雲端運算基礎課程(三)
Google App Engine 體驗課程

2009.11.30

國網中心格網技術組
專案助理研究員 鄭宗碩, Zong-shuo Jheng
c00aso00@nchc.org.tw

Outline

- 雲端上的引擎 － Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 － Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 － Template
- 資料黑盒子 － DataStore

Outline

- 雲端上的引擎 － Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 － Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 － Template
- 資料黑盒子 － DataStore

雲端運算是什麼?! 雲端又在哪?!

- 雲 = 網路

隨時(Anytime)

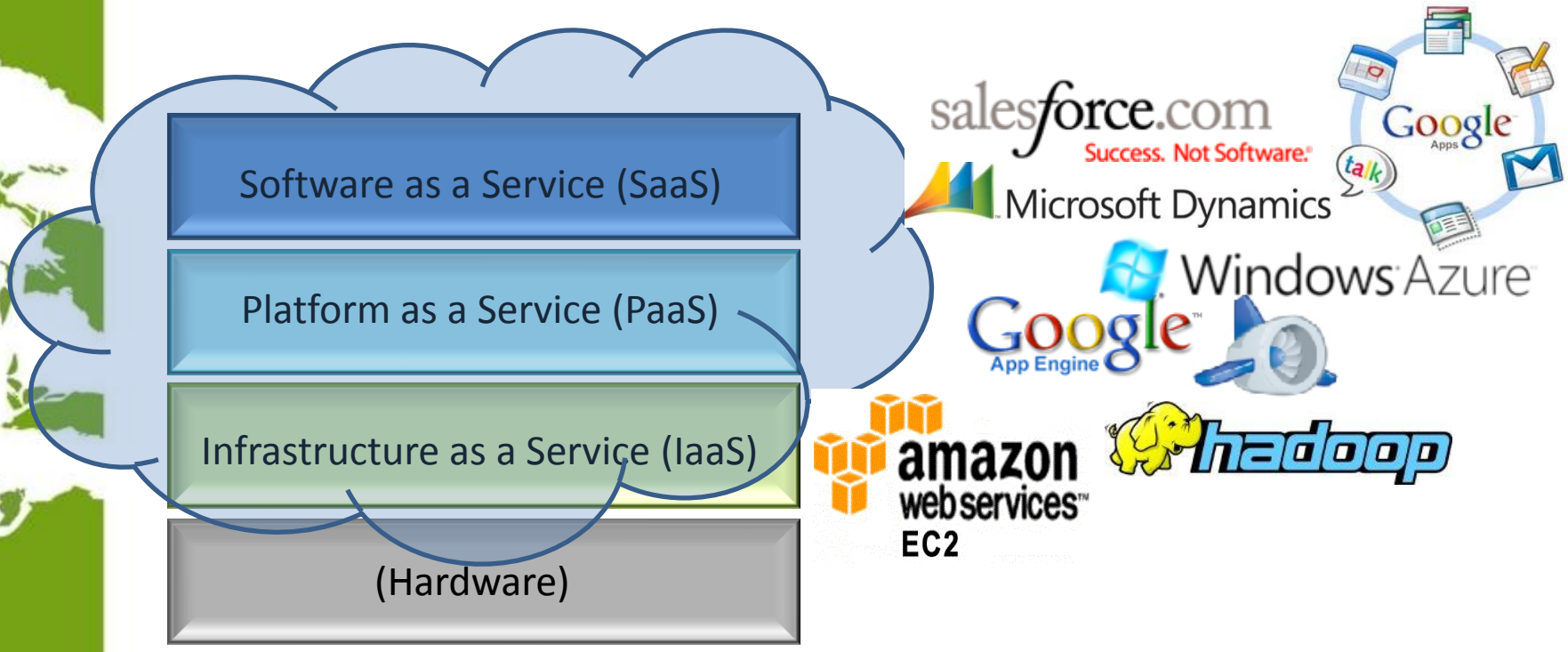
隨地(Anywhere)

使用任何裝置(With any devices)

存取各種服務(Accessing services)

穿透雲層

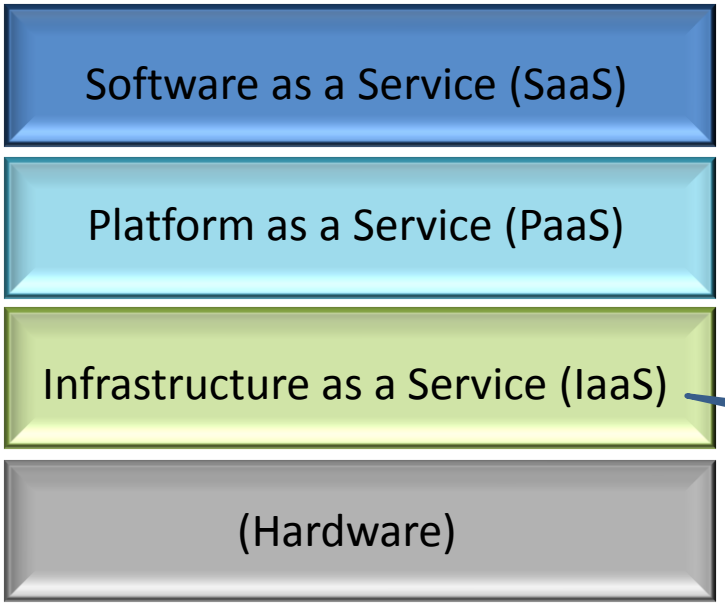
- 雲端的服務分層



圖片參考來源: http://edgewatertech.wordpress.com/

我在哪?!

- 立足於PaaS層, 開發SaaS提供服務



工商服務時間

雲端運算課程(一) Hadoop實務課程
王耀聰先生、 陳威宇先生
雲端運算課程(二) Xen 虛擬化實務課程
郭文傑先生、涂哲源先生

Why Google App Engine?

- 強健的Google雲端平台
  - Google account, BigTable, Google File System
- 有限度的資源免費使用
  - 10 個應用程式帳號, 500MB儲存空間, 每個月5百萬次的網頁瀏覽, 每次2 0億(2 billion) CPU clock cycles運算, 每日10G流量(up/down)
- 豐富的API支援及DataStore
  - Memcache, URL Fetch, Mail service, Image Processing, Google account

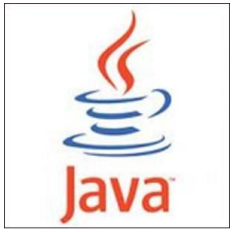
Outline

- 雲端上的引擎 – Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 – Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 – Template
- 資料黑盒子 – DataStore

成為新手駕駛之前

- Python runtime
  - http://www.python.org

- Java runtime
  - http://www.java.com/



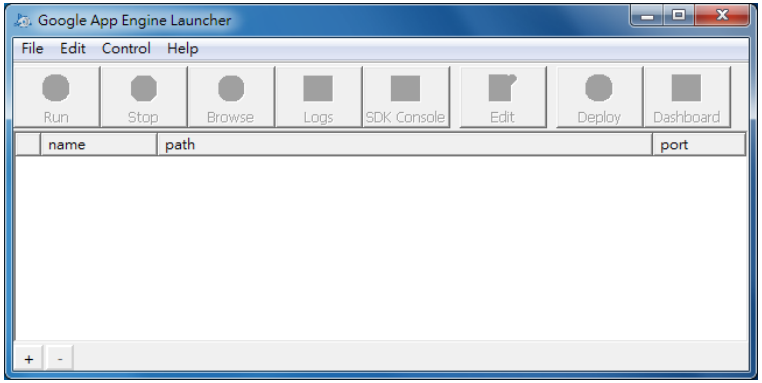
- Google App Engine SDK
  - http://code.google.com/appengine/

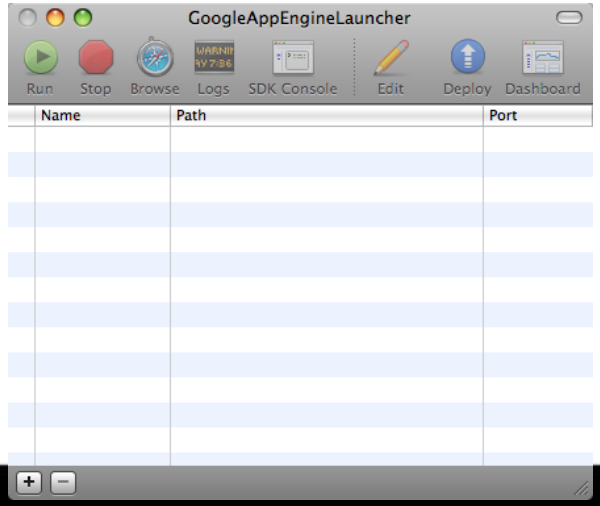
開發Google App Engine第一步

- 報考Google App Engine駕訓班
  - 申請Google App Engine專案帳號

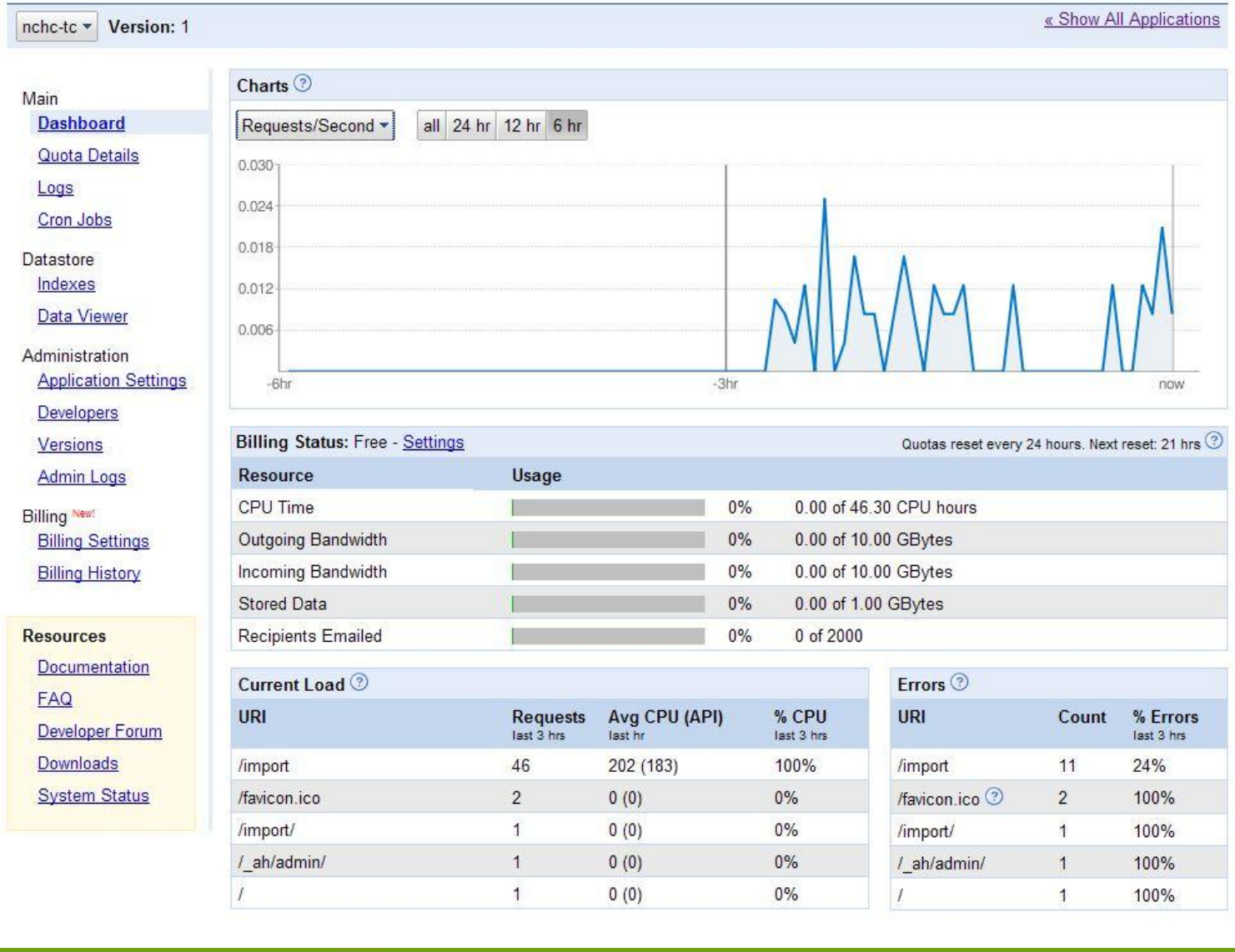
    http://appengine.google.com/

- 工欲善其事, 必先利其器





檢視儀錶板


nchc-tc
Version: 1
« Show All Applications
Main
Dashboard
Quota Details
Logs
Cron Jobs
Datastore
Indexes
Data Viewer
Administration
Application Settings
Developers
Versions
Admin Logs
Billing New!
Billing Settings
Billing History
Resources
Documentation
FAQ
Developer Forum
Downloads
System Status
Charts
Requests/Second
all 24 hr 12 hr 6 hr
0.030
0.024
0.018
0.012
0.006
-6hr
-3hr
now
Billing Status: Free - Settings
Quotas reset every 24 hours. Next reset: 21 hrs
Resource Usage
CPU Time 0% 0.00 of 46.30 CPU hours
Outgoing Bandwidth 0% 0.00 of 10.00 GBytes
Incoming Bandwidth 0% 0.00 of 10.00 GBytes
Stored Data 0% 0.00 of 1.00 GBytes
Recipients Emailed 0% 0 of 2000
Current Load
URI Requests last 3 hrs Avg CPU (API) last hr % CPU last 3 hrs
/import 46 202 (183) 100%
/favicon.ico 2 0 (0) 0%
/import/ 1 0 (0) 0%
/_ah/admin/ 1 0 (0) 0%
/ 1 0 (0) 0%
Errors
URI Count % Errors last 3 hrs
/import 11 24%
/favicon.ico 2 100%
/import/ 1 100%
/_ah/admin/ 1 100%
/ 1 100%

Outline

- 雲端上的引擎 – Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 – Python
- 在Google雲端上向世界說"哈囉"
- 引擎組成元件
- 華麗的引擎外殼 – Template
- 資料黑盒子 – DataStore

Outline

- 雲端上的引擎 – Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 – Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 – Template
- 資料黑盒子 – DataStore

Hello, World!

<<app.yaml>>

```
application: hello-gae
version: 1
runtime: python
api_version: 1

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

<< hello.py >>

```
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class MainPage(webapp.RequestHandler):
  def get(self):
    self.response.out.write('Hello Google App Engine!')

def main():
  application = webapp.WSGIApplication([('/', MainPage)],
                                       debug=True)
  run_wsgi_app(application)

if __name__ == "__main__":
  main()
```

把Hello送出去

- 本機端執行與測試

```
./dev_appserver.py -h
./dev_appserver.py You_Hello_World_Project_Name
```

- 發佈到Google雲端平台

```
./appcfg.py -h
./appcfg.py update You_Hello_World_Project_Name
```

練習：說聲“哈囉”吧!

Outline

- 雲端上的引擎 － Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇－ Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 － Template
- 資料黑盒子 － DataStore

專案主要元素

- Python script files (or Java)
- Configure files
  - app.yaml
  - index.yaml
  - cron.yaml
- Static files
  - css, js, image files, etc.
- Google App Engine APIs

圖片來源: http://blogs.zdnet.com/Hinchcliffe/?p=362

如何透過GAE向世界說哈囉

<<app.yaml>>

```
application: hello-gae
version: 1
runtime: python
api_version: 1

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

<< hello.py >>

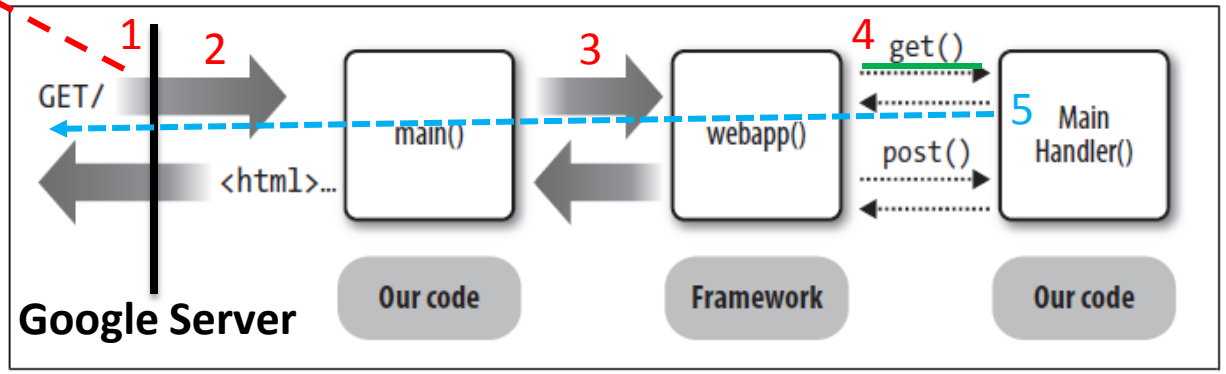
```
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class MainPage(webapp.RequestHandler):
  def get(self):
    self.response.out.write('Hello Google App Engine!')

def main():
  application = webapp.WSGIApplication([('/', MainPage)], debug=True)
  run_wsgi_app(application)

if __name__ == "__main__":
  main()
```

GET
http://your-hello-URL/



app.yaml設定檔 (1/3)

- Google App Engine專案設定檔

```
application: myapp
version: 1
runtime: python
api_version: 1

handlers:

- url: /user
script: user.py

- url: /
script: home.py
```

http://myapp.appspot.com/

專案根目錄

由home.py處理來自使用者端的要求

http://myapp.appspot.com/user

➡ 由user.py處理來自使用者端的要求

http://myapp.appspot.com/error_test

➡ 沒有指定處理類別... 錯誤發生!!

參考網頁：http://nchc-gae.blogspot.com/2009/05/appyaml-cronyaml.html

app.yaml設定檔 (2/3)

- 正規表示式基礎

. 任何字元	Eg. **/.** 可以為 /abc /1234 錯誤 / abc 1234
* 出現0次以上	Eg. **/.*** 可以為 / /abc /1234 錯誤 abc 1234
\ 跳脫字元	Eg. **/index\.html =** /index.html 錯誤 ≠ /index\1234
() 群組符號	Eg. **(\d*)** 可以為 0 1234 789 錯誤 a123 567b
\d 數字符號	
[] 出現次數	Eg. **(\d[4])** 可以為 2009 1998 錯誤 123 567b
| or符號	Eg. **(png|gif|jpg)** 可以為 png或gif或jpg 錯誤 bmp
\數字 群組編號	Eg /\1/\2 則 /abc/def 群組1為abc, 群組2為def

app.yaml設定檔 (3/3)

以正規表示式(Regular Expression)為基礎

```
……
handlers:
- url: /index\.html
script: home.py

- url: /css
static_dir: stylesheets

- url: /(.*\.(gif|png|jpg))
static_files: static/\1
upload: static/(.*\.(gif|png|jpg))

- url: /admin/.*
script: admin.py
login: admin

- url: /.*
script: not_found.py
```

http://myapp.appspot.com/**index.html**

➡ 由index.html回覆給用者

http://myapp.appspot.com/**gae.css**

➡ 根目錄下css的gae.css

http://myapp.appspot.com/**gae.png**

➡ 根目錄下 static資料夾中的 gae.png

http://myapp.appspot.com/**admin**

➡ 轉到Google登入介面，通過認證後由admin.py處理使用者的要求

http://myapp.appspot.com/**error**

➡ 由notfound.py處理來自使用者端的要求

參考網頁：http://nchc-gae.blogspot.com/2009/05/appyaml-cronyaml.html

cron.yaml設定檔

排程服務設定檔

使用範例：

```
cron:
-description: daily summary job
 url: /tasks/summary
 schedule: every 24 hours
-description: monday morning mailout
 url: /mail/weekly
 schedule: every monday of month 09:00
 timezone: Australia/NSW
```

("every"ordinal) (days) ["of" (monthspec)] (time)

特定範圍　星期幾　月/日　時間

時間格式範例:
- every 5 minutes
- every 12 hours
- 2nd,third mon,wed,thu of march 17:00
- every monday 09:00
- 1st monday of sep,oct,nov 17:00
- every day 00:00

參考網頁：http://nchc-gae.blogspot.com/2009/05/appyaml-cronyaml.html

輕鬆建立會員管理機制

User class:
- email
- nickname()
- email()
- user_id()

```
from google.appengine.api import users

class MainPage(webapp.RequestHandler):
  def get(self):
    user = users.get_current_user()
    if user:
     greeting = ("Welcome, %s! (<a href=\"%s\">sign out</a>)" %
                 (user.nickname(), users.create_logout_url("/")))
    else:
      greeting = ("<a href=\"%s\">Sign in or register</a>." %
                  users.create_login_url("/"))
    self.response.out.write("<html><body>%s</body></html>" % greeting)
```

create_login_url(dest_url)
create_logout_url(dest_url)
get_current_user()
is_current_user_admin()

```
user = users.get_current_user()
if user:
  self.response.out.write("Welcome, %s!" % user.nickname() )
  if users.is_current_user_admin():
    self.response.out.write("<a href=\"/admin/\">Go to admin area</a>“)
```

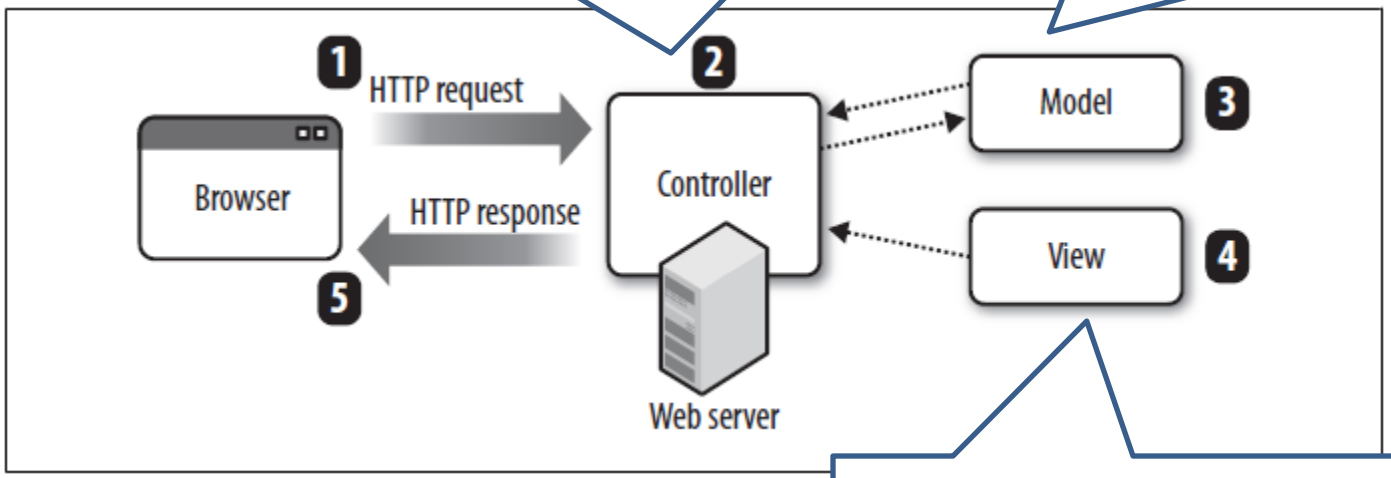
Outline

- 雲端上的引擎 － Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 － Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 － Template
- 資料黑盒子 － DataStore

MVC 設計模式

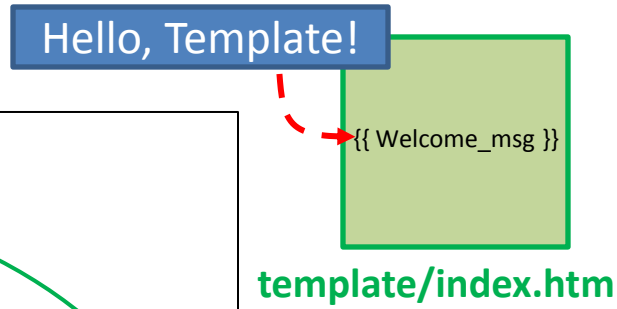
<<User-defined Handler class>>
Eg. class Main_handler(webapp.RequestHandler)

<<DataStore>>
from google.appengine.ext
import db



<<Template>>
from google.appengine.ext.webapp
import template

套用樣版



<<template/index.htm>>

```
<html>
 <head><title>Hello Template</title></head>
 <body>   {{ Welcome_msg }}   </body>
</html>
```

<<index.py>>

```
import os
from google.appengine.ext.webapp import template

class MainPage(webapp.RequestHandler):
        def get(self):
                template_value = { 'Welcome_msg': 'Hello, Template!' }
                path = os.path.join(os.path.dirname(__file__),'template/index.htm')
                outstr = template.render(path, template_value)
                self.response.out.write(outstr)
```

Outline

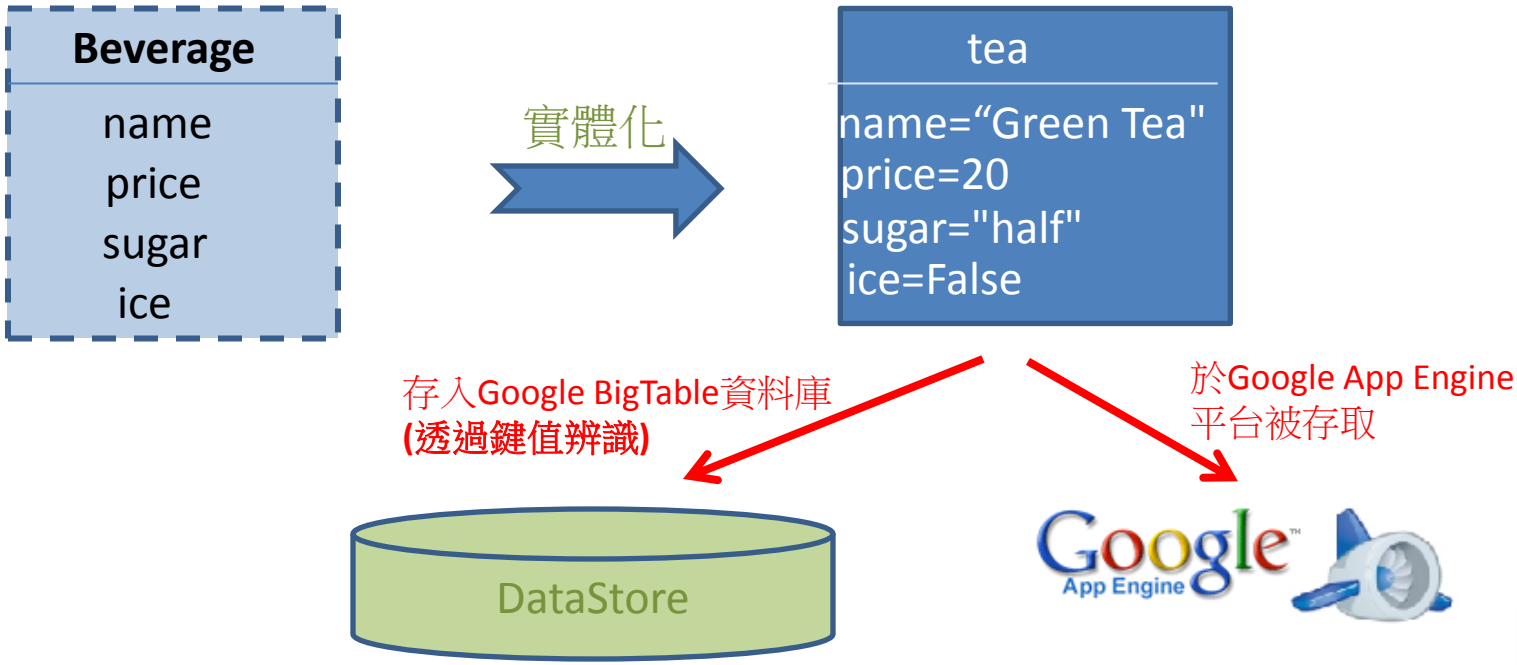
- 雲端上的引擎 － Google App Engine
- 新手駕駛飛上Google雲端
- 推動引擎的大蟒蛇 － Python
- 在Google雲端上向世界說”哈囉”
- 引擎組成元件
- 華麗的引擎外殼 － Template
- 資料黑盒子 － DataStore

資料型態與模型(1/3)

- Entity and Model

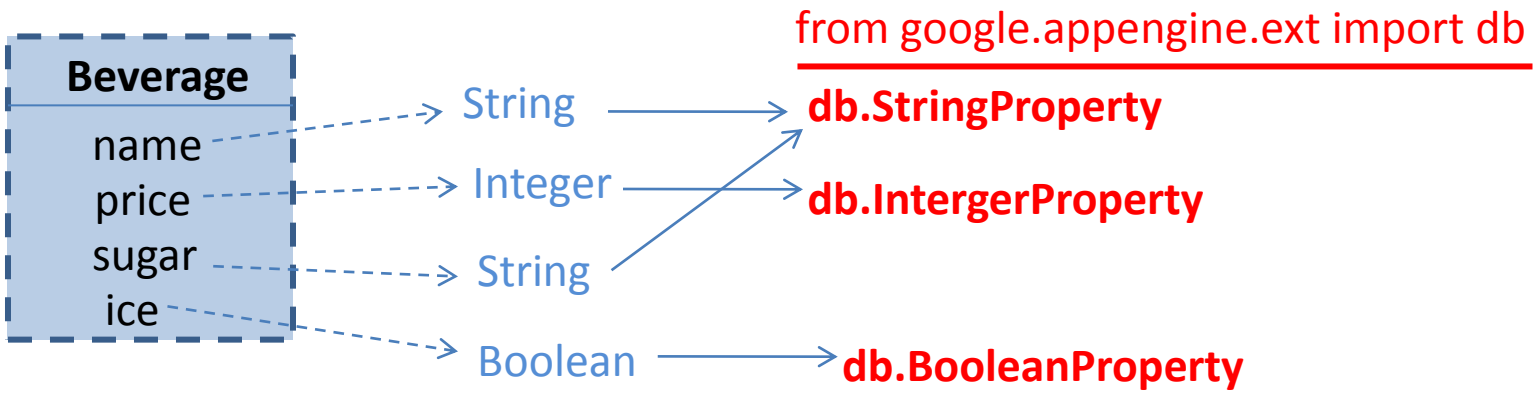
Model: 類同於物件導向中class角色

Entity: 類同於物件導向中instance角色



資料型態與模型(2/3)

- Types and Property Classes



db.ByteStringProperty, db.TextProperty, db.FloatProperty,
db.DateTimeProperty/DateProperty/TimeProperty,
db.ListProperty/StringListProperty,
db.ReferenceProperty/SelfReferenceProperty, db.RatingProperty
db.UserProperty/LinkProperty/EmailProperty/IMProperty,
db.PhoneNumberProperty/PostalAddressProperty

參考網頁：http://code.google.com/intl/en/appengine/docs/python/datastore/typesandpropertyclasses.html

資料型態與模型(3/3)

- Entity and Model

```
from google.appengine.ext import db

class Beverage(db.Model):
  name = db.StringProperty(required=True)
  price = db.IntegerProperty(required=True)
  sugar = db.StringProperty(required=True,choices=set(["full","half","none"]))
  ice = db.BooleanProperty(default=True)
```

```
tea = Beverage(name="Green Tea",
                    price=20,
                    sugar="half")
 tea.ice=False

u_key = tea.key()
```

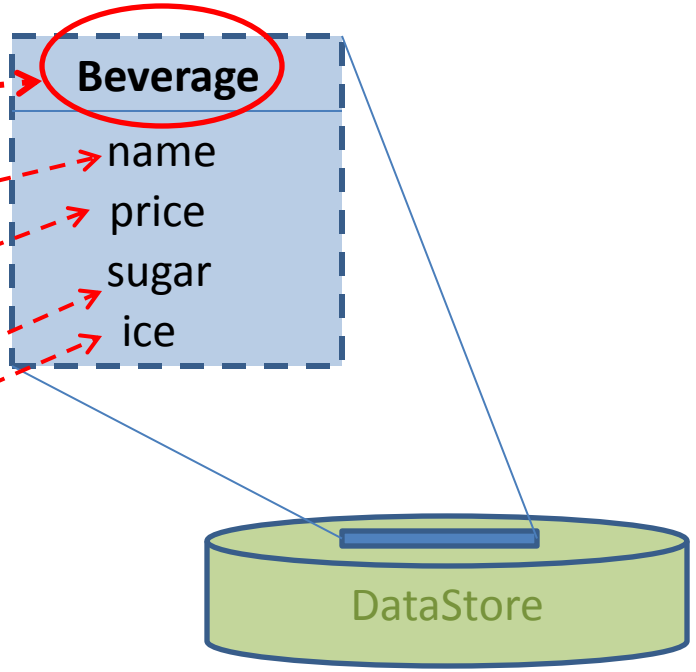
取得Entity於datastore之鍵值

index.yaml設定檔

- 索引服務設定檔

使用範例：

```
indexes:
  - kind: Beverage
    ancestor: no
    properties:
      - name: name
        direction: asc
      - name: price
        direction: desc
      - name: sugar
      - name: ice
```



資料新增刪除與取得

Creating and Deleting Data

from google.appengine.ext import db

pet.put() / db.put(pet), pet.delete() / db.delete(pet)

Getting data

```
class Story(db.Model):
  title = db.StringProperty()
  date = db.DateTimeProperty()

query = Story.all()
  query.filter('title =', 'Foo')
  query.order('-date')
  query.ancestor(key)
  = query.filter('title =', 'Foo').order('-date').ancestor(key)
```

資料搜尋

```
query = db.GqlQuery("SELECT * FROM Story WHERE title = :1 "
                    "AND ANCESTOR IS :2 "
                    "ORDER BY date DESC",
                    'Foo', key)
```

```
query = Story.gql("WHERE title = :title "
                  "AND ANCESTOR IS :parent "
                  "ORDER BY date DESC",
                  title='Foo', parent=key)
```