Advanced Programming Techniques

Web Technologies 2

Christopher Moretti

Forms with Client-side Action

```
<html>
 <head>
   <script>
     function setfocus() { document.srch.q.focus(); }
     function loadGoog(val) { window.location.assign("http://www.google.com/
search?q="+val) }
     function loadWiki(val) { window.location.assign("http://en.wikipedia.com/
wiki/"+val) }
   </script>
 </head>
 <body onload='setfocus();'>
   <h1>Basic Lookup Form</h1>
   <form name=srch
         action="http://www.google.com/search?q="+srch.q.value>
     <input type=text size=25
            id=q name=q value="" onmouseover='setfocus()'>
     <input type=button value="Google" name=but
            onclick=loadGoog(srch.q.value)>
     <input type=button value="Wikipedia" name=but
            onclick=loadWiki(srch.q.value)>
     <input type=reset onclick='srch.q.value=""' >
   </form>
 </body>
</html>
```

http://www.cs.princeton.edu/~cmoretti/cos333/lookup.html

Actions in tags, buttons, and images

```
<html>
  <body onLoad='alert("Welcome to my page")'>
    <form>
      <input type=button value="Hit me"
             onClick='alert("Ouch! That hurt.")'>

      <input type=button value="Color me "
             onClick='document.bgColor=color.value'>
      <input type=text name=color value='type a color'>
      <input type=button value="Bleach me "
               onClick='document.bgColor="white"'>
      <br />

      <input type=text name=url size=40 value="http://">
      <input type=button value="open"
             onClick='window.open(url.value)'>
      <input type=text name=url2 size=40 value="http://">
      <input type=button value="load"
               onClick='window.location=url2.value'>
    </form>
    <img src="P.png" width="256px" onMouseover='src="tiger.jpg"'
         onMouseout='src="P.png"'>
  </body>
</html>
```

http://www.cs.princeton.edu/~cmoretti/cos333/silly.html

Whack-a-Mole

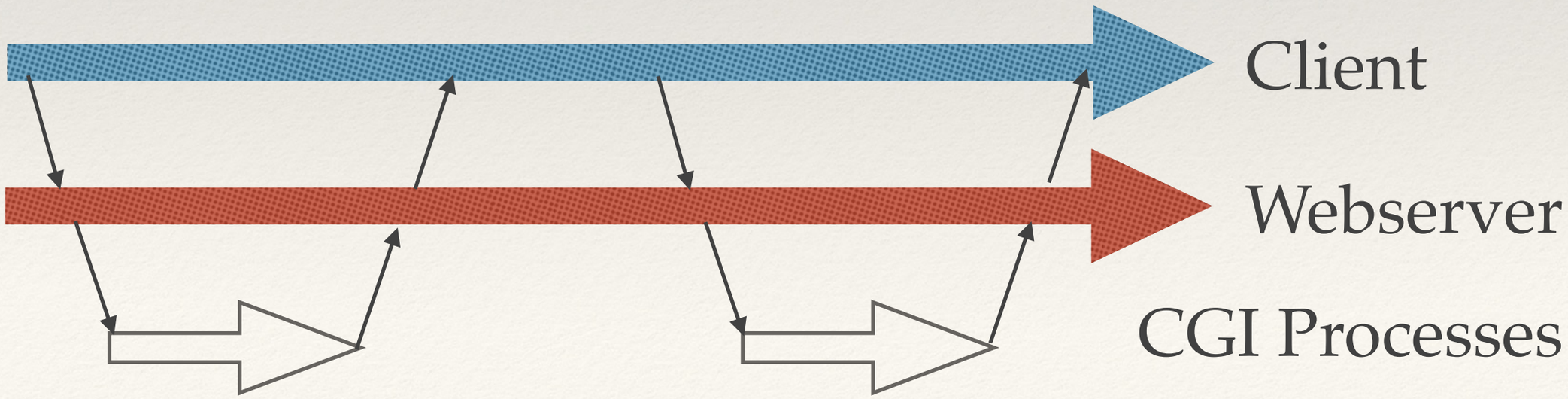
```
<html>
  <head>
    <script>
      …
    </script>
  </head>
  <body>
    <img src="mole.jpg" width="250px" id="whackee" onClick='hit()'
         style="position:absolute; left:100px; top:60px" / >
    <form>
      <p>Speed <input type=text id=interval value="1000" size=5> msec
      <p><input type=button value="Start" onClick='newgame()'>
      <p><input type=button value="Stop"
onClick='clearInterval(setint)'>
      <p><input type=button value="Change" onClick='changePic()'>

    </form>
  </body>
</html>
```

http://www.cs.princeton.edu/~cmoretti/cos333/whack.html

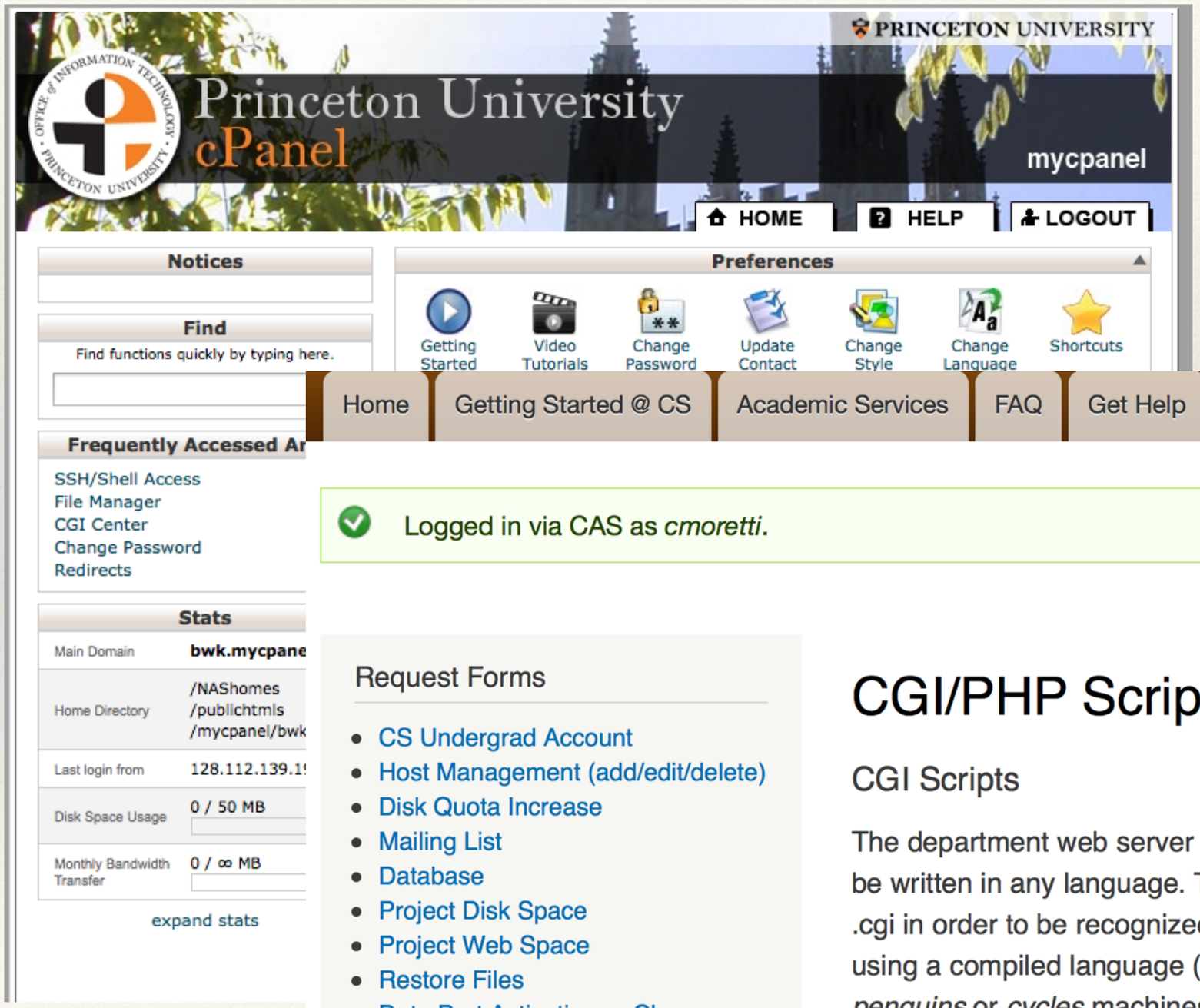
```
<html>
  <head>
    <script>
      var hits = 0, interval = 1000, index=0, setint, start, elapsed
      var pics = ["mole.jpg", "chem.jpg", "mohel.jpg", "chckn.jpg", "yale.png"]
      function newgame() {
       clearInterval(setint)
       hits = 0
       interval = document.getElementById("interval").value
       start = (new Date()).getTime()
       setint = setInterval('moveit()', interval)
       moveit()
      }
      function moveit() {
       var dog = document.getElementById("whackee")
       dog.style.left = 300 * Math.random() + "px"
       dog.style.top = 300 * Math.random() + "px"
       window.status = hits + " hits, " + ((new Date()).getTime() - start)/1000 + " sec"
      }
     function hit() {
       if (++hits > 3) {
       clearInterval(setint)
       elapsed = ((new Date()).getTime() - start) / 1000
       if (confirm(elapsed + " seconds.  Another game?", ""))
        newgame()
       }
       moveit()
      }
function changePic() {
     index = (index + 1) % pics.length
     var dog = document.getElementById("whackee")
     dog.src = pics[index]
     }
   </script>
```

Common Gateway Interface: CGI

- Ask the server to run a program with client's input
  - Typically via an HTML form
  - Program on server must be executable, but can be in any language — most have libraries for parsing input
- Program produces HTML to send back to client



CGI at Princeton



Getting Started: Front-end Forms

```
<html>
  <form action="hello1.cgi" method="POST">
    <input type="submit" value="hello1: bash, plain">
  </form>

  <form action="hello2.cgi" method="POST">
    <input type="submit" value="hello2: bash, html">
  </form>
</html>
```

http://www.cs.princeton.edu/~cmoretti/cos333/hello1.html

Getting Started: Back-end Scripts

```
#!/bin/bash
echo 'Content-Type: text/plain’
echo
echo ‘Hello, world!’
```

```
#!/bin/bash
echo 'Content-Type: text/html
<html>
 <title>Hello2</title>
 <body>
  <h1 style="color: #ff0000">Hello, world!</h1>'
echo '  <h2 style="background-color: #000000; color:#bbbbbb">Right now
it is' "`date`" '</h2>'
echo ' </body>
</html>'
```

User-submitted Data - Front-end Forms

```
<html>
 <title> COS 333 Survey </title>
 <body>
 <h2> COS 333 Survey </h2>
 <form method=GET action="survey.cgi"
Name: <input type=text name=Name size=40> <p>
Class: <input type=radio name=Class value=16> '16
       <input type=radio name=Class value=17> '17
       <input type=radio name=Class value=18> '18
<p> CS courses:
<input type=checkbox name=c126> 126
<input type=checkbox name=c217> 217
<input type=checkbox name=c226> 226
<p> Experience?
<textarea name=Exp rows=3 cols=40 wrap></textarea>
<p>
<input type=submit> <input type=reset>
</form>
</body></html>
```

http://www.cs.princeton.edu/~cmoretti/cos333/survey.html

User-submitted Data - Back-end Script

```
#!/usr/bin/python

import os
import cgi
form = cgi.FieldStorage()

print "Content-Type: text/html"
print ""
print "<html>"
print " <title> COS 333 Survey </title>"
print " <body>"
print "  <h1> COS 333 Survey </h1>"
for i in form.keys():
        print "  %s = %s <br>" % (i, form[i].value)
print "<p>"
for i in os.environ.keys():
        print "  %s = %s <br>" % (i, os.environ[i])
print " </body>"
print "</html>"
```

Oops …

"Please also take care in writing your scripts with regards to security. Please keep security in mind during the writing of your script to avoid exposing directories that might allow files to be placed on the server, exposure of system files or information, or other such unintended effects."

–CS Guide (https://csguide.cs.princeton.edu/publishing/cgi)

HTTP Request Methods

- `GET` - retrieve data with no other effects.
  - Visible in URL, browser logs; limited length
  - `name=value;` pairs, `keyword+keyword+…` lists
- `POST` - submits "posted" content to the server.
  - Data is enclosed in request read into server on stdin
  - More flexible, but can't be bookmarked, etc.
- `HEAD`, `PUT`, `DELETE`, `CONNECT`, etc.

About that POST flexibility …

- Defensive programming is important!

```
char postString[1024];
contentLength =
atoi(getenv(“CONTENT_LENGTH"));
cin.read(postString,contentLength);
```

(Actual text from a C++ book, ca. 2003)

- “Always validate all your inputs — the world outside your function should be treated as hostile and bent upon your destruction.”

  Howard & LeBlanc, *Writing Secure Code*

Yay 226: http://nifty.stanford.edu/2016/wayne-autocomplete-me/

XMLHttpRequest ("XHR")

- Client-Server interactions are usually synchronous
  - incurs significant delay, complete page re-dray
- XHR provides *asynchronous* communication with server
- Core of AJAX: Asynchronous Javascript and XML
- First widespread use '05 by Google (Maps, Mail, Suggest)
  - "The real importance of Google's map and satellite program, however, is not its impressive exterior but the novel technology, known as Ajax, that lies beneath."

(James Fallows, *NY Times*, 4/17/05)

Structure of AJAX in Browser

```
var req;
function geturl(s) {
   if (s.length > 1) {
      url = ‘http://www.cs.princeton.edu/~cmoretti/cos333/phone.cgi?' + s;
      loadXMLDoc(url);  // loads asynchronously
   }
}
function loadXMLDoc(url) {
   req = new XMLHttpRequest();
   if (req) {
      req.onreadystatechange = processReqChange;
      req.open("GET", url);
      req.send(null);
   }
}
function processReqChange() {
   if (req.readyState == 4) {    // completed request
      if (req.status == 200)     // successful
         show(req.responseText); // could be responseXML
   }
}
function show(s) {   // show whatever came back
   document.getElementById("place").innerHTML = s
}
```

Structure of AJAX in Browser

```
var req;
function geturl(s) {
   if (s.length > 1) {
      url = ‘http://www.cs.princeton.edu/~cmoretti/cos333/phone.cgi?' + s;
      loadXMLDoc(url);  // loads asynchronously
   }
}
function loadXMLDoc(url) {
   req = new XMLHttpRequest();
   if (req) {
      req.onreadystatechange = function() {
         window.status = req.statusText;
         if (req.readyState == 4) {  // completed request
            if (req.status == 200)   // successful
               show(req.responseText);
         }
      };
      req.open("GET", url);
      req.send(null);
   }
}
function show(s) {   // show whatever came back
   document.getElementById("place").innerHTML = s
}
```

Note: no longer works on web because new CS web server doesn't have ldapsearch, but script works on penguins

Server Script

```
q1=`echo $QUERY_STRING | gawk '{split($0,x,"%20"); print x[1]}' `
q2=`echo $QUERY_STRING | gawk '{split($0,x,"%20"); print x[2]}' `
/usr/local/bin/ldapsearch -x -h ldap.princeton.edu -u -b \
   o='Princeton University,c=US' "(cn=*$q1*)"  uid cn telephoneNumber \
            studenttelephoneNumber studentstreet street ou |
php -r '
while (!feof(STDIN)) {
   $d = (fgets(STDIN));
   if (preg_match("/^#/", $d)) continue;
   if (preg_match("/^dn:|^ufn:/", $d)) continue;
   if (preg_match("/^cn:/", $d))
      if (strlen($d) > strlen($cn)) $cn = $d;
   if (preg_match("/telephoneNumber|street/", $d))
      $out = $out . " " . trim($d);
   if (preg_match("/^ou:/", $d)) $out = $out . " " . trim($d);
   if (strlen(trim($d))==0 && strlen($cn . $out) > 0) {
      $out = trim($cn) . " " . $out;
      $out = preg_replace("/Undergraduate Class of/", "", $out);
      $out = preg_replace("/cn:|ou:|telephoneNumber:|(student)?street:/", "", $out
      $out = preg_replace("/@Princeton.EDU/", "", $out);
      print "$out\n";
      $out = $cn = "";
   }
}' | grep -i ".*$q2" | sed -e /Success/d
```

Simpler Server Script

```
#!/bin/sh

echo "Content-Type: text/html"; echo

q1=`echo $QUERY_STRING |
      gawk '{ n=split($0, x, "%20"); print x[1]}'`
q2=`echo $QUERY_STRING |
      gawk '{ n=split($0, x, "%20"); print x[2]}'`
q3=`echo $QUERY_STRING |
      gawk '{ n=split($0, x, "%20"); print x[3]}'`

grep -i "$q1" phone.txt |
grep -i ".$q2" |
grep -i ".$q3"
```

Much simpler with pre-computed data …

http://www.cs.princeton.edu/~cmoretti/cos333/unphonebook2.html

www.cs.princeton.edu/~cmoretti/cos333/maps.html

Satellite

The Aquinas Institute - Princeton University's...

William St

Charlton St

You are here, more or less. Unless you're skipping class >:(

Princeton University Press

Shapiro Walk

Map data ©2016 Google Imagery ©2016 | Terms of Use | Report a

Google Maps API

```
<style type="text/css">
  html { height: 100% }
  body { height: 100%; margin: 0px; padding: 0px }
  #map { height: 100% }
</style>
<script type="text/javascript"
    src="http://maps.google.com/maps/api/js?sensor=true">
</script>
<script type="text/javascript">
  function initialize() {
    …
  }
</script>
</head>
<body onload="initialize()">
  <div id="map" style="width:100%; height:100%"></div>
```

Google Maps API

```
…
<script type="text/javascript"
    src="http://maps.google.com/maps/api/js?sensor=true">
</script>
<script type="text/javascript">
  function initialize() {
    var latlong = new google.maps.LatLng(40.35019, -74.65308);
    var opts = {
      zoom: 20, center: latlong,
      mapTypeId: google.maps.MapTypeId.HYBRID };
    var map = new google.maps.Map(document.getElementById("map"), opts);
    var marker = new google.maps.Marker({
      position: latlong, map: map, title:”Friend 101" });

    var infowindow = new google.maps.InfoWindow({
      content: "You are here, more or less. Unless you're skipping class >:("
    });
    marker.addListener('click', function() {
      infowindow.open(map, marker);
    });
  }
</script>
…
```

http://www.cs.princeton.edu/~cmoretti/cos333/maps.html

Avoid "Cargo Cult" Programming

- The temptation is there with such a broad class & project, but be careful that you really know what the code you steal does
- "What's the name of the --> operator?"

```
#include <stdio.h>
int main()
{
    int x = 10;
    while( x --> 0 ) // x goes to 0?
    {
      printf("%d ", x);
    }
}
```