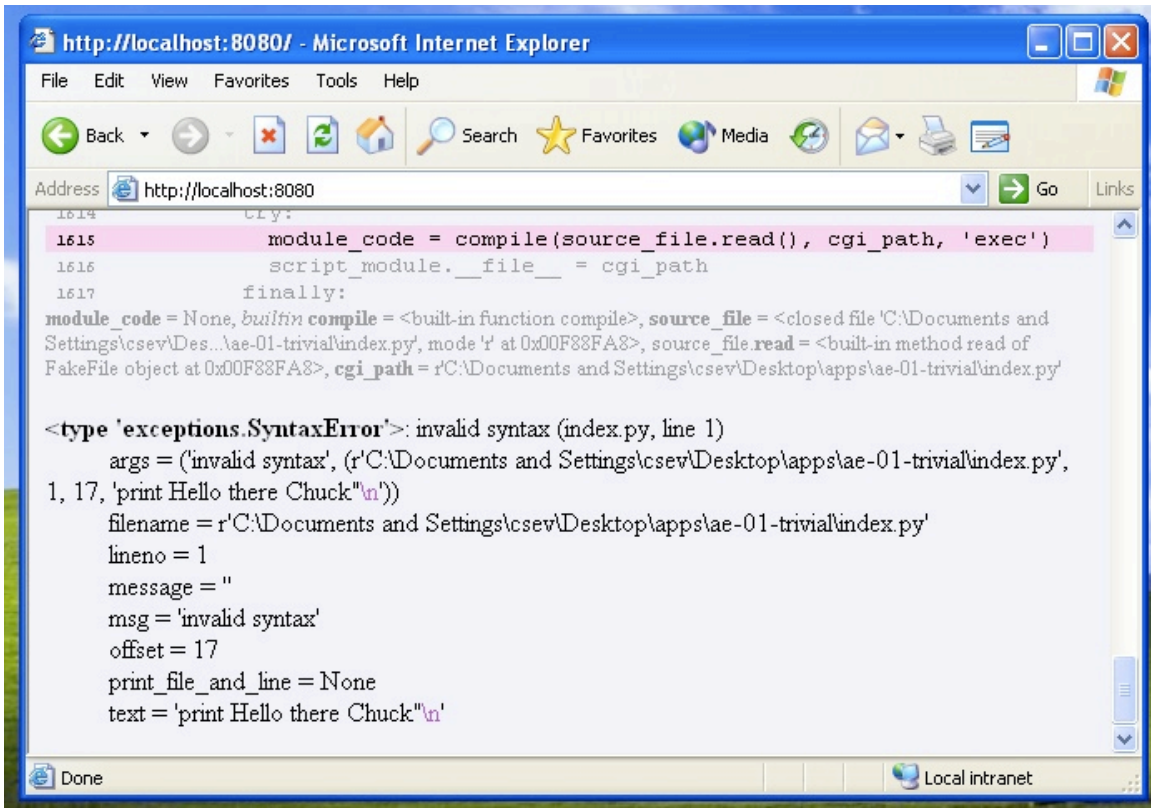


In this instance – the mistake is mis-indenting the last line in the **app.yaml** (line 8).

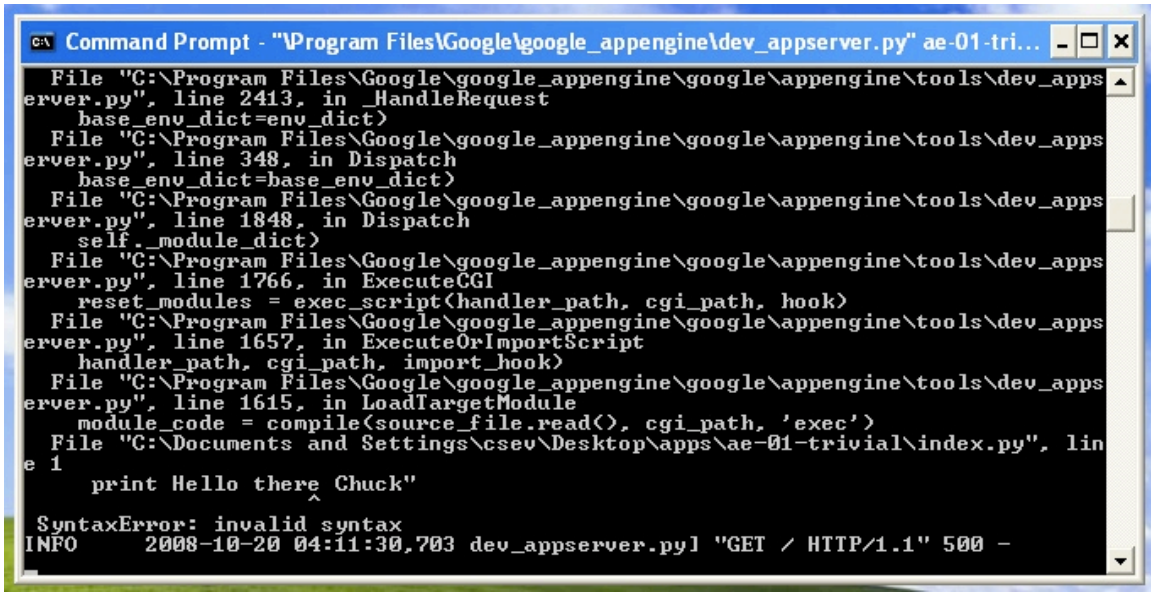
If you make a syntax error in the **index.py** file, the error will appear in your browser. The error looks terrible and looks like everything went wrong:



Do not be alarmed – ignore most of the output and scroll to the very bottom of the error output. The error you need to see is likely to be the very last line of the output – in this case I made a Python syntax error on line one of our one-line application.



If you are watching the window where you started the application server, the log will also show the error as follows:



Again, there is a lot of seemingly important output. Most of it can be ignored – it is "trace back" telling exactly where within the Google App Engine the error was detected. You can ignore most of is and find the mistake you made at the very end.

See also: http://en.wikipedia.org/wiki/Stack_trace

When you make a mistake in the **app.yaml** file – you must the fix the mistake and attempt to start the application again.

If you make a mistake in a file like i**ndex.py**, you can simply fix the file and press refresh in your browser – there is no need to restart the server.

Shutting Down the Server

To shut down the server, simply go into the window where you started the server and close the window to abort the server. When the server is shutdown, you will notice that navigating to **http://localhost:8080** will fail because there is no software running on and listening to port 8080.

Comments and questions to csev@umich.edu www.dr-chuck.com