§Web Programming

## Topic: Python CGI

## Introduction

## This worksheet is designed to encourage you to develop your web page writing and design skills using Python CGI.

Your tutor will help you by demonstrating what is required and also answer any questions you may have.

**Pre-requisite:**

1. Your memory stick is ready with xampp and Python34.

2. You need a web server (e.g. apache) to load your python CGI script. This means you should save your python scripts in htdocs/webprog folder in xampp and run it in web browser by typing URL e.g. <http://loaclhost/webprog/yourprogram.py>. Please make sure apache server is running.

3. You have practiced HTML exercises and are familiar HTML elements especially <Form> element.

**Post-requisite:** You should stop apache server and exit from xampp control panel. Also, you should properly unplug your memory stick. It is always a good idea to zip your webprog folder and email it to yourself or make a copy e.g. in Dropbox.

If you need help - **ask**. If you are curious - **ask**. If you want to know - **ask**.

You are encouraged to discuss the workshop with other students in the class. Share and Learn!

**IMPORTANT:** You must check if shebang line is referring to correct version of Python34 folder on your memory stick. Shebang line should be something like: #! /Python34/python

## What you must do

Again you should work in pairs (two or three students). Download pythoncgi.zip in webprog folder. There are many python cgi examples included in the extracted folder. Read and understand all programs, load them in a webbrowser to see the outcome. If anything is not clear then ask!

**Objective here is to learn basic of python CGI to develop basic web pages.**

* Make sure you have access to the relevant sites/lectures
* Make sure you have bookmarked links to web help pages and references
* Quickly read python cgi examples and/or any lectures on html (this week).

Examples included are (don’t forget to change shebang line – see above note):

***Python3cgibasic.py*** 🡪 creates a simple web page using python cgi and displays ‘Hello World’

***demo.py*** 🡪 Creates a simple HTML page using multiple print statements. Try to understand how a variable value is embedded in HTML contents.

***p3cgi1.py*** 🡪 demonstrates use of multiline string to construct full HTML web page and print it using python print statement.

***demo1.py*** 🡪 Creates a simple HTML page using multiline string variable and creating template to embedded another variable value in HTML contents.

***helloWorldFile.py*** 🡪 Creates simple HTML page contents using print statements. Also, creates a text file and writes a sentence into it. Then reads contents of the file and displays on the webpage.

***cgiHotel.py and hotel.html*** 🡪 cgiHotel.py reads hotel.html file and displays its contents as webpage.

***helloWorldcgi.py*** 🡪 prints all system environment variables in a web page

***templatecgip3.py*** 🡪 demonstrates how to create templates and fill data values using specific variable names

***dumpsVar.py*** or ***dumps3.py*** 🡪 receives any number of form values from client side and displays it on a web page. Note this script is useful for debugging your client server scripts. For example, dumpsform.html sends two input values id and name and dumpsVar receive them and show on a webpage.

***helloWorldoneform.html*** and ***helloWorldoneform.py*** 🡪 demonstrate an example of HTML form and server side scripting.

***simplecookiep3.py*** and ***session\_cookiep3.py*** 🡪 examples of cookies for session management

***usefulexample*** folder has two programs ***webexamplclient.html*** and ***webexampleserver.py*** (these are in two version webexampleserver-v1.py and webexapleserver-v2.py) and a sub-folder ***example*** where there is one text file ***registration.txt*** 🡪 This program demonstrates getting form data from client side and processing it on server side. Server also saves the data in a text file and later reads data from the file and displays it on the webpage in an HTML table.

## Exercise 1 (Basic python CGI):

Create a Module specification web page using Python CGI (you can reuse html contents from last week’s practical session). Follow these steps:

1. include shebang
2. import cgi
3. print header i.e. print (‘content-type: html/text’)
4. print an empty line i.e. print()
5. create html elements/tags using print statement

## Exercise 2 (Dynamically generating web pages):

**a.** Create a text file e.g. content.txt with following contents:

Module code, Module name, Assignment marks, Exam marks

ABC123, Web programming, 30, 70

DEF456, CaNS, 50, 50

GHI789, OOSD, 50, 50

JKL012, PoC, 40, 60

**b.** Save content.txt in webprog folder.

**c.** Create a python cgi script that reads content.txt and displays its contents in a tabular format on a webpage. You can follow following steps:

i). Read file contents

ii). Use regular expression to split each row into comma separated items i.e. for first row there are four items ‘Module code’, ‘Module name’, ‘Assignment marks’ and ‘Exam marks’. (Hint: you may use re.split() function)

iii). Use first row of the file to create html table header. (e.g. <th>)

iv). Use all other rows of the file to fill the data in table cells. (e.g. <tr> and <td>)

Hint: Do not forget to use print statement for steps (iii) and (iv).

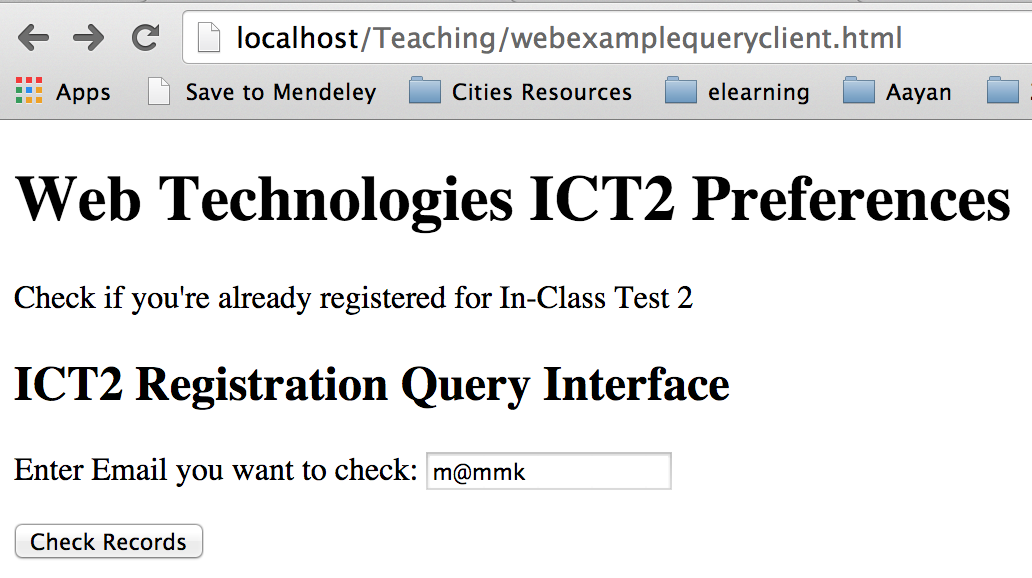
## Exercise 3 (Querying Data from a File on Server):

**Assumption:** you have fully understood programs in ***usefulexample***. You have tried to run webexampleclient.html with v1 and then v2 and have spotted the difference in behaviour of both programs. Then you have looked at the code of both programs to understand the difference. This will help you a lot when you’ll start working on your coursework assignment.

Let’s assume you need to create a query interface which can search data from the registration.txt file on the server and generate proper output.

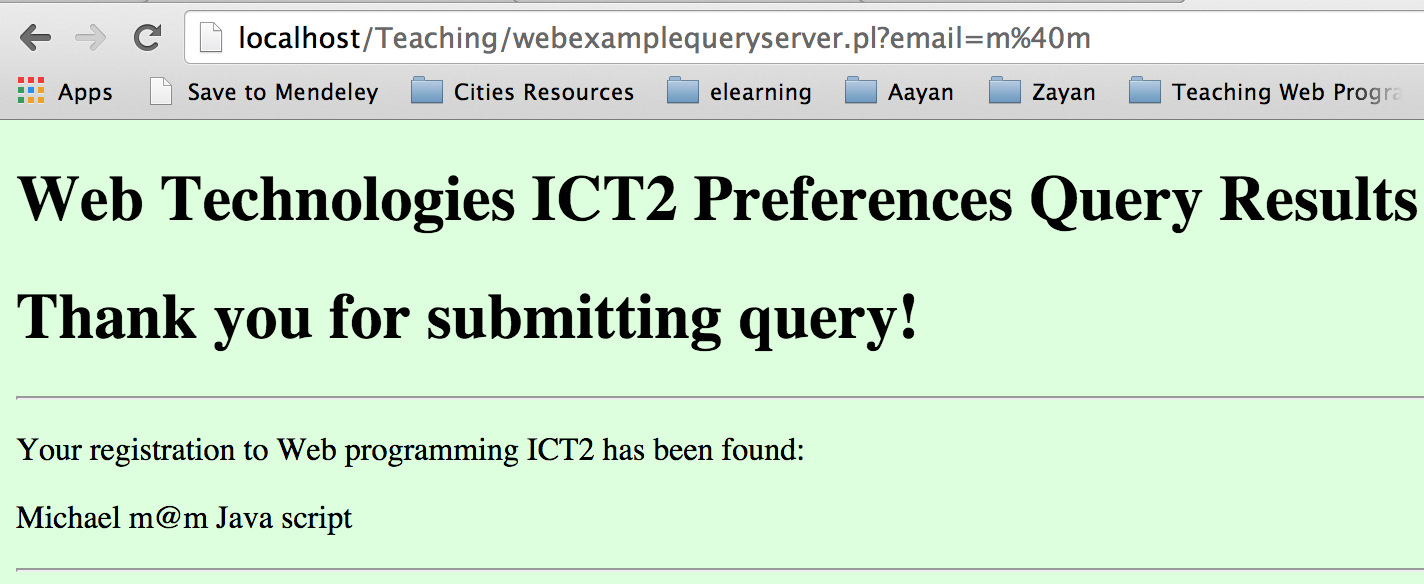
In ***usefulexample*** folder there is another file named: ***webexamplequeryclient.html***. Use this html file to get email address. Create a server side script ‘***webexamplequeryserver.py***’ that should receive email address and check in the registration.txt file if somebody has already registered his/her interest for the ICT2.

***webexamplequeryclient.html*** output looks like:

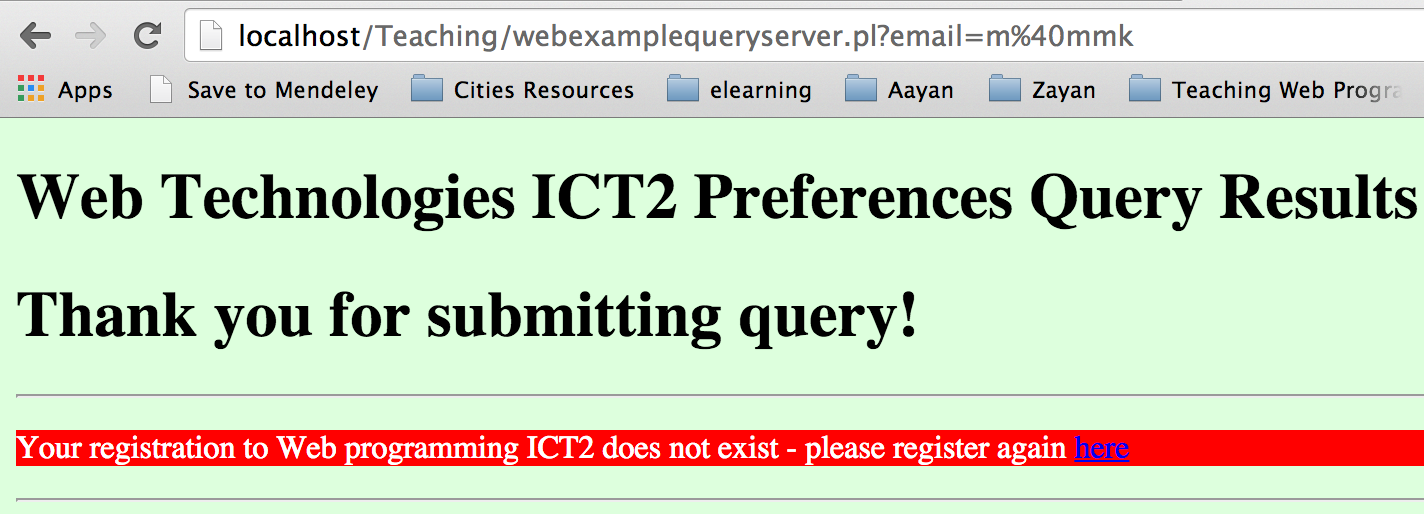


You program ***webexamplequeryserver.py*** should generate server side output as below:

(when record is found in the registration.txt file)



(when record is not found in the registration.txt file)



## Exercise 4 (Find and Replace feature):

Create a form (findreplace.html) that gets two strings: ‘find’ and ‘replace’ keywords in two text fields. When ‘submit’ button is pressed, ‘find’ and ‘replace’ are sent to ‘findreplace.py’ server script. findreplace.py server script reads ‘test2\_animals.txt’ (see pythonexamples from previous practical), searches the ‘find’ string and displays contents of the file on a webpage by replacing the matching keyword with ‘replace’ string.

**Note: Please complete all above exercises (especially exercise 4) and get ready for next In Class Test.**