Rapid Application Development

GAE (A Practical Introduction)

Coventry University

Outcomes

Installing and Configuring the GAE Launcher

Create a new project using the Launcher

Understand the purpose of the GAE files

Managing projects from the Launcher

Deploy a GAE Application to the cloud

Delete a deployed application

What is Google App Engine?

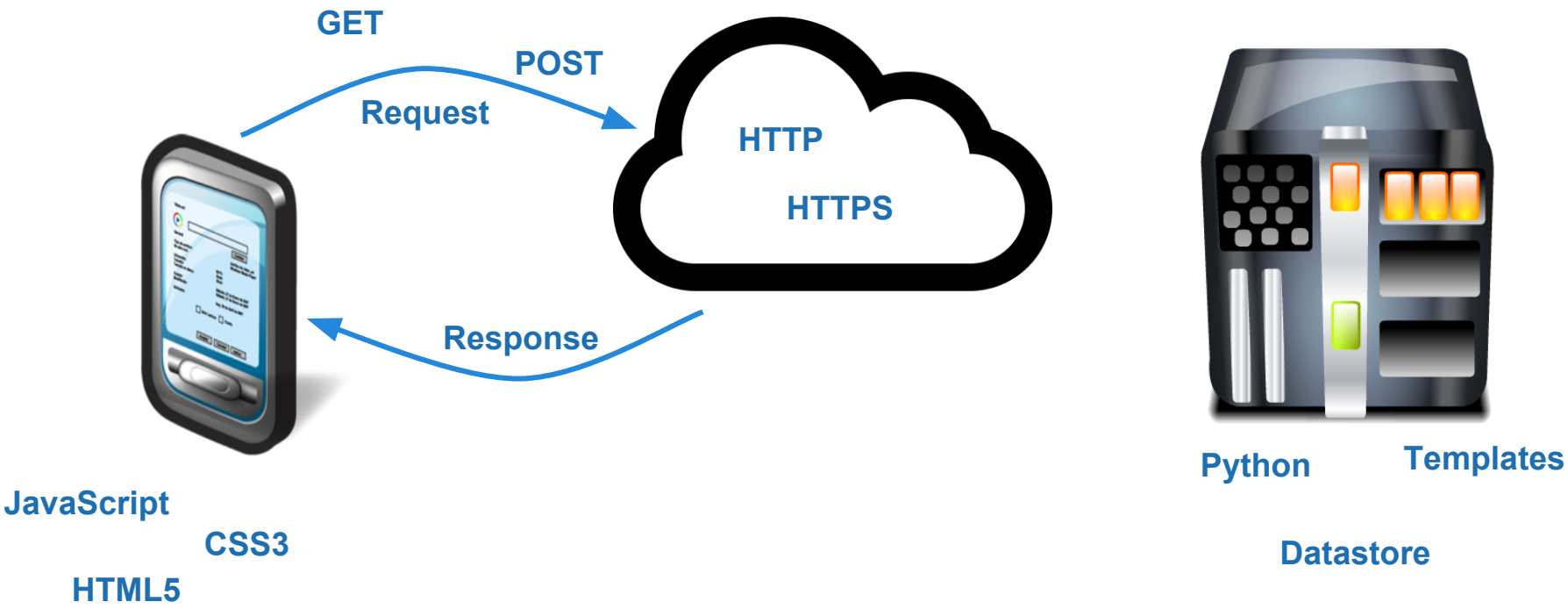
App Engine is a platform as a service that uses familiar technologies to build and host applications on the same infrastructure used at Google

https://cloud.google.com/products/app-engine/

Getting Started

A Quick Video Introduction (2 min)

How Google App Engine Works



Why Use Google App Engine?

Exposure to leading new technologies and concepts: cloud computing, big data, NoSQL, scalability

Easy to get started

Using some already familiar technologies

Programming in the Cloud

Programmers operate in a controlled environment

Programs do their programming thing - code + data

A complex software framework manages getting the right code and data to/from the right servers.

Software developers are unaware of geography

Source: http://www-personal.umich.edu/~csev/books/gae/lectures/01-AppEngine-Intro.pdf

Web Application Hosting Service

Real-time dynamic applications

Applications scale automatically

Free to start; pay only for resources that are used e.g. CPU usage, storage per month, incoming and outgoing bandwidth

Uses quotas

Services

Users supports user login via existing Google accounts

DataStore allows for data persistence using noSQL

Blobstore serves large data objects such as videos that exceed the maximum datastore size

URL Fetch enables the application to communicate with RESTful web services

Mail allows the sending of email messages via Google Mail

XMPP Chat enables interaction with any messaging service that uses XMPP such as Google Talk

Serving Static Files

Static files: CSS, HTML, JavaScript, images

App Engine provides a separate set of servers to deliver static files

Configurations aspects: content types, instructions for caching, URLs

Storing Data

App Engine Datastore

A schemaless object datastore with automatic caching, a query engine, and transaction support

Google Cloud SQL

A relational SQL database based on MySQL

Google Cloud Storage

A storage service for objects and files up to terabytes in size, and accessible using the Google Cloud Storage client library.

Google Account Integration

App Engine integrated with Google Accounts and OpenID

Users that already have Google accounts can sign in your app using their existing accounts

Full support for OAuth authentication to grant a third party limited permission to access your application without sharing user credentials

HTTP (Revision)

Hypertext Transfer Protocol

The main protocol of the web.

Used by the browsers to communicate with web servers.

A request from your browser for the URL: http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

GET /wiki/Hypertext_Transfer_Protocol HTTP/1.1 (Method, Path, Version)

The two main methods of HTTP are GET (fetch a resource) and POST (add a new resource)

Google App Engine SDK

Download and Install

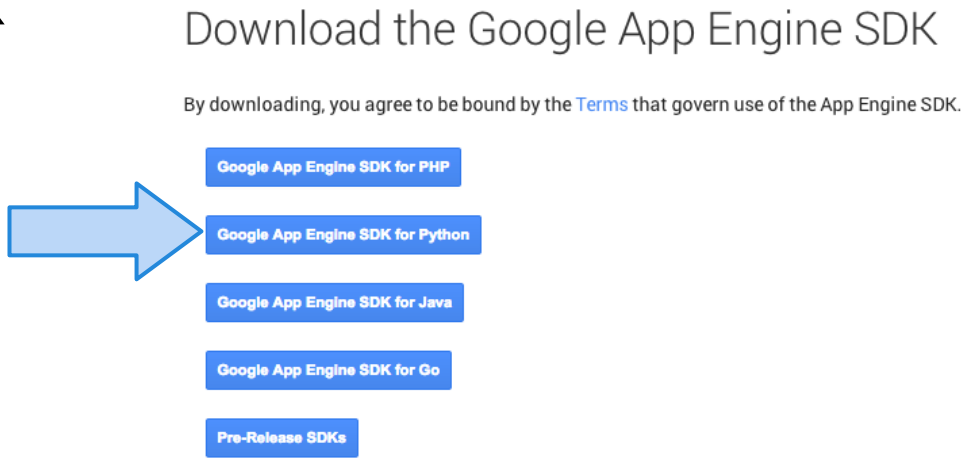
https://developers.google.com/appengine/downloads

Make sure you have python 2.7 installed

Choose the appropriate SDK

Includes the

App Engine Launcher



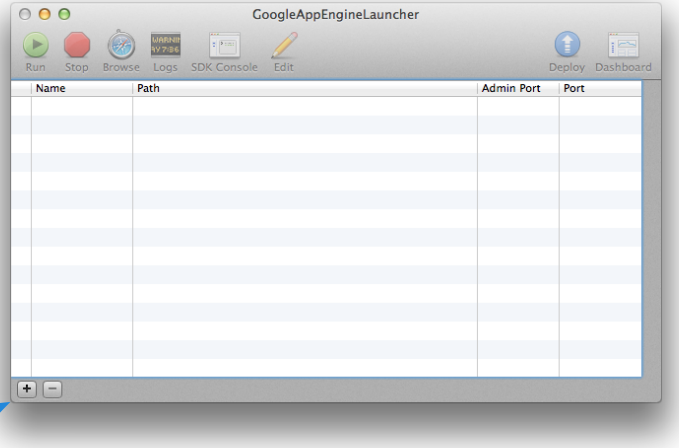
App Engine Launcher

Two options for compiling and installing apps

1. Command line
2. App Engine Launcher

Launcher is more convenient

New app button



Downloading an Editor

You will need to install a compatible editor

Recommend using Brackets

http://brackets.io/

Make sure this is installed before continuing

Brackets Editor

Brackets – The Free, Open Source Code Editor for the Web

brackets.io

Brackets Contribute API Docs Blog Support

Code the Web

Brackets is an open source code editor for web designers and front-end developers.

Download Brackets
Release 0.42 (OSX)

Other Downloads Requirements Installation

Live HTML Development
As you code, HTML changes are instantly pushed to browser without having to save or reload the page.
See it in action.

JS Debugging with Theseus
Theseus makes inspecting variables and control flow easy, even in asynchronous code.
Learn more.

Linux: New & Improved
Over the last few releases we've made progress on Linux (Debian/Ubuntu) support for Brackets.
Check it out.

Effective Development
Imagine the ability to fluently edit external styles and code without even leaving your document!
See how it looks.

Configuring the GAE Editor

Convenient to configure Brackets as your default editor

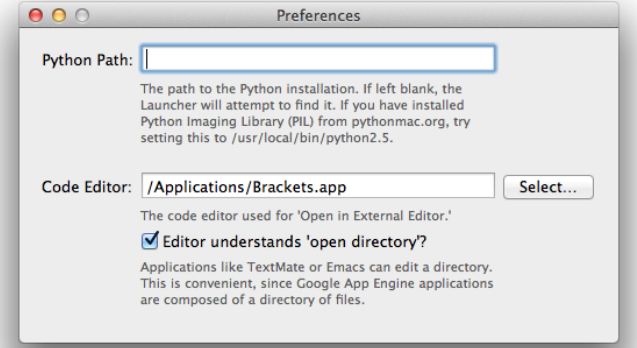
Open the App Engine Launcher preferences

Browse to your Brackets editor

You should not have to specify a Python path

Brackets can open a directory

so check that option.



Creating a New Project

To create a new project click on the + button

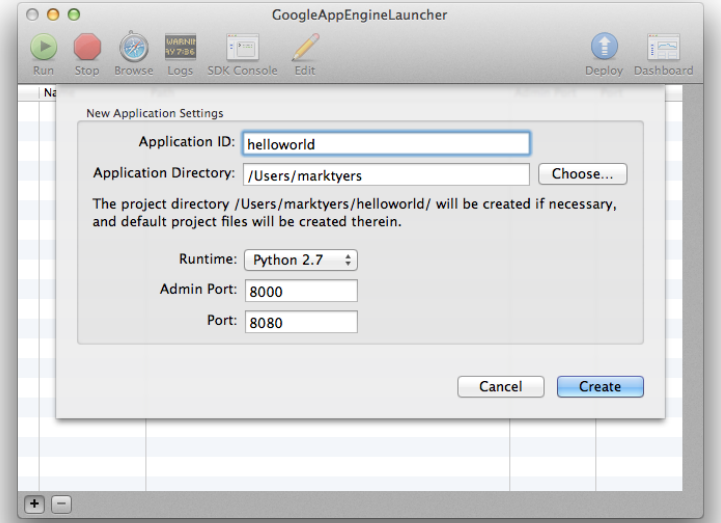
You will need to give your project a name

Choose an application directory

on your local hard drive

For now accept the default values

for everything else

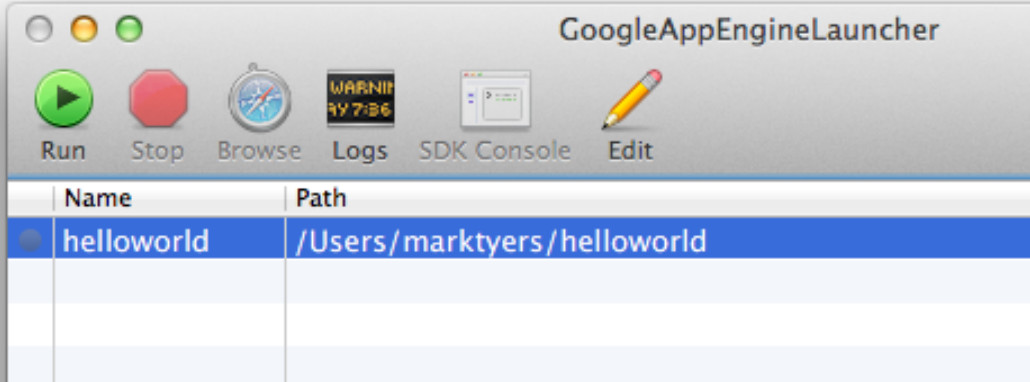


Running a Project

There are two steps:

Run – this starts your development server and runs the application

Browse – launches the web browser and points it to your running application

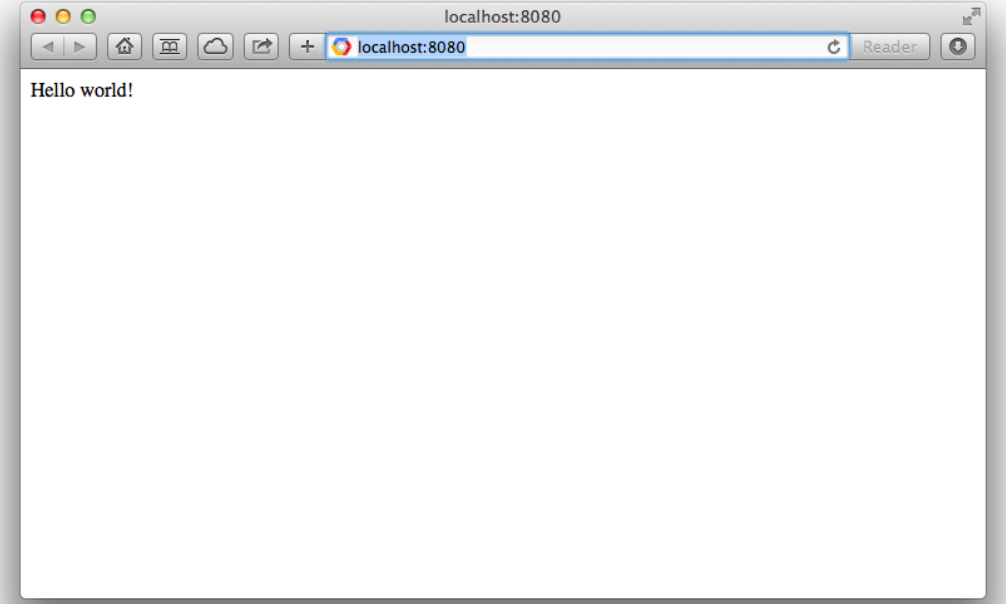


Viewing Your Application

The Development Server creates its own web server

Defaults to port 8080

Access using browser



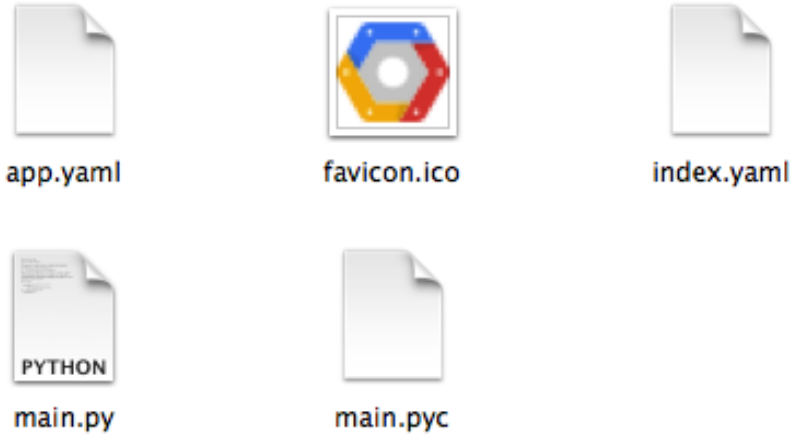
Project Files

Datastore index (index.yaml) do not edit this!

Python script (main.py) compiled as main.pyc

Configuration file (app.yaml)

Application icon (favicon.ico)



Key files to edit:

- app.yaml
- main.py

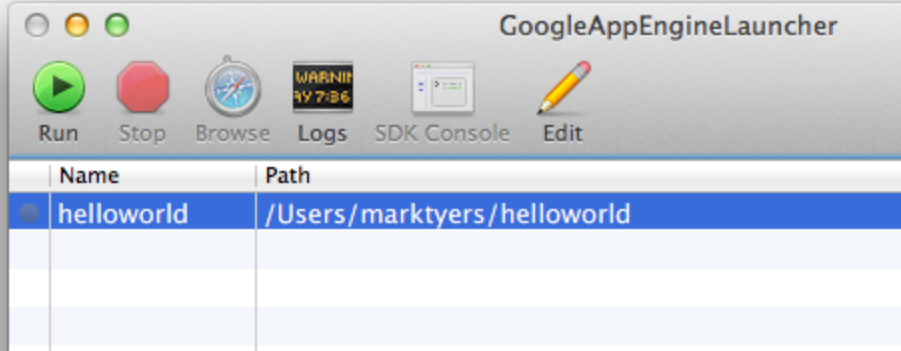
Editing the Files

Files can be edited directly in the folder

Brackets editor can open a folder

Click on the edit button to open Brackets with all project files

No need to start and stop the web server when editing!




Brackets
helloworld
app.yaml
favicon.ico
index.yaml
main.py
CODE
THE
WEB

Configuration File

https://developers.google.com/appengine/docs/python/config/appconfig

```yaml
application: helloworld
version: 1
runtime: python27
api_version: 1
threadsafe: yes

handlers:
- url: /favicon\.ico
  static_files: favicon.ico
  upload: favicon\.ico

- url: .*
  script: main.app

libraries:
- name: webapp2
  version: "2.5.2"
```

Application Settings

Information about your application

Handlers

A list of URL patterns and how they should be handled. Evaluated top to bottom until a match is found.

Libraries

Third party modules needed by the application

Python Script

This file contains the application logic

This code will get run when a request is received

Creates an instance of the appropriate class and builds an Application Object

```python
#!/usr/bin/env python

import webapp2

class MainHandler(webapp2.RequestHandler):
    def get(self):
        self.response.write('Hello world!')

app = webapp2.WSGIApplication([
    ('/', MainHandler)
], debug=True)
```

Import

Importing libraries listed in the configuration file.

Class Definition

Contains the code to be run when a request is received.

Creating Application

Creates the application object to handle the requests

Handling a Request

1. Client makes a web request
2. URL determines the application to run
3. Selects a server to handle the request
4. Calls the application with the HTTP content request
5. Receives the response data from the application
6. Returns response to the client

Handling Multiple Users

App Engine creates and destroys instances as needed to accommodate an app traffic

Multithreading feature - single instance of an app can handle multiple requests concurrently achieving better utilization of resources

The Sandbox

To reduce security risks, applications run in ‘sandboxes’ which prevents one application interacting with others

Imposes a number of restrictions:

- Applications can’t write to the filesystem (must use a datastore)
- Applications can only communicate using ports 80 and 443
- Applications can’t take more than a few seconds to respond to requests (can’t tie up system resources)
- Applications can’t make system calls

The WebApp2 Framework

Someone has already written the common code that knows all the details of HTTP (HyperText Transport Protocol)

We just import it and then use it.

It conforms to a common standard for Python web applications frameworks Web Server Gateway Interface (WSGI)

No need to know much about WSGI

WebApp2 In Action

We defines one request handler, MainPage, mapped to the root URL (/).

When webapp2 receives an HTTP GET request to the URL /, it instantiates the MainPage class and calls the instance's get method.

The method returns self.response which contains the information to be displayed. We write our output to this

The application itself is represented by a webapp2. WSGIApplication instance.

Debugging

We pass the parameter debug=true to the constructor of the WSGIApplication instance

It tells webapp2 to print stack traces to the browser output if a handler encounters an error or raises an uncaught exception.

When we deploy the application to the public server this should be set to false

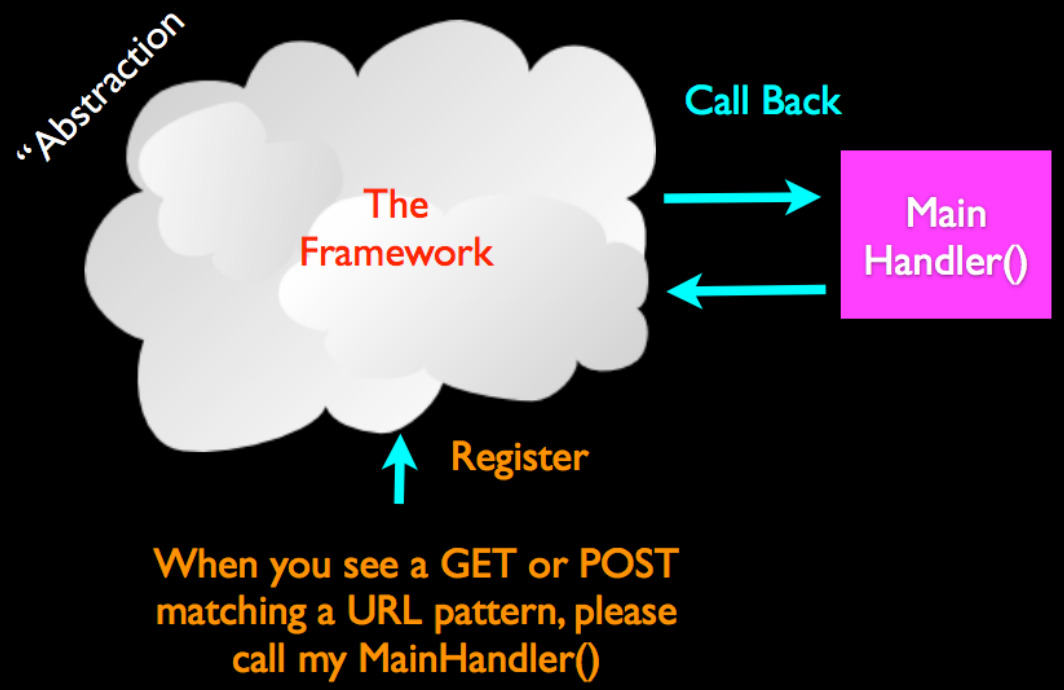
The Main Handler

When we are dealing with a framework - at times the framework needs to ask us a question or involve us with some bit of processing.

Often this is called “event processing” or “event Handling”. Another word for this is “callbacks”

We register interest in certain actions and then when those actions happen - we get called.

The Framework and The Handler



The Callback Pattern

The callback pattern is very common in Object Oriented programming.

The basic idea is that we hand over the main responsibility for handling something to a framework (i.e. a large amount of code we did not write) – and then let the framework call one of our objects back – at some important moment when it wants us to participate in the handling of the request.

This pattern is used in many situations ranging from graphical user interfaces to message/event handling.

How the Callback Pattern Works

We initially communicate to the framework those “happenings” or event that we are interested in

and give the framework a bit of our code to call to “handle” those events.

That is why we use the convention of naming these bits of code with “Handler” in their names

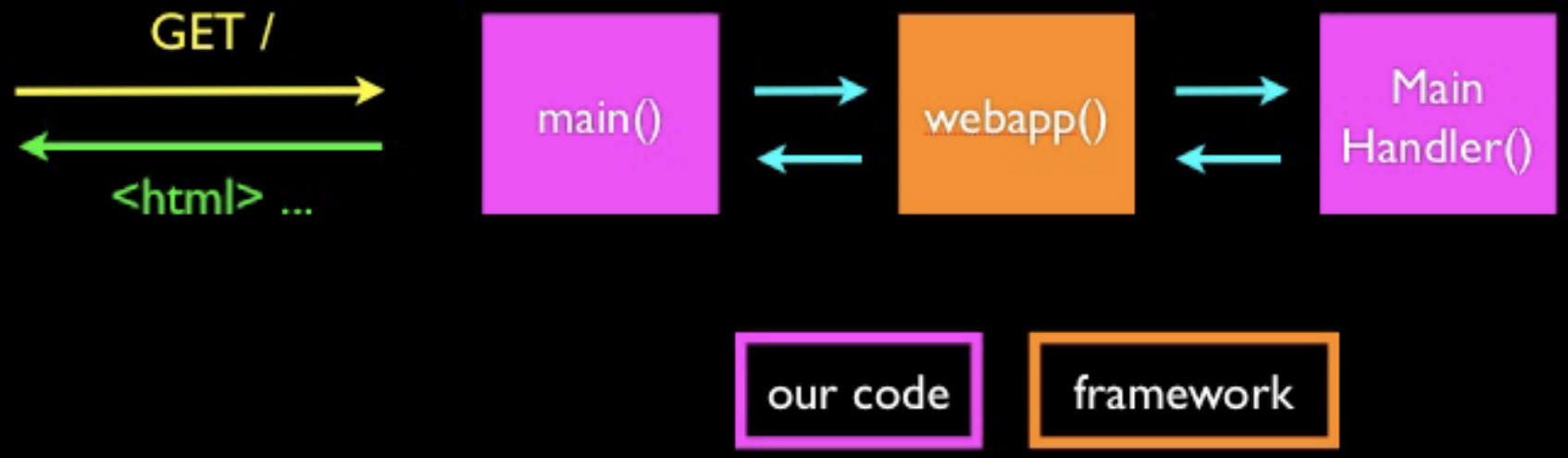
they are designed to “Handle” something.

The Callback Pattern in Action

The incoming HTTP request arrives to our main program. Instead of handling the request directly, we simply set up the framework and tell it under what conditions (urls that match /.*) and where (MainHandler) to call us back when it needs some “assistance” from us.

Then the framework starts up and looks at the HTTP request, figuring out which kind of request it is – parsing all of the data, converting file input if necessary – and then calls out MainHandler – using either the get() or post() method as appropriate

The Callback Pattern



Working with the Launcher

There are a couple of features of the App Engine Launcher that are deserving of further study:

Removing an application

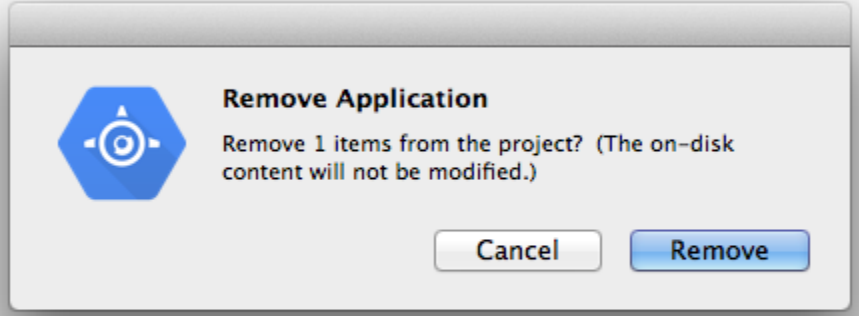
Adding an existing application

Removing an Application

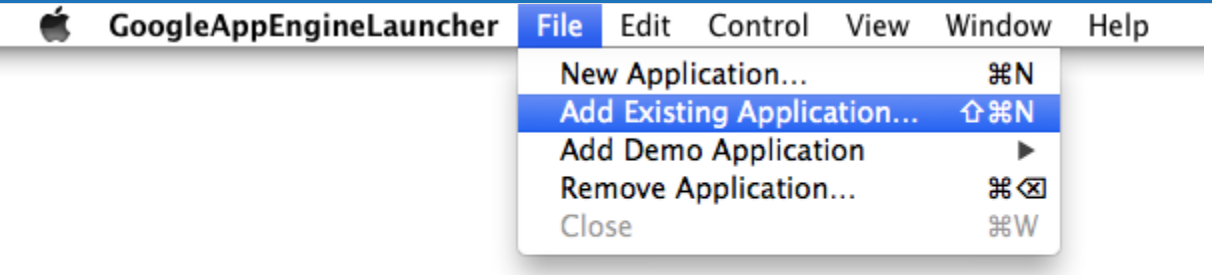
We should only have applications in the launcher we are actively working on

Clicking the (-) button removes the application from the launcher

But does not delete the files



Adding an Existing Application



If you download a sample application from the web

Or want to work on an application you have previously removed

Use the menu to add it to you launcher

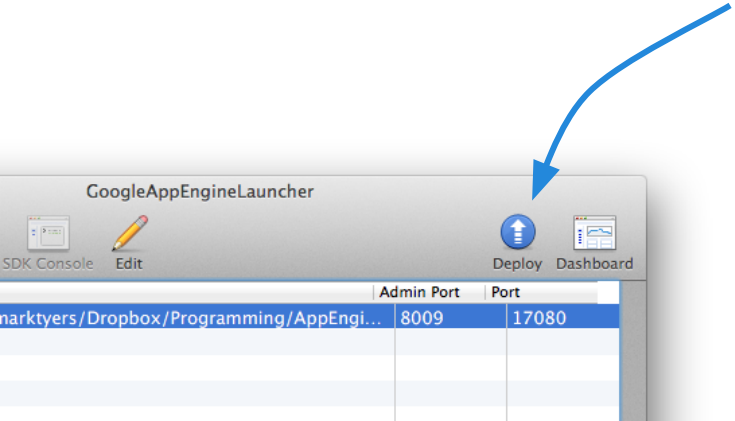
This will not move the project folder

Deploying Applications

Whilst it is useful to be able to develop and debug your application on a local computer

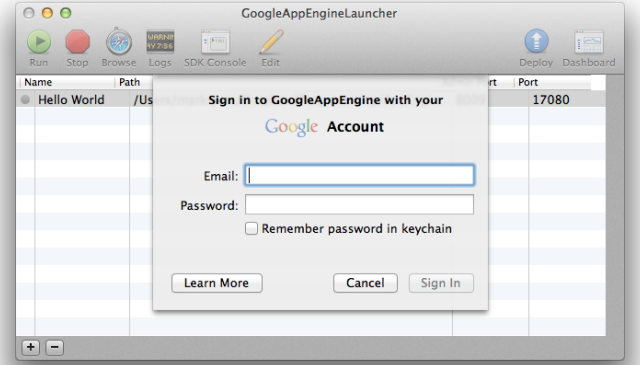
Eventually you will need to deploy it to the Google Cloud

This is handled using the deploy button on the launcher

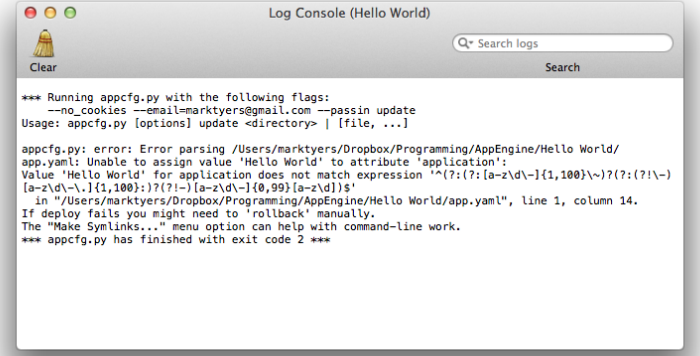


Logging in and Deploying

We need to log in using a Google account



But we get an error...



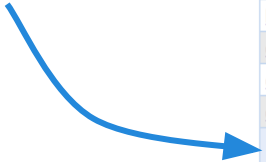
The GAE Console

A browser-based tool to create, configure and monitor deployed applications

appengine.google.com

Dashboard showing all deployed applications

Create new



My Applications

‹ Prev 20 **1-4 of 4** Next 20 ›

Application	Title	Storage Scheme	Status
bookshopjs	bookshop	High Replication	**Disabled by developer** [?]
covuni69	Test App	High Replication	**Disabled by developer** [?]
cu-1234567	CU Lab Exercises	High Replication	Running
cubookshop	Coventry University Bookshop	High Replication	**Disabled by developer** [?]

Create Application

You have 24 applications remaining.

‹ Prev 20 **1-4 of 4** Next 20 ›

Creating a New Application

Create an Application

You have 25 applications remaining.

Application Identifier:

marktyers .appspot.com Check Availability **Yes, "marktyers" is available!**

All Google account names and certain offensive or trademarked names may not be used as Application Identifiers.
You can map this application to your own domain later. Learn more

Application Title:

Hello World

Displayed when users access your application.

Authentication Options (Advanced): Learn more

Google App Engine provides an API for authenticating your users, including **Google Accounts**, **Google Apps** , and **OpenID**. If you choose to use this feature for some parts of your site, you'll need to specify now what type of users can sign in to your application:

- (•) **Open to all Google Accounts users (default)**
  If your application uses authentication, anyone with a valid Google Account may sign in.
- ( ) **Restricted to the following Google Apps domain:**
  e.g. foo.com
  If your application uses authentication, **only members of this Google Apps domain may sign in**. If your organization uses Google Apps, use this option to create an application (e.g. an HR tracking tool) that is only accessible to accounts on your Google Apps domain. This option cannot be changed once it has been set.
- ( ) **(Experimental) Open to all users with an OpenID Provider**
  If your application uses authentication, anyone who has an account with an OpenID Provider may sign in.

Create Application Cancel

Confirmation Screen

Application Registered Successfully

The application will use **marktyers** as an identifier. This identifier belongs in your application's configuration as well. Note that this identifier cannot be changed. Learn more

The application uses the **High Replication** storage scheme. Learn more

If you use Google authentication for your application, **Hello World** will be displayed on Sign In pages when users access your application.

Choose an option below:

- View the dashboard for Hello World.
- Use appcfg to upload and deploy your application code.
- Add administrators to collaborate on this application.

Modifying the Config File

Remember the error when we tried deploying previously?

The Application setting in the config file must match the Application Identifier we assigned to our new application online

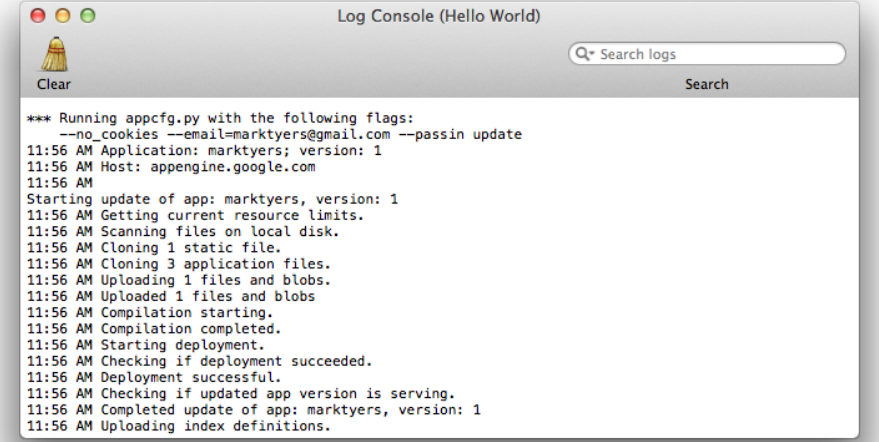
In this case the name needs to be marktyers

```yaml
application: marktyers
version: 1
runtime: python27
api_version: 1
threadsafe: yes

handlers:
- url: /favicon\.ico
  static_files: favicon.ico
  upload: favicon\.ico

- url: .*
  script: main.app

libraries:
- name: webapp2
  version: "2.5.2"
```

Deploying the Application (Success)

Now we can deploy to the cloud

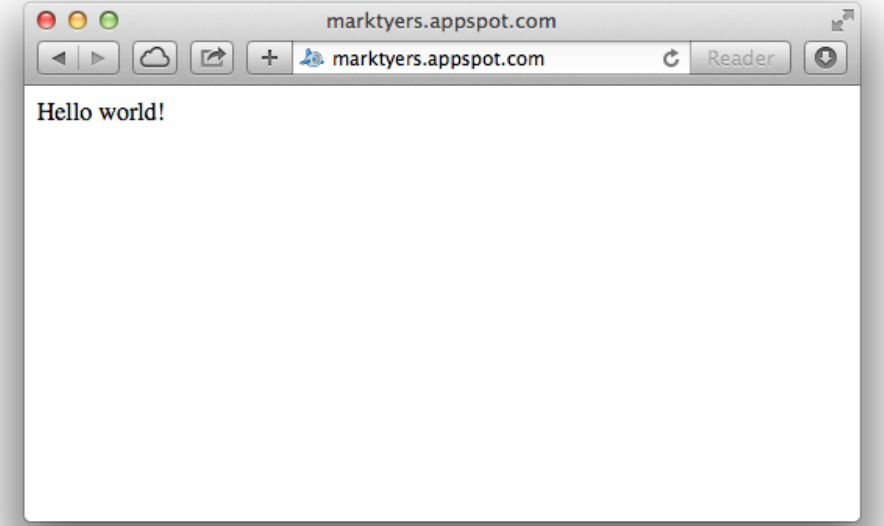
Open the Log Console to monitor the progress



Testing the Application

Navigate to the appropriate URL to see the online application

http://marktyers.appspot.com



Deleting an Application

This is a three-step process to prevent accidental deletion

- Disable application (all data is retained)
- Request deletion
- Application will be deleted 72 hours later

Disabling an Application

Administration settings on the application dashboard

Select Application Settings

Button halfway down the page

Administration

- Application Settings
- Permissions
- Blacklist
- Admin Logs

Disable or Delete Application

Disable or delete this application. Learn more

Disable Application...

Disable Application

This application is enabled. Do you want to disable it? Serving requests will stop immediately, but you will not lose any data or state. You can re-enable it at any time. Learn more

Disable Application Now | Cancel

Delete Application

Your changes have been saved.

This application is disabled. Serving requests has been stopped, but all your data and state have been saved. Do you want to re-enable it? Learn more

Re-enable Application Now | Cancel

Do you want to request permanent deletion of your application? If you click the button below, you will have 72 hours to change your mind. After that time, the application's data and state will be inaccessible, and you cannot get it back. However, the application id (s~cu-1234567) will remain reserved approximately forever. Learn more

Request Permanent Deletion

Delete Application

Your changes have been saved.

This application is disabled. Serving requests has been stopped, but all your data and state have been saved. Do you want to re-enable it? This will also revert the deletion request. Learn more

Re-enable Application Now | Cancel

Deletion of this application has been requested by marktyers@gmail.com. **This application will be deleted after September 10, 2014 03:49 AM (US/Pacific time).** Until then you can still revert the deletion request. Do you want to revert the deletion request? Learn more

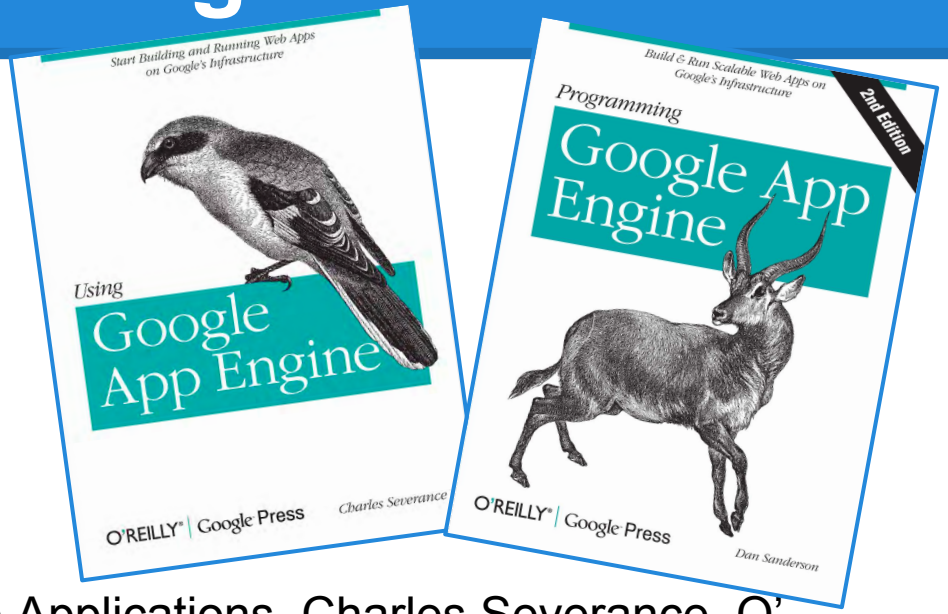
Revert Deletion Request

Recap

You should now be able to do the following:

- Installing and Configuring the GAE Launcher
- Create a new project using the Launcher
- Understand the purpose of the GAE files
- Managing projects from the Launcher
- Deploy a GAE Application to the cloud
- Delete a deployed application

Recommended Reading



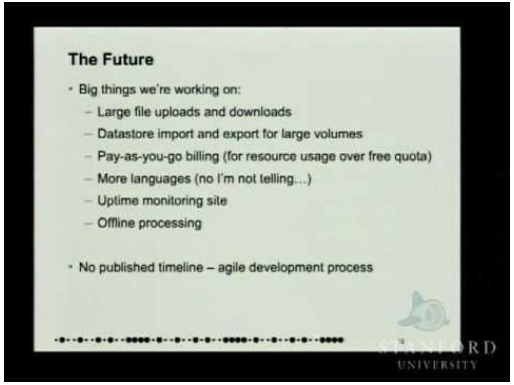
Programming Google App Engine With Python, Dan Sanderson, O'Reilly Media 2015

Using Google App Engine Building Web Applications, Charles Severance, O' Reilly Media 2009

Programming Google App Engine, 2nd Edition, Dan Sanderson, O'Reilly Media 2012

Useful Videos

Intro to the Google Cloud Platform



Google App Engine

App Engine Overview

More Video Resources

App Engine Video Resources

Many Googlers have given talks at Google I/O and elsewhere about App Engine systems and best practices. Below is some of the video content available on the web.

Contents

https://developers.google.com/appengine/docs/videoresources

References

Chapter 1, Using Google App Engine Building Web Applications, Charles Severance, O'Reilly Media 2009, http://www.appenginelearn.com/

Chapter 1, Programming Google App Engine, 2nd Edition, Build & Run Scalable Web Applications on Google's Infrastructure, Dan Sanderson, O'Reilly Media 2012, http://ae-book.appspot.com/

https://developers.google.com/appengine/docs/python/

Polling: http://en.wikipedia.org/wiki/Polling_(computer_science)

http://www-personal.umich.edu/~csev/books/gae/lectures/01-AppEngine-Intro.pdf