Outline

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

MCS 260 Lecture 18
Introduction to Computer Science
Jan Verschelde, 22 February 2016

networking and the internet
markup languages

The World Wide Web

WWW: historical development

- 1970s: development of TCP/IP = Transmission Control Protocol/Internet Protocol.
  Main benefit: capability of electronic mail (*email*).
- mid 1980s: connections between computer facilities.
  In 1989, Tim Berners-Lee of CERN developed HTML, HyperText Markup Language.
- Mosaic was the first web browser developed at NCSA, released in 1993, leading to Netscape.
  Search engines originated at the end of the nineties.
- Web 2.0: publishing $\rightarrow$ participation.
  Wiki is server software that allows users to freely create and edit Web pages using any Web browser.

networking and the internet
markup languages

network addresses and domain names

Each node on the internet has an *IP address*.
An IP address consists of four bytes.

Each node has a symbolic name.
For example, `people.uic.edu`.
The `edu` stands for universities. The other principal domains are `gov` (government), `mil` (military), `com` (commercial), and `org` (other organizations).

The command `nslookup people.uic.edu`
or `host people.uic.edu` returns
the numerical IP address: 128.248.156.140.

The Internet is a *Wide Area Network* (WAN), linking machines over a greater distance. A *Local Area Network* (LAN), links computers in one room or building.

Uniform Resource Locator (URL)

A URL is an addressing scheme
to provide a path to an internet resource.

Example: `http://www.math.uic.edu/~jan/mcs260.html`.

The format of a URL is

`protocol://host.domain-name/path/dataname`

where

- `protocol` refers to the type of protocol to be used
- `host` refers to the server where the resouce is stored
- `domain-name` contains the name and type of the domain of the server
- `path/dataname` refers to the location of the data

To preview pages offline, use the protocol `file`.

networking and the internet
markup languages

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

Client/Server Networks

A *client* is a computer in the network that *requests* for access to data and services from another computer.

A *server* is a computer in the network *receives and processes* requests from clients.
Access permissions are determined by the server.

A *client/server network* consists of several computers connected in a network, acting as clients and/or servers.

Client/server computing emerged in the nineties to distribute applications (such as database administration) over a network.

Hypertext Transfer Protocol (HTTP)

exchange data between the client and the server

HTTP is based on request-response between
a web browser (the client) and a web server.
A typical transaction between browser and server:

1. A TCP/IP connection is established
   between browser and server.
2. The browser sends a request for a web page.
3. The server locates the file and responds,
   sending the content of the requested web page.
4. The TCP/IP connection is closed.

creating your own homepage

ACCC provides a web publishing service, visit

1. `http://accc.uic.edu/service/web-publishing/personal`
   click on Activate my site, which asks you to login with your netid.
2. After entering netid and corresponding password, agree to be nice and click on the Activate <netid>.people.uic.edu (where <netid> is replaced by your netid.
   When it works, you get to the "Site summary" web page.
3. Point your web browser to `http://<netid>.people.uic.edu` where `<netid>` is your netid.
4. The hostname is `people.uic.edu`. With `ssh` (secure shell) we can login, as `ssh people.uic.edu`.
5. Version 2.7.5 of Python is installed on the Linux computer.
6. Personal web pages are stored in the subdirectory `public_html` in your homedirectory.

networking and the internet
markup languages

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

Markup Languages

to write web pages

To write web pages, we use

HTML hypertext markup language
written to display information, the language in which web pages are written.

XML extensible markup language
XML is a widely supported open technology for describing data.

MathML mathematical markup language
MathML can display complex mathematical expressions. As it is created with XML, MathML is a so-called *XML vocabulary*.

The world wide web consortium (`http://www.w3c.org`) is a source for many protocol standards for the web.

our first web page: hello_world.html

```
<HTML>
<HEAD>
<TITLE> MCS 260 Lec 9.5: hello world </TITLE>
</HEAD>

<BODY>
<H1> Hello World! </H1>

This is our <EM>first</EM> web page!

<P>
To go to <A HREF="http://www.uic.edu">
UIC's home page </A>, click on the
underlined text.

</BODY>
</HTML>
```

secure copy: `scp`

To see this web page on our site, we use secure copy `scp` to transfer the file `hello_world.html` into the directory `public_html` at `people.uic.edu`.

For example, in a Terminal window on a Mac OS X:

```
$ scp hello_world.html janv@people.uic.edu:~/public_html
janv@people.uic.edu's password:
hello_world.html                              100%  252
$
```

To see how the page looks, point your browser to
`http://janv.people.uic.edu/hello_world.html`

XML to exchange data: hello_world.xml

XML focuses on data, not its formatting

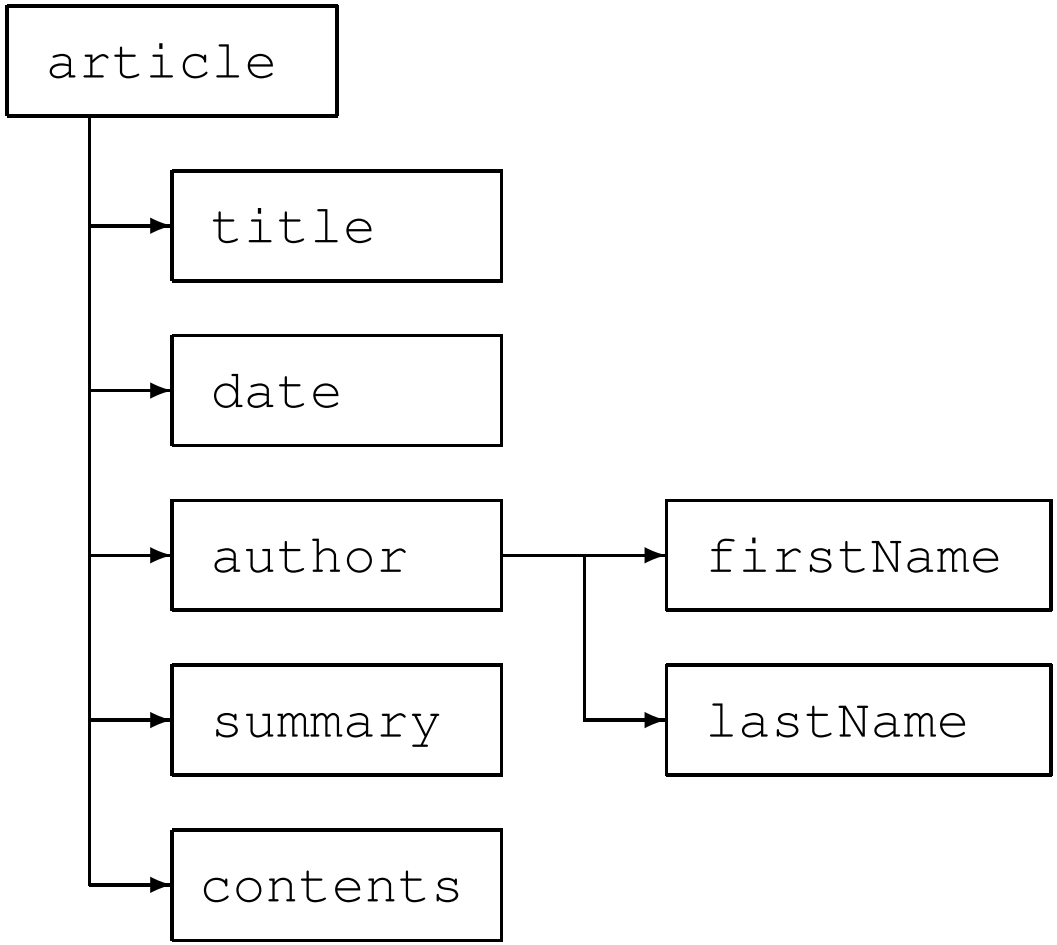
```
<xml>
<head>our first xml example</head>
<body>
<article>
<title>hello world</title>
<date>23 February 2015</date>
<href>http://www.uic.edu</href>
<content>This is our first XML example.</content>
</article>
</body>
</xml>
```

A browser will display *the document tree*.

XML + HTML = XHTML

Document Object Model (DOM)

The tree structure for `article.xml`:



networking and the internet
markup languages

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

Typesetting Mathematical Expressions

Mathematical expressions like

$$\int_{-1}^{10} \frac{e^{-x^2} \sin(20x)}{\sqrt{2+x^8}} dx$$

are encoded with LaTeX as

```
\int_{-1}^{10} \frac{e^{-x^2} \sin(20 x)}
                    {\sqrt{2+x^8}} dx
```

With Maple (also to produce LaTeX code):

```
> f := Int(exp(-x^2)*sin(20*x)/sqrt(2+x^8),x=-1..10);
```

Menu: `File`, `Export As`, choose `HTML with MathML`
to generate MathML.

MathML generated by Maple

```
<p align="center">
<math xmlns='http://www.w3.org/1998/Math/MathML'>
  <mrow>
    <mi>f</mi>
  </mrow>
  <mo>:=</mo>
  <mrow>
    <mrow>
      <munderover>
        <mo>&Integral;</mo>
        <mn>-1</mn>
        <mn>10</mn>
      </munderover>
...
```

encodes $f := \int_{-1}^{10}$.

networking and the internet
markup languages

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

python forecast.py

```
$ python forecast.py
opening http://weather.noaa.gov//pub/data/forecasts/state//il/ilz013.txt ...

   TODAY    TUE      WED      THU      FRI      SAT      SUN
   FEB 23   FEB 24   FEB 25   FEB 26   FEB 27   FEB 28   MAR 01

   CHICAGO DOWNTOWN
   VRYCLD   FLRRYS   VRYCLD   VRYCLD   VRYCLD   PTCLDY   SNOW
     /11    7/28     11/18    6/14     4/17     6/24     20/35
      /00    00/30    10/00    30/20    10/10    10/10    50/50

   CHICAGO OHARE
   VRYCLD   FLRRYS   VRYCLD   VRYCLD   VRYCLD   PTCLDY   SNOW
     /11    7/29     11/18    6/14     2/17     6/24     19/35
      /00    00/30    10/00    20/20    10/10    10/20    50/50
```

data from the web

The module `urllib` exports `urlopen`,
`urlopen` returns a file like object.

Template for retrieving data from web pages:

```
from urllib import urlopen
url = ' < internet address > '
f = urlopen(url)
s = f.readline()
```

To get the weather forecast: `http://www.weather.gov/`
National Oceanic and Atmospheric Administration's National Weather Service

The web site provides data for downloading:
`http://weather.noaa.gov/pub/data/`

the script `forecast.py`

```
from urllib.request import urlopen
HOST = 'http://weather.noaa.gov/'
FCST = '/pub/data/forecasts/state/'
URL = HOST + FCST + '/il/ilz013.txt'
print('opening ' + URL + ' ...\n')
DATA = urlopen(URL)
while True:
    LINE = DATA.readline().decode()
    if LINE == '':
        break
    L = LINE.split(' ')
    if 'FCST' in L:
        LINE = DATA.readline().decode()
        print(LINE + DATA.readline().decode())
    if 'CHICAGO' in L:
        LINE = LINE + DATA.readline().decode()
        LINE = LINE + DATA.readline().decode()
        print(LINE + DATA.readline().decode())
```

networking and the internet
markup languages

1. Internet Basics
   - evolution of the web
   - IP addresses and URLs
   - client/server and HTTP
2. Markup Languages
   - HTML, XML, MathML
   - MathML generated by Maple
3. Retrieving Data
   - the weather forecast
4. CGI Programming
   - showing current date and time in browser
5. Summary + Assignments

Python and CGI

Common Gateway Interface

A sequel to GUIs: run programs through a browser.
Web interfaces are also *event driven*.

Python as scripting language for the web:
transform XML into XHTML pages.

The *Common Gateway Interface* (CGI) describes a set of protocols through which CGI programs interact with web servers and browsers.

CGI is common because it is not specific to any operating system or to any programming language or to any web server software.

We will use Python to generate dynamic web pages.

Showing Current Time

Three steps to run Python scripts on the web:

1. Open browser at `http://localhost`
   What should be visible is the message `If you can see this, it means that the installation of the Apache web server software on this system was successful.`
2. On MacOS X: `/Library/Webserver/CGI-Executables` contains Python scripts.
   On Linux: `/var/www/cgi-bin` is the directory for scripts.
3. Write the Python script to show current time.
   From the module `time` we use the functions `time()` and `ctime()`.

showtime.py

```
#!/usr/bin/python
# L-18 MCS 260 Mon 23 Feb 2015 : showtime.py
"""
Illustration of writing the current time on a web page.
On Unix, save this script in /var/www/cgi-bin
and execute pointing the browser to
dezon.math.uic.edu/cgi-bin/showtime.py
"""
print "Content-Type: text/html\n\n"
import time

def print_header(title):
    """
    prints the title of the web page
    """
    print """
<html>
<head><title>%s</title></head>
<body>""" % title

print_header("current date and time")
print time.ctime(time.time())
print "</body></html>"
```

some comments

Two points:

1. The first line is the location of the Python interpreter.
2. The first `print` indicates that html is written, opposed to plain text code.

Summary + Assignments

Background: §4.2,3 in *Computer Science, an overview*.
More of chapter 5 of *Python Programming in Context*.

Assignments:

1. Make your own web page.
   Consult the ACCC help pages on web publishing.
2. Use Maple to generate MathML to display monomials, e.g. $8x^3$, and general polynomials.
3. Add a legend to the `forecast.py` script, using a dictionary to spell out the abbreviations `PTCLDY`, `MOCLDY`, etc.
4. Design a GUI dedicated to browsing weather forecasts. Which widgets will you use? What is the layout?
5. Write Python code for the GUI of the previous exercise.