

**CS1315:
Introduction to
Media Computation**

HTML:
How to automatically generate
HTML

Content

- Generating home pages using boilerplate HTML intermingled with personalized content
- Generating a samples (thumbnail) page
- Inserting current data (e.g. temperature) into web page

HTML is not a *programming* language

- Using HTML is called “coding”.
- But HTML has no
 - **Loops**
 - **IFs**
 - **Variables**
 - **Data types**
 - **Ability to read and write files**
- So HTML does not specify *process*!

But we can use programs to *generate* HTML

```
def makePage():
    file = open("generated.html", "wt")
    file.write("<<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML
4.01 Transition/EN\"
http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>The Simplest Possible Web Page</title>
</head>
<body>
<h1>A Simple Heading</h1>
<p>Some simple text.</p>
</body>
</html>")
    file.close()
```

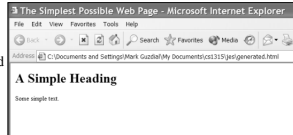
As far as Python is concerned,
this is just a big text string

A Generated Page

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>The Simplest Possible Web
Page</title>
</head>
<body>
<h1>A Simple Heading</h1>
<p>Some simple text.</p>
</body>
</html>

```



Tailoring the output

- Why would you want to have a program generate what you can enter manually with a text editor?
 - **Q: Why you write programs:**
 - **A: When you have to do the same thing many times, maybe with some variation and customization!**
- E.g. We'll make generator of home pages that are tailored for individuals.
 - **A home page should have *your* name, and other personal information (e.g. one of your interests).**
- But first...how would you tailor any text output?

A homepage generator

```

def makeHomePage(name, interest):
    file = open("homepage.html", "wt")
    file.write("<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>"" + name + "'s Home Page</title>
</head>
<body>
<h1>Welcome to "" + name + "'s Home Page</h1>
<p>Hi! I am "" + name + "", This is my home page!
I am interested in "" + interest + ""</p>
</body>
</html>""")
    file.close()

```

More useful than a Spam generator, but the same principle. Most text is boilerplate

But this is customized

makeHomePage("Dinsdale", "organized crime")

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Molly's Home Page</title>
</head>
<body>
<h1>Welcome to Dinsdale's Home
Page</h1>
<p>Hi! I am Dinsdale. This is my
home page!
I am interested in organized crime</p>
</body>
</html>

```

makeHomePage("George P. Burdell","removing T's, driving old cars, and swimming.")

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN"
"http://www.w3.org/TR/html4/loose.d
td">
<html>
<head>
<title>George P. Burdell's Home
Page</title>
</head>
<body>
<h1>Welcome to George P. Burdell's
Home Page</h1>
<p>Hi! I am George P. Burdell. This is
my home page!
I am interested in removing T's, driving
old cars, and swimming.</p>
</body>
</html>

```



This Works...but it's painful

- Suppose we try to change the home page code
 - **E.g. Maybe insert a picture, or another line about interests, or a favorite URL.**
- It would be hard and error-prone
 - **It's hard to track down all those quotes, insert the +'s and variables in the right place.**
 - **It's one very long string.**
- Can we make it easier to work with?
 - **Yes. Use multiple functions and think about problem using top-down programming!**

New Homepage Program

```

def makeHomePage(name, interest):
    file=open("homepage.html","wt")
    file.write(doctype())
    file.write(title(name+"s Home Page"))
    file.write(body(name,interest))
    file.close()

def doctype():
    return "<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN"
"http://www.w3.org/TR/html4/loose.dtd">"

def title(titlestring):
    return "<html><head><title>"+titlestring+"</title></head>"

def body(bodystring):
    return "<body>"+bodystring+"</body></html>"

```

Up here on top is where we deal with the parts that we might likely change.

Bury the yucky doctype here — may we never deal with it again!

Here are more details we don't really want to deal with.

makeHomePage("George P. Burdell","removing T's, driving old cars, and swimming.")

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transition//EN"
"http://www.w3.org/TR/html4/loose.dtd"><html><head><title>G
eorge P. Burdell's Home
Page</title></head><body>
<h1>Welcome to George P. Burdell's
Home Page</h1>
<p>Hi! I am George P. Burdell.
This is my home page!
I am interested in removing T's,
driving old cars, and
swimming.</p></body></html>

```



Content

- Generating home pages using boilerplate HTML intermingled with personalized content
- Generating a samples (thumbnail) page
- Inserting current data (e.g. temperature) into web page

Where can we get Web content from? ANYWHERE WE WANT!

- We've learned a lot of ways of generating textual information over the last weeks.
- We can use these to create all kinds of Web pages.
 - **Grabbing information out of directories using the os module**
 - **Grabbing information out of other Web pages**
 - **Generating random sentences**
 - **Generating Web pages from databases**

Generating a samples page

```
import os

def makeSamplePage(directory):
    samplesfile = open(directory + "//samples.html", "wt")
    samplesfile.write(doctype())
    samplesfile.write(title("Samples from " + directory))
    # Now, let's make up the string that will be the body.
    samples = "<h1>Samples from " + directory + " </h1>\n"
    for file in os.listdir(directory):
        if file.endswith(".jpg"):
            samples = samples + "<p>Filename: " + file
            samples = samples + '<head-><title->" + titlestring + "</title-></head->"

def body(bodystring):
    return "<body->" + bodystring + "</body-></html->"
```

Just the part that we care about

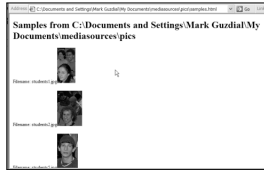
Take these functions for granted. They're the same as before

"Just the part I care about" is how you should think about programming.

- Once you write the utility functions, remember them just the way you remember library functions like `open()` and `getSampleValueAt()`
 - **They do a job for you.**
 - **Don't worry about how they do it. Once you have got them right, take them for granted.**
- This allows you to focus on the *important* parts.
 - **The parts you care about.**

makeSamplePage("C:\Documents and Settings\Mark Guzdial\My Documents\mediasources\pics")

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
HTML 4.01 Transitional/EN"
"http://www.w3.org/TR/html4/loose.dtd"
><html><head><title>Samples from
C:\Documents and Settings\Mark
Guzdial\My
Documents\mediasources\pics</title></hea
d><body><h1>Samples from
C:\Documents and Settings\Mark
Guzdial\My Documents\mediasources\pics
</h1>
<p>Filename: students1.jpg</p>
<p>Filename: students2.jpg</p>
<p>Filename: students5.jpg</p>
<p>Filename: students6.jpg</p>
<p>Filename: students7.jpg</p>
<p>Filename: students8.jpg</p>
</body></html>
```



Content

- Generating home pages using boilerplate HTML intermingled with personalized content
- Generating a samples (thumbnail) page
- Inserting current data (e.g. temperature) into web page **Next time.....**

You don't need a browser to use the WWW

- Python (and other languages) have modules that allow you to access the Internet.
 - In Python, we can read from any website as if it were a file on our computer
 - So, instead of downloading the source of a web page before we run a program that reads from that file, we can read from the website live.
 - A website's location is a URL ("universal resource locator"), so the Python module is called *urllib*.
 - E.g. www.gatech.edu is a URL

Opening a URL and reading it

```
>>> import urllib
>>> connection = urllib.urlopen("http://www.ajc.com/weather")
>>> weather = connection.read()
>>> connection.close()
```

Note the resemblance to opening a file for reading, reading it, and closing it.

The module and functions have different names, but that's all.

