Google App Engine Programming Session

ae-09-session

Textbook: Using Google App Engine (Chapter 7)

First Look: Sessions are Magic!

- Sessions are usually part of the built-in web application framework
  - Ruby on Rails
  - Java Web Applications
  - PHP
- The framework does all the cookie setting and data finding

First Look: Sessions are Magic!

- In our controller code we simply ask to create and/or access a session
- We treat the session like a dictionary storing whatever we like in the session under a set of string keys that we choose

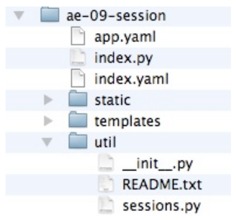
Session Best Practice

- Keep them small - we don’t want to put too much in the session or we start taxing memory and other storage resources and slowing down our application
- Focus on data that is used on nearly every incoming request - the lookup key of the current user - the email address of the current user
- Sessions generally go away when the user closes their browser (cookie is lost) or after a period of inactivity (1-3 hours)

Best Practice

- Indication of the current user - management of the login and log out process
- Shopping cart - items / quantities

Our Magic - sessions.py



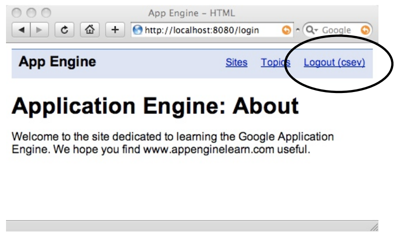
```
ae-09-session
    app.yaml
    index.py
    index.yaml
    static
    templates
    util
        __init__.py
        README.txt
        sessions.py
```

- Since the Google Application Engine does not provide a session capability, we need to add one - extending our application
- Download from

  http://www.appenginelearn.com/downloads/util.zip

- Install in your application in the directory util to make it available in your application

Using the Session



Application Engine: About

Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.

```
from util.sessions import Session

class LogoutHandler(webapp.RequestHandler):

  def get(self):
    self.session = Session()
    self.session.delete_item('username')
    doRender(self, 'index.htm')
```

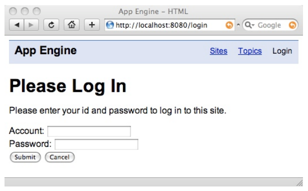
The Session() call either establishes a session or accesses the current session.

Inside the Session() call

- We use a session cookie to look up our session
- If the cookie exists and the session exists, return that session
- If not pick a large random number as the session key, make a session and set a temporary cookie with the session key as its value
- See Chapter 11 for more details

The Login/Logout Pattern

- We use a key named 'username' in the session to indicate that the user is logged in
    - If the key is missing the user is logged out
    - If the key is present, its value is the account of the logged in user (e.g. "csev")



Please Log In

Please enter your id and password to log in to this site.

Account:
Password:

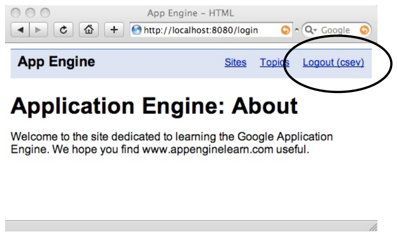
Get the Session

```
def post(self):
   self.session = Session()
   acct = self.request.get('account')
   pw = self.request.get('password')
   logging.info("Checking account="+acct+" pw="+pw)

   self.session.delete_item('username')

   if pw == "" or acct == "":
     doRender(self,"login.htm",{'error' : 'Please specify Acct/PW'} )
   elif pw == "secret":
     self.session['username'] = acct
     doRender(self,"index.htm",{ } )
   else:
     doRender(self,"login.htm",{'error' : 'Incorrect password'} )
```

Log out previous user

Log in new user

Logout



Application Engine: About

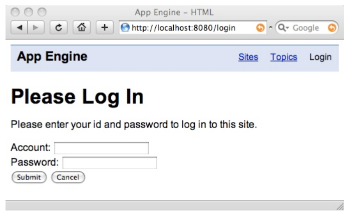
Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.

```
from util.sessions import Session

class LogoutHandler(webapp.RequestHandler):

  def get(self):
    self.session = Session()
    self.session.delete_item('username')
    doRender(self, 'index.htm')
```

Get the Session

Log out previous user

Navigation

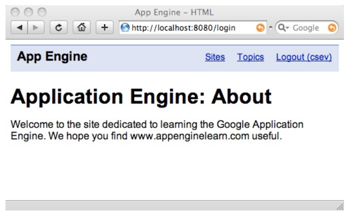


App Engine Sites Topics Login

Please Log In

Please enter your id and password to log in to this site.

Account:
Password:



App Engine Sites Topics Logout (csev)

Application Engine: About

Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.

- We want to have the Login / Logout button flip when we log in or out and we want to see the name of the current logged in user.

_base.htm

```
  <li><a href="topics.htm"
        {% ifequal path '/topics.htm' %}
              class="selected"
        {% endifequal %}
      >Topics</a></li>
{% ifequal username None %}
  <li><a href="/login"
        {% ifequal path '/login' %}
              class="selected"
        {% endifequal %}
      >Login</a></li>
{% else %}
  <li><a href="/logout">Logout ({{username}})</a></li>
{% endifequal %}
```

In the view template, we send an additional context variable to the template called "username" if the user is logged in. We use logic in the template to either generate the Login link or the Logout + account name link.

```
def doRender(handler, tname = "index.htm", values = { }):
    logging.info(tname)
    temp = os.path.join(os.path.dirname(__file__),'templates/'+tname)
    if not os.path.isfile(temp):
      return False

    # Make a copy of the dictionary and add basic values
    newval = dict(values)
    if not 'path' in newval:
        path = handler.request.path
        newval['path'] = handler.request.path

    if not 'username' in newval:
        handler.session = Session()
        if 'username' in self.session:
          newval['username'] = handler.session["username"]

    outstr = template.render(temp, newval)
    handler.response.out.write(outstr)
    return True
```

We check to see if the username is in the session and if username is in the session we add it to the context variables to be passed into the template.

Summary

- The Cookies and Session work together to give us a relatively simply way to programmatically stash data associated with a particular user/browser
- While the mechanisms are a bit complex, the session pattern turns out to be pretty simple to use in our applications
- The Google Application Engine does not provide us with a Session feature - so we need to write or borrow some code
- Clever use of session is important to application performance