CSC309 Winter 2016

Programming on the Web

Lecture 1

Thanks to Arnold Rosenbloom, Eyal de Lara, Karen Reid, Manos Papagelis and Mashiyat Ahmed Shah for their previous course materials.

Instructor

Larry Zhang

Office: DH-3076

Email: ylzhang@cs.toronto.edu

Today’s agenda

➔ **Why** take CSC309

➔ **What** is in CSC309

➔ **How** to do well in CSC309

➔ Start learning HTTP, HTML

Why study web programming?

The I**nternet and the Web** has been the greatest invention of mankind, it totally changed how people acquire and share knowledge, or how people live their lives.

What make the Web great are the **web applications** created by people who learned web programming**.**

The power of the Web is still **far from** being fully exploited.

Learn some web programming & be part of this is simply **exciting**.

You'll learned a variety of interesting concepts from web programming.

It may inspire you to change the way software is developed.

Also, it gives you the tools to become fabulously wealthy.

What is in CSC309?

Outline of the course

- ➔ The fundamentals
  - ◆ HTTP: web servers, browsers, how they communicate
- ➔ Front-end (client side) programming
  - ◆ HTML, CSS, Javascript
- ➔ Back-end (server side) programming
  - ◆ PHP, database, MVC frameworks, NodeJS, WebSocket ....
- ➔ Deployment
  - ◆ scalability, performance, security ...

Teaching goals

1. Can’t teach everything in detail
2. Teach the important fundamental concepts that may be ignored when you self-learn the stuff.
3. Teach the tools that you need to do the assignments.
4. Expose you to various technologies so that you will know where to go if you want to learn more and deeper.

Speaking of the assignments ...

You will develop three web applications in the course

- ➔ A simple static web page with style
    - ◆ pure front-end
- ➔ A real web service where people can register and do cool stuff
    - ◆ front-end + back-end
- ➔ An online game
    - ◆ advanced front-end + back-end

These three assignments will be increasingly more complex, more creative and cooler.

- ➔ Something real so you can potentially put it online for the world.
- ➔ Something you will feel proud to talk about in your future interviews.

something you'll be able to do

https://lightsaber.withgoogle.com/

How to do well in CSC309?

First and foremost...

Be Interested.

Course web page

http://www.cs.toronto.edu/~ylzhang/csc309w16/

All course related info will be available here.

Textbook

There is no textbook.

There will be slides and pointers to various online materials.

Will be posted on the course web page.

Marking Scheme

A1:	20%
A2:	20%
A3:	27%
Final exam:	33%
Total:	100%

Assignments

A1 and A2: work in groups of 2, work individually is OK but not recommended.

A3: work in groups of 2 - 3, don’t work individually on A3

You may starting looking for partners now.

Tutorials

Wednesday 5-6pm and 6-7pm

Work on exercises that are useful for the assignments, with the help of the TA.

Some tutorials will be dedicated for assignment problem solving.

Tutorials are as important as lectures.

Final Exam

There is no midterm.

Final Exam will test on

➔ what we learned in the lectures

➔ what we learned in the tutorials

➔ what we learned from the assignments

Discussion Board (Piazza)

https://piazza.com/utoronto.ca/winter2016/csc309h5

All course announcements will be sent through Piazza.

Daily reading is required.

Good way to get your questions answered, also benefits the whole class.

Also a good way to find partners for assignments, use the “Search for Teammates” post.

Office hours

Tuesday 5pm - 6:30pm, priority for CSC309 students

Monday 5pm - 6:30pm, priority for CSC258 students

Friday 5pm - 6:30pm, priority for CSC263 students

or by appointment

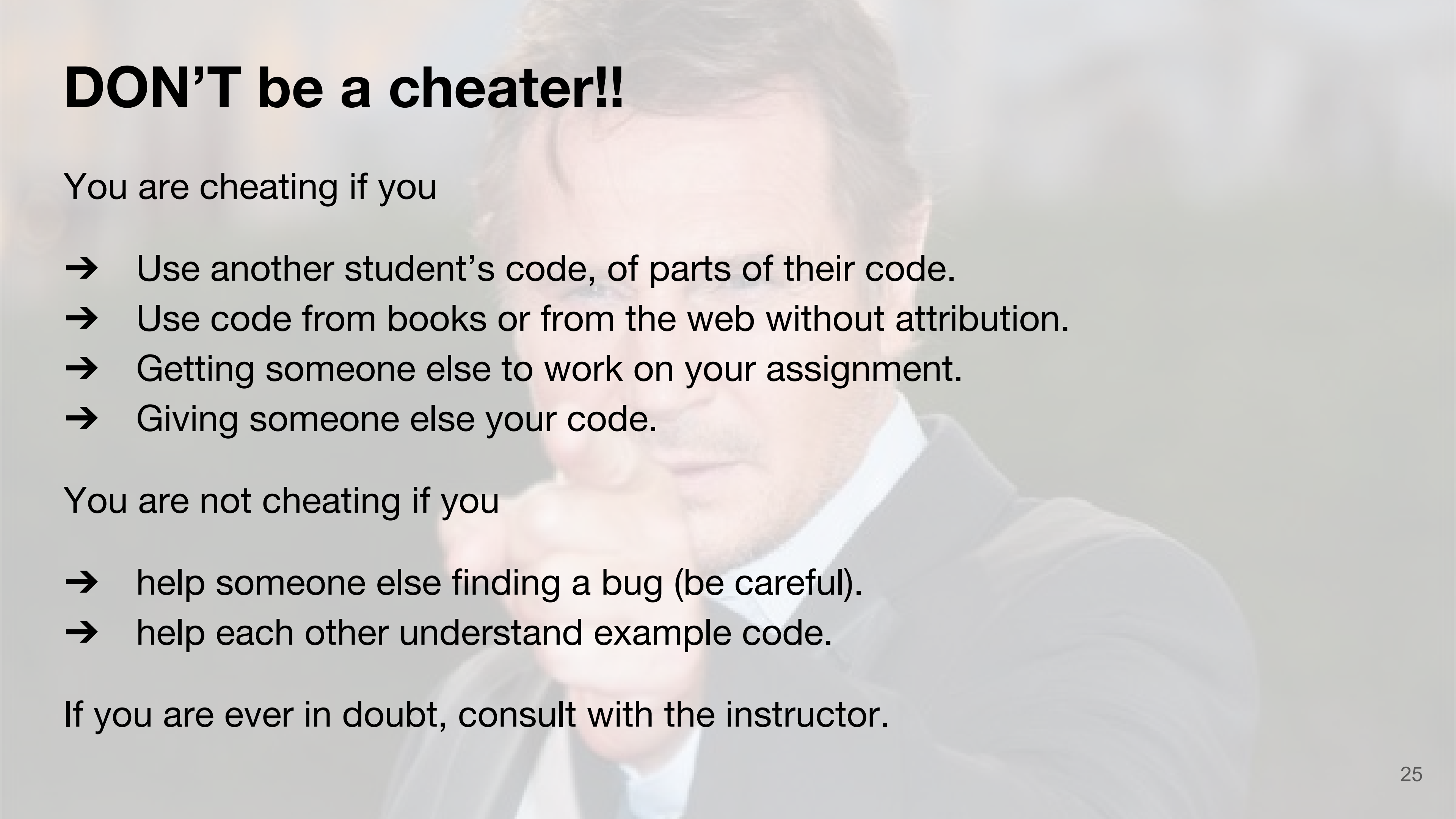
Going to professor’s office hours is a good habit to foster in college. Feel free to just drop by to say hi or tell me your name.

MLH
Algorithms
DCS
GODIVA

So, how to do well in CSC309?

This is course is highly practical, i.e., you learn most of the things by getting your hands dirty in the assignments.

- ➔ Develop something that you think is cool, and be excited about it.
- ➔ Pay attention in lectures and go to tutorials.
- ➔ Start early with working on assignments.
- ➔ Communicate well with your teammates.
- ➔ Develop your time management skills, and meet the deadlines.
- ➔ Ask for help when get stuck (Piazza, office hours, tutorials, ….)
- ➔ Do well in the final exam



DON’T be a cheater!!

You are cheating if you

- ➔ Use another student’s code, of parts of their code.
- ➔ Use code from books or from the web without attribution.
- ➔ Getting someone else to work on your assignment.
- ➔ Giving someone else your code.

You are not cheating if you

- ➔ help someone else finding a bug (be careful).
- ➔ help each other understand example code.

If you are ever in doubt, consult with the instructor.

Weekly feedback form

http://goo.gl/forms/pJnhjEj3AM

Have your issues addressed on a weekly basis, rather than termly basis.

Very important!

Let’s start learning!

The **Internet** and the **Web**

Are they the same thing?

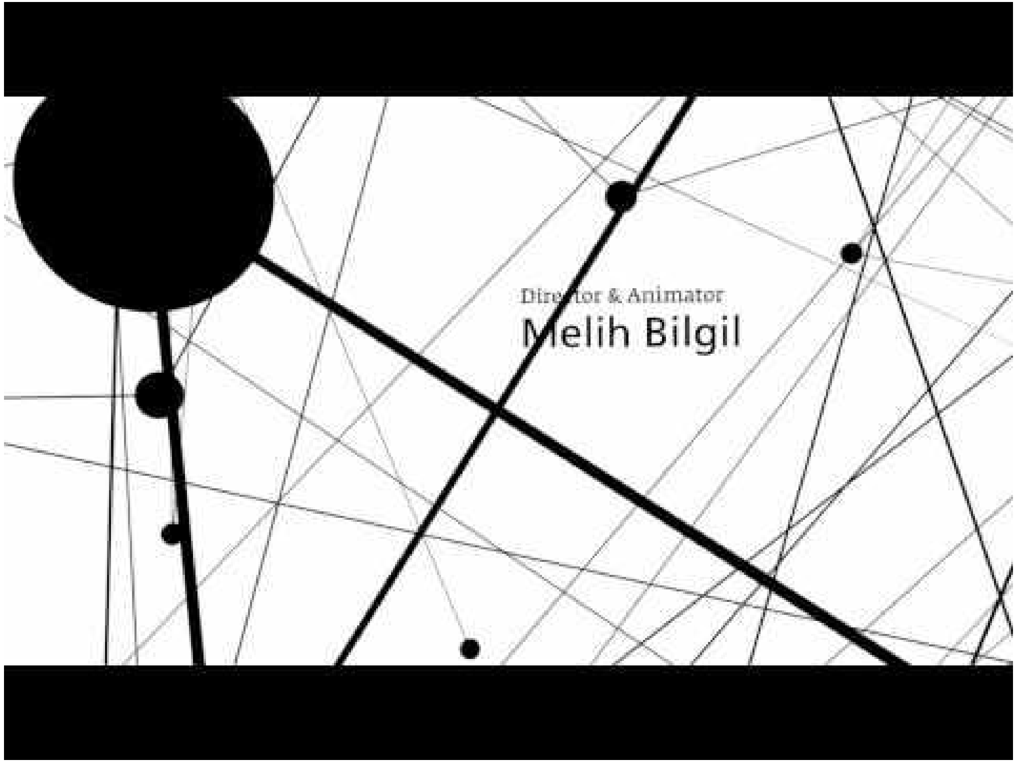
Analogy: people writing letters to each other

- The Web is about how to write a letter, how to read a letter, how submit the letter to a postal office.
    - more about things like the format of the document
- The Internet is about how to **deliver** the letter from one location to another.
    - which streets to travel, how to avoid traffic jam, etc
- The Internet is the lower level **communication** layer, the Web is an **application** on top of the Internet.

A brief history of the Internet

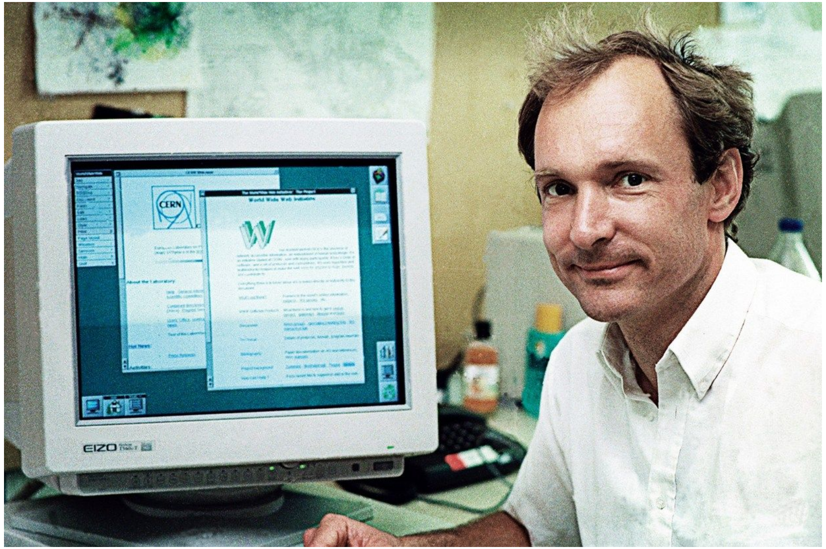
- ➔ 1960s, ARPANET, a network for the US Department of Defense
- ➔ 1960s, Packet switching
- ➔ 1971, Email is born
- ➔ 1974, TCP/IP invented, a decentralized, scalable and reliable communication protocol
- ➔ 1979, USENET, a worldwide distributed discussion system
- ➔ By 1987, the Internet include nearly 30,000 hosts
- ➔ **The Web was not born yet at this point.**

Video: History of Internet



History of the Web

- ➔ 1989, created by Tim Berners-Lee
- ➔ Popular web browsers released
  - ◆ Netscape: 1994
  - ◆ Internet Explorer: 1995
- ➔ 1995, Amazon.com
- ➔ 1996, Google
- ➔ 2001, Wikipedia
- ➔ 2003, MySpace
- ➔ 2004, Facebook
- ➔ …, then you had all grown up and knew what happened till now.



The Web is still evolving, what’s the future?

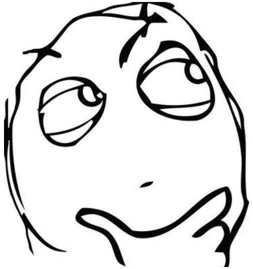
Tim Berners-Lee talks about “the next web”.

➔ https://www.ted.com/talks/tim_berners_lee_on_the_next_web?language=en

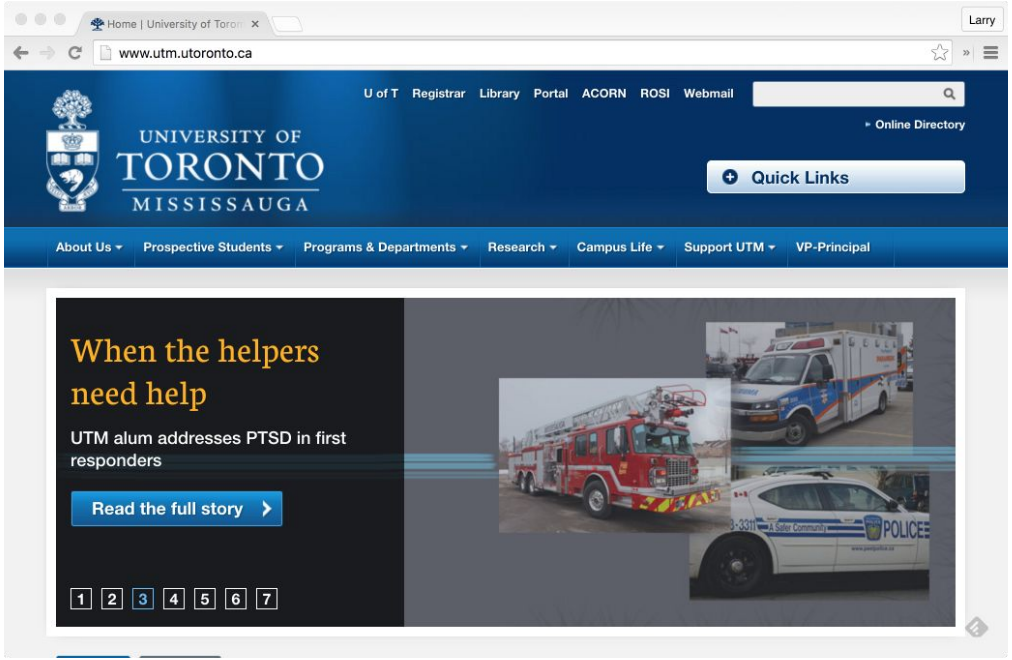
Some important organizations

- ➔ Internet Engineering Task Force (**IETF**): Internet protocol standards
- ➔ Internet Corporation for Assigned Names and Numbers (**ICANN**): decides top-level domain names
- ➔ World Wide Web Consortium (**W3C**): web standards

enough history and politics,
let’s go technical



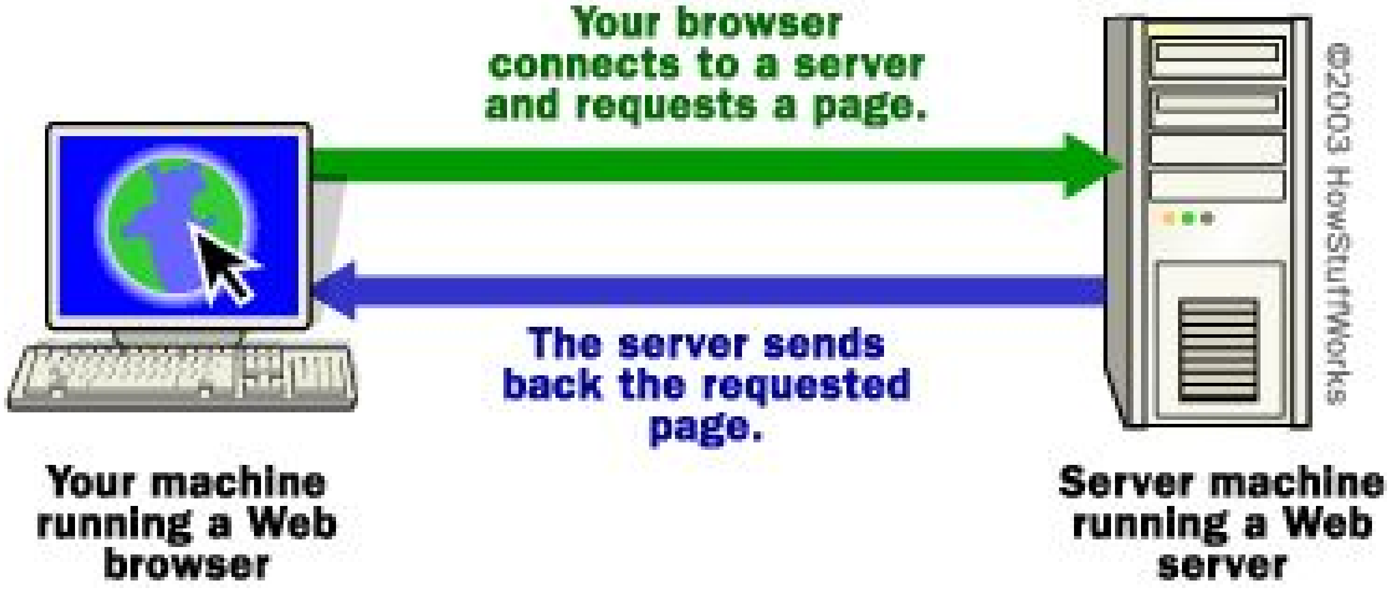
So I type “www.utm.utoronto.ca” in my browser, then a web page shows up. **What exactly happened there**?



When the helpers need help

UTM alum addresses PTSD in first responders

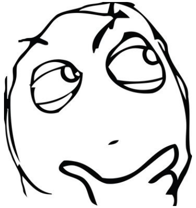
The vague story


Your browser connects to a server and requests a page.
The server sends back the requested page.
Your machine running a Web browser
Server machine running a Web server
©2003 HowStuffWorks

The vague story

- ➔ By the **URL** that’s typed in, we can somehow **locate the address** of the web server (somewhere on the Internet) which contains the **document** of the desired web page.
- ➔ After locating the web server, the browser as a **client** establish a **connection** to the web server and sends a **request** to the web server which basically says “**get me the homepage’s document!**”.
- ➔ The web server **handles** the request, and send the client (the browser) the desired document as a **response**, via the established connection.
- ➔ The browser receives the response and **displays** it in the browser (in a formatted way) so the user can see the UTM home page.
- ➔ The established connection between the browser and the server is **closed.**

Make it less vague



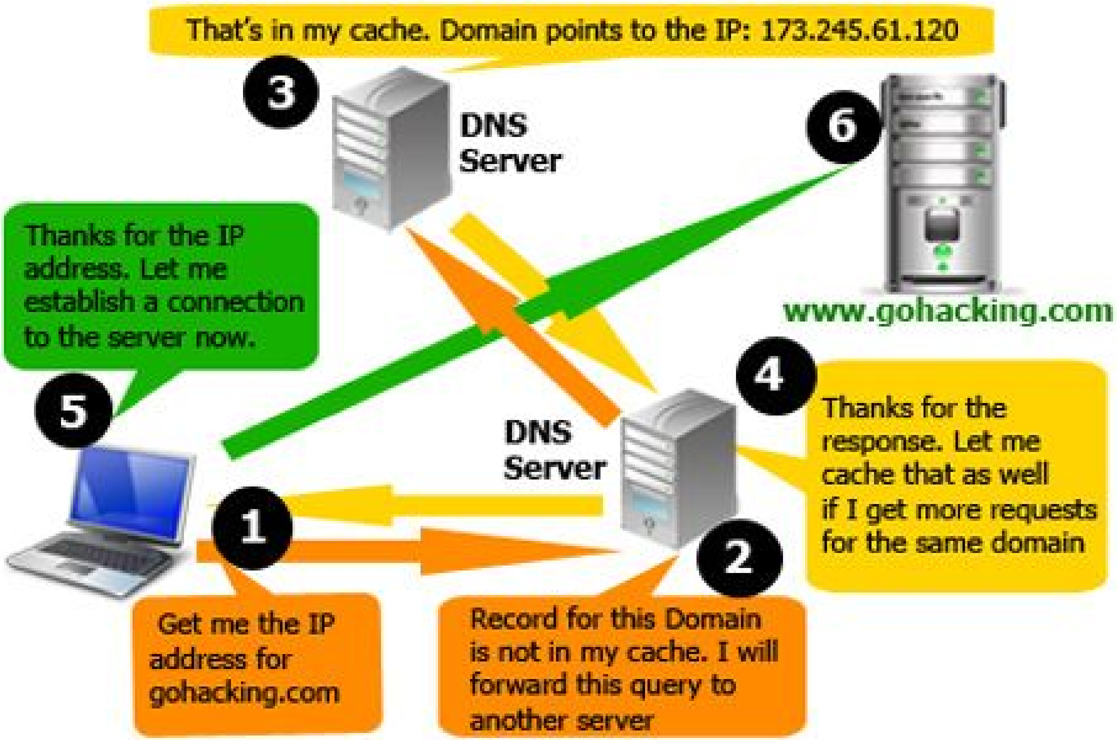
What do you mean “**address of the web server**”?

- ➔ The address is an **IP address**, like “142.150.1.50”
- ➔ IP stands for Internet Protocol
- ➔ Default port for web server is 80, so it’s actually 142.150.1.50:80

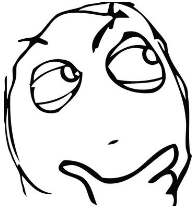
How do we get from “www.utm.utoronto.ca” to “142.150.1.50”

- ➔ Look it up by asking the DNS servers
- ➔ DNS stands for Domain Name Servers
- ➔ This process may involves multiple message passing between DNS servers, more details in CSC358.

How DNS works


That's in my cache. Domain points to the IP: 173.245.61.120
3
DNS Server
6
www.gohacking.com
Thanks for the IP address. Let me establish a connection to the server now.
5
DNS Server
4
Thanks for the response. Let me cache that as well if I get more requests for the same domain
1
2
Get me the IP address for gohacking.com
Record for this Domain is not in my cache. I will forward this query to another server

Make it less vague



OK, so we got the address of the web server, but we requested a **document**. There could be many documents on that server, which one to get?

Uniform Resource Locator (URL)

When we are accessing "www.utm.utoronto.ca", we are actually requesting "www.utm.utoronton.ca**/index.html**"

The "/index.html" part locate a document on the web server.

The web server sends the content of the document **index.html** as **response** to the client (browser), then the browser **displays** the content of index.html.

HyperText Transport Protocol (HTTP)

Developed by IETF and W3C.

It is basically a protocol of the request / response format between the browser and the web server.

HTTP **Request** (version 1.0 and 1.1)

`GET`: request to download a resource

`POST`: similar to GET but also send some data along in the request to the server to be processed.

`PUT`: request to upload a resource

`DELETE`: delete the specified resource

`HEAD`: like GET but only expect header in the response

HTTP **Response** (version 1.0 and 1.1)

- **Header**
  - error code
    - 200: OK
    - 301-303: page has moved (permanently or temporarily)
    - 403: access forbidden
    - 404: page not found
    - 500: internal server error
    - http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
  - Content Type: text/html, text/plain, text/gif, img/gif, img/jpeg/, video/quicktime, etc.
  - other info such as timestamp, content length, etc.
- **Content of the requested resource / document**
  - HTML document, image file content, etc.

demo

http://www.cs.toronto.edu/~ylzhang/csc309w16/files/lec01-http.txt

Note:

HTTP/1.0 allows one IP address to host only one website (domain name)

HTTP/1.1 allows one IP address to host multiple websites, therefore the request MUST include a “Host” field to specify which website is being requested.

HTTP/1.0 closes the connection immediately after response is received.

HTTP/1.1 allows connection to persist across multiple requests, the connection closes after a timeout.

Reading: HTTP Made Really Easy

https://www.jmarshall.com/easy/http/

Just so you know, HTTP version 2.0 is out, most major browsers added support by the end of 2015.

5.6% of top 10 million websites are using HTTP/2

https://en.wikipedia.org/wiki/HTTP/2

```
HTTP/1.1 200 OK
Date: Sun, 03 Jan 2016 20:21:24 GMT
Server: Apache/2.4.7 (Ubuntu)
Last-Modified: Sun, 03 Jan 2016 20:16:25 GMT
ETag: "3f-52873adc0d840"
Accept-Ranges: bytes
Content-Length: 63
Content-Type: text/html

<!DOCTYPE html>
<html>
    <body>
        Hello world!
    </body>
</html>
```

This is an **HTML** document

Hyper**T**ext **M**arkup **L**anguage

HTML

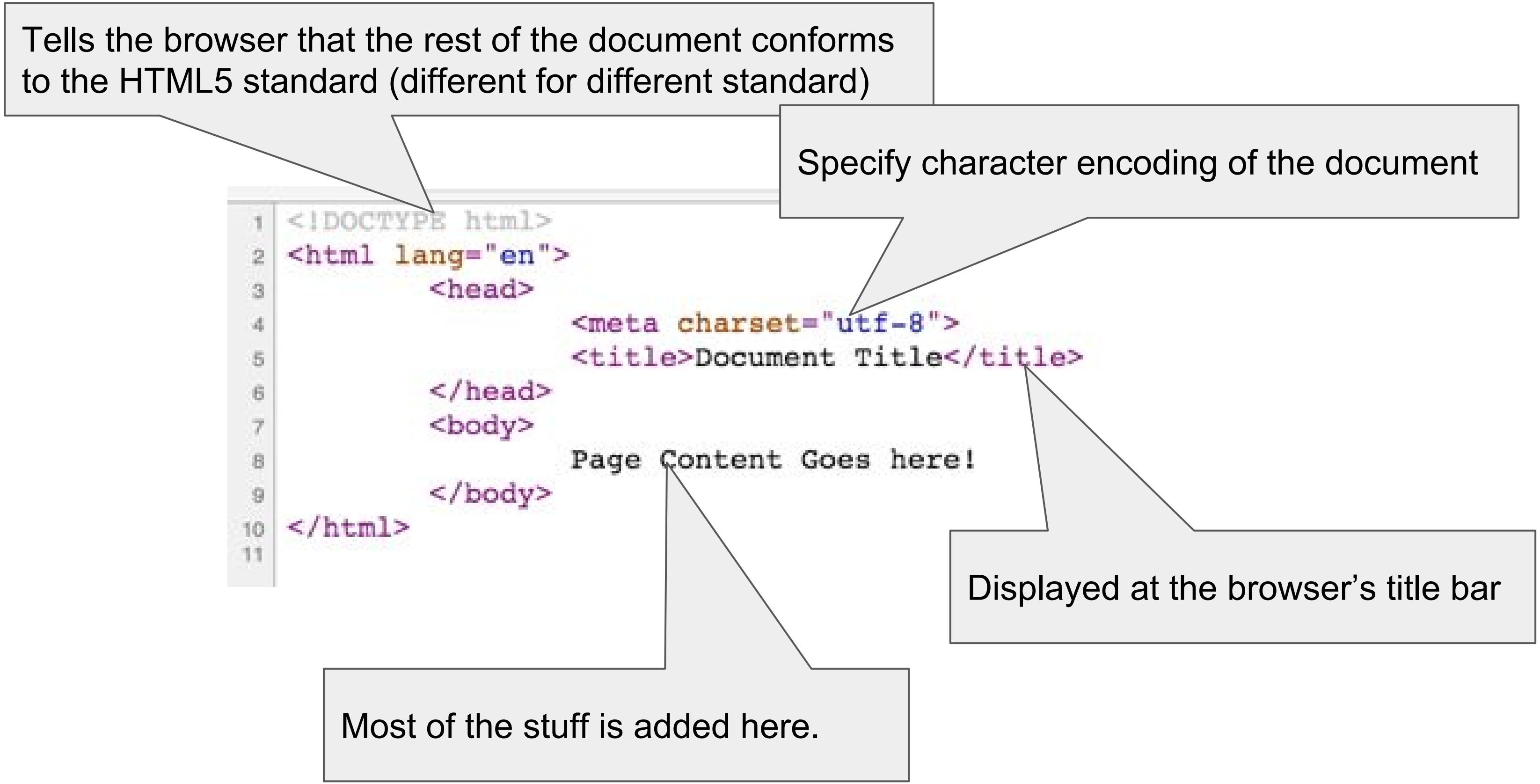
- It marks the the **content** and **structure** of information on a web page.
  - what content is in a block
  - which blocks are contained by a block
- Note: it does NOT mark the **appearance** of a web page.
  - appearance is defined by CSS, which we will learn later
- Basic structure <some tag> content </some tag>
  - e.g., `<p>This is a paragraph</p>`
  - `<div>nested: outer block <div>inner block</div> outer ends</div>`
- Unless otherwise specified, we are talking about **HTML5**.

What do you think about this code?

```
<div>Below is paragraph.<p>some text</div></p>
```

Invalid nesting!

Starter HTML file: hello.html



Tells the browser that the rest of the document conforms to the HTML5 standard (different for different standard)

Specify character encoding of the document

```html
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="utf-8">
                <title>Document Title</title>
        </head>
        <body>
                Page Content Goes here!
        </body>
</html>
```

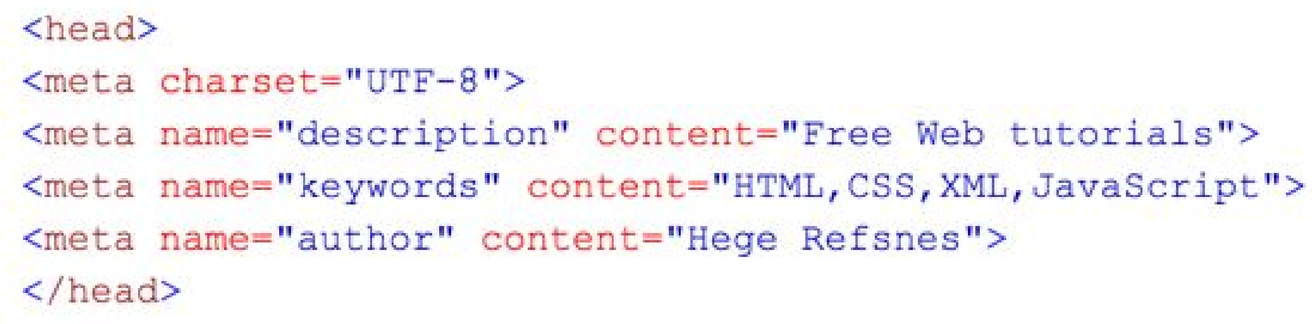
Displayed at the browser’s title bar

Most of the stuff is added here.

More about the <meta> tag

Data about the HTML document.

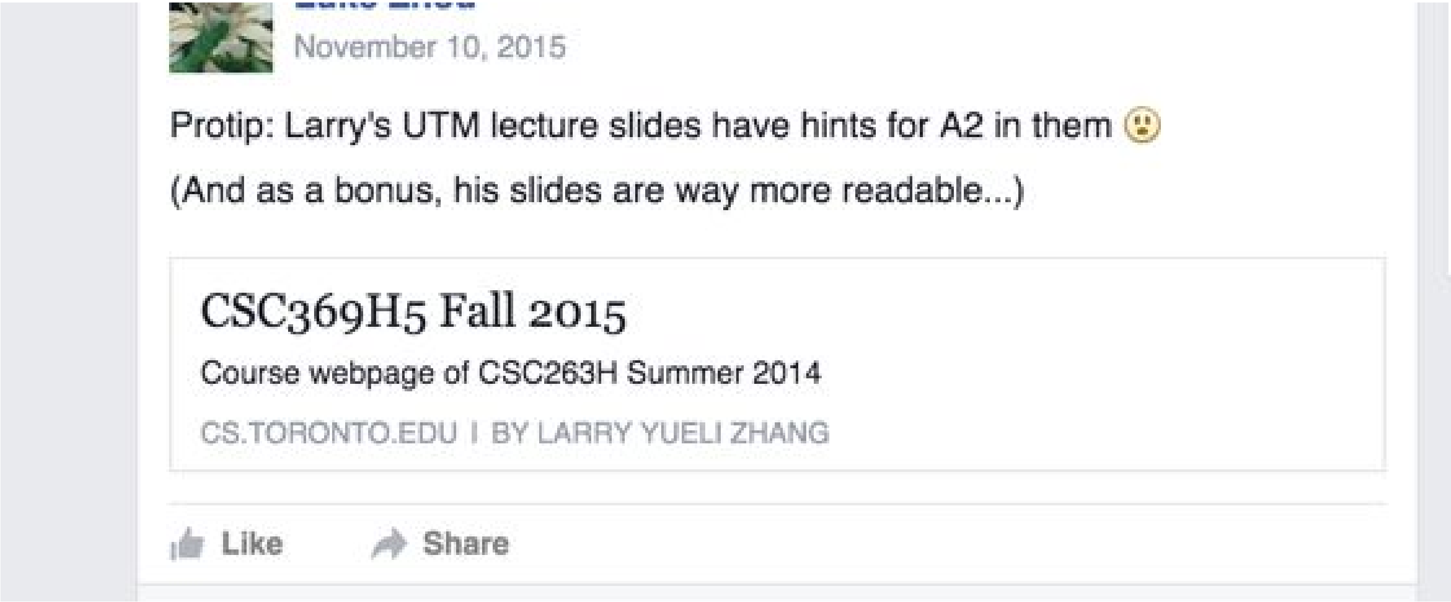
Note: **no** closing </meta>



```
<head>
<meta charset="UTF-8">
<meta name="description" content="Free Web tutorials">
<meta name="keywords" content="HTML,CSS,XML,JavaScript">
<meta name="author" content="Hege Refsnes">
</head>
```

Properly set <meta> increase the chance that search engines find you correctly.

Example of improperly set <meta>



November 10, 2015

Protip: Larry's UTM lecture slides have hints for A2 in them 😮

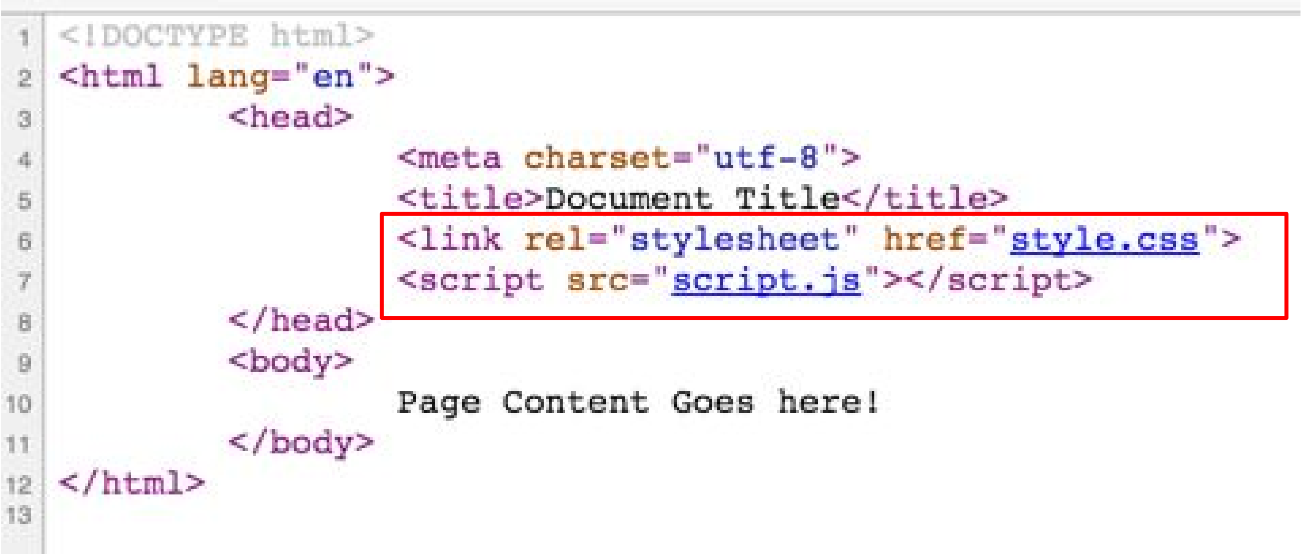
(And as a bonus, his slides are way more readable...)

CSC369H5 Fall 2015

Course webpage of CSC263H Summer 2014

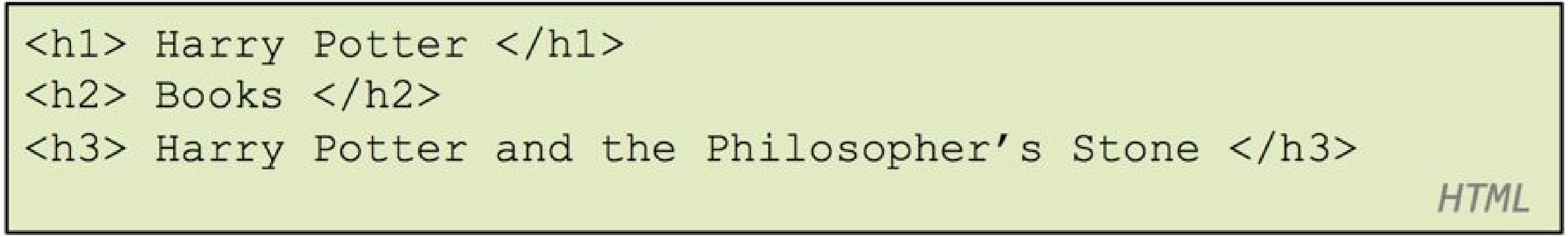
CS.TORONTO.EDU | BY LARRY YUELI ZHANG

With stylesheet and javascript

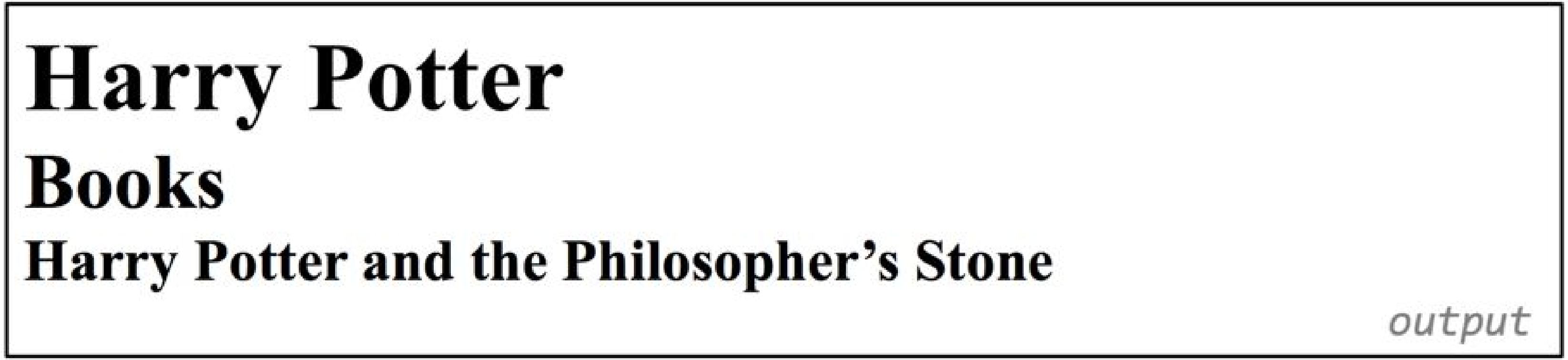


```
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="utf-8">
                <title>Document Title</title>
                <link rel="stylesheet" href="style.css">
                <script src="script.js"></script>
        </head>
        <body>
                Page Content Goes here!
        </body>
</html>
```

Example: headings



```html
<h1> Harry Potter </h1>
<h2> Books </h2>
<h3> Harry Potter and the Philosopher's Stone </h3>
```

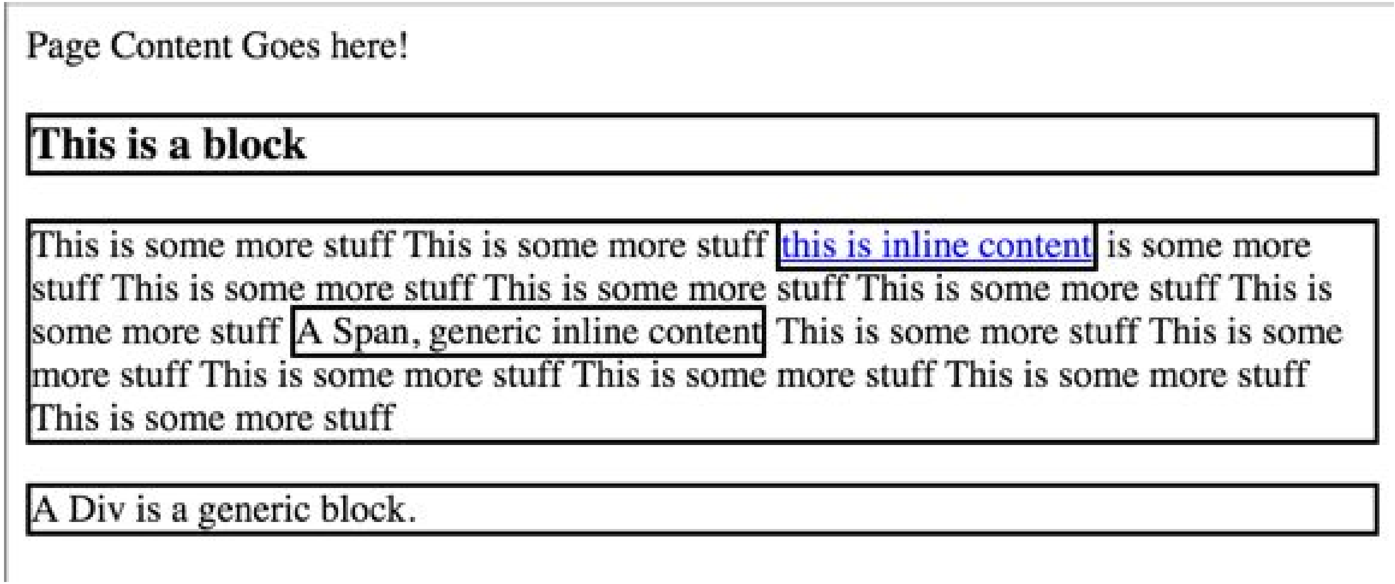


Harry Potter

Books

Harry Potter and the Philosopher's Stone

Example: **div** and **span**



Page Content Goes here!

This is a block

This is some more stuff This is some more stuff this is inline content is some more stuff This is some more stuff This is some more stuff This is some more stuff This is some more stuff A Span, generic inline content This is some more stuff This is some more stuff This is some more stuff This is some more stuff This is some more stuff This is some more stuff

A Div is a generic block.

demo link: http://www.cs.toronto.edu/~ylzhang/csc309w16/html/divspan.html

Tags with attributes

```
<tag attribute1=”value1” attribute2=”value2”>
```

For example: links

```
<a href=”hello.html”>Link Text</a>

<!-- relative path -->



<a href=”http://yahoo.ca” target=”_blank”>Link Text</a>

<!-- absolute path, open in new tab -->
```

Comments

```
<!-- content of comments -->
```

Some elements are open and closed by **one** tag

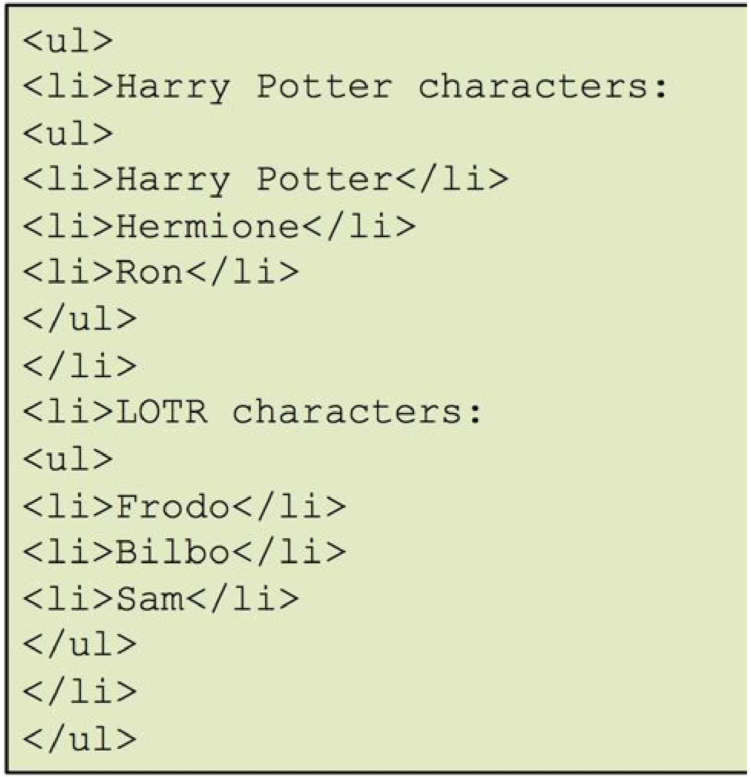
Image

```
<img src=”image.jpg” />
```

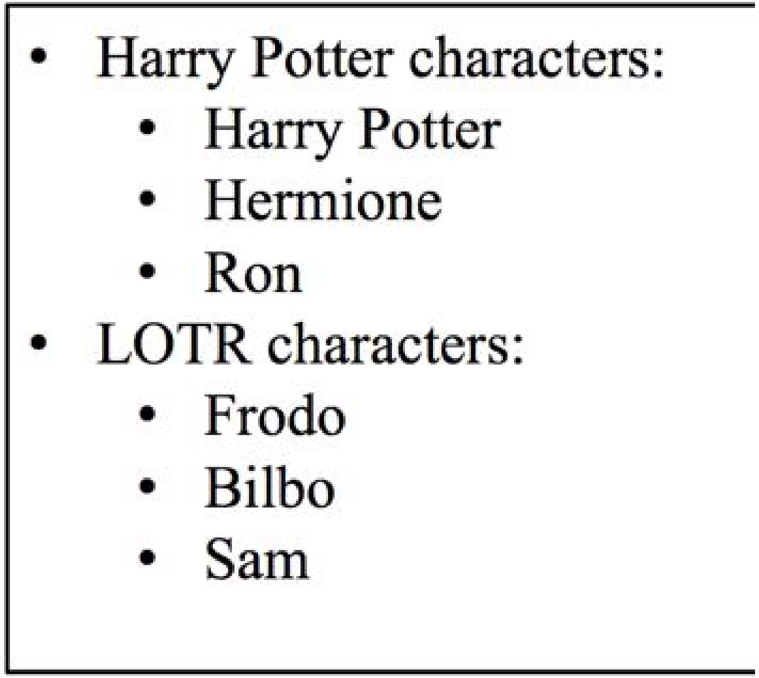
Line break

```
<br />
```

Lists: ul (unordered), ol (ordered)



```
<ul>
<li>Harry Potter characters:
<ul>
<li>Harry Potter</li>
<li>Hermione</li>
<li>Ron</li>
</ul>
</li>
<li>LOTR characters:
<ul>
<li>Frodo</li>
<li>Bilbo</li>
<li>Sam</li>
</ul>
</li>
</ul>
```



- Harry Potter characters:
  - Harry Potter
  - Hermione
  - Ron
- LOTR characters:
  - Frodo
  - Bilbo
  - Sam

No way we can cover all the elements in lecture

A comprehensive list of elements in HTML5

- ➔ https://developer.mozilla.org/en-US/docs/Web/HTML/Element
- ➔ This is required reading!

Each of them with **examples** that you can play with

- ➔ http://www.w3schools.com/tags/default.asp

Learn by playing with this!

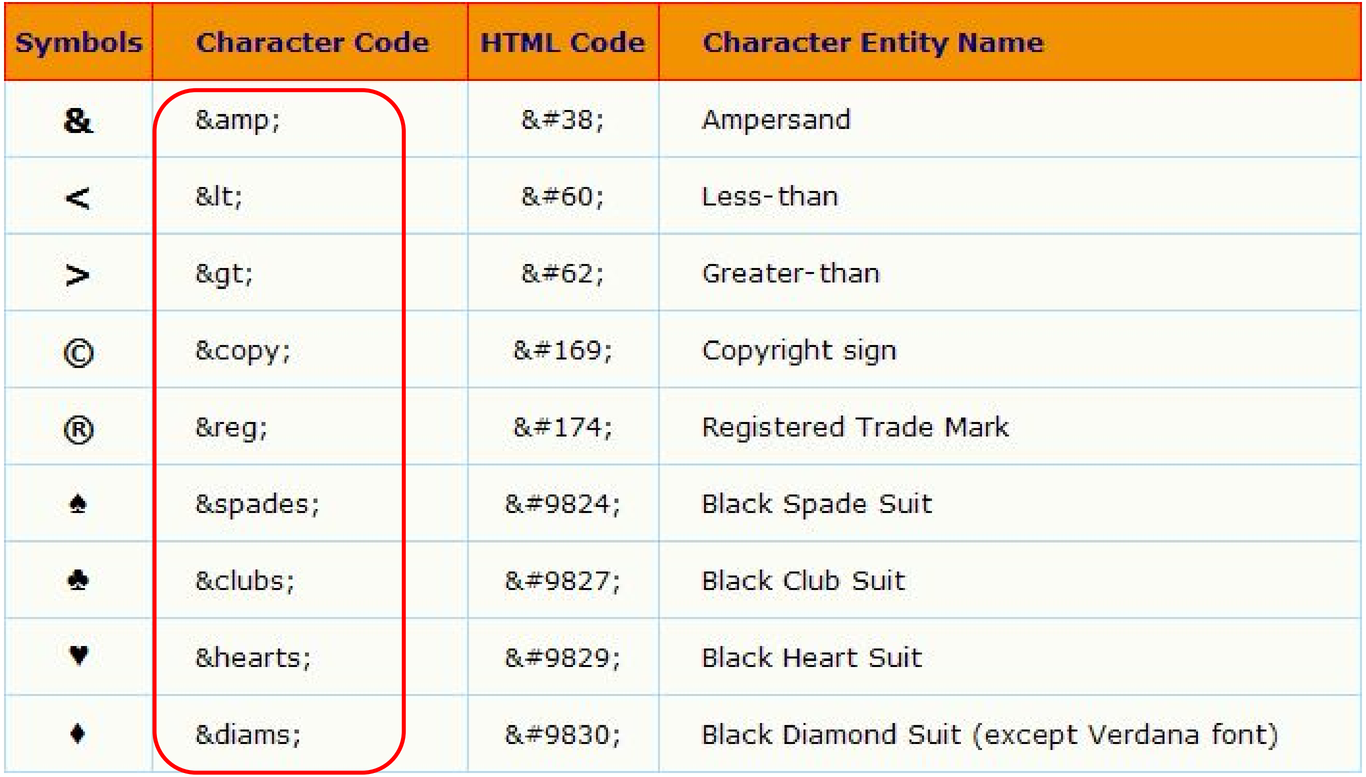
Your HTML code is supposed to conform to the W3C standard, so that you know you web page is going to be displayed consistently across platforms.

Here is a tool to verify whether the standard is satisfied:

https://validator.w3.org/

Very useful for assignments!

Special Characters in HTML



Symbols	Character Code	HTML Code	Character Entity Name
&	&amp;	&#38;	Ampersand
<	&lt;	&#60;	Less-than
>	&gt;	&#62;	Greater-than
©	&copy;	&#169;	Copyright sign
®	&reg;	&#174;	Registered Trade Mark
♠	&spades;	&#9824;	Black Spade Suit
♣	&clubs;	&#9827;	Black Club Suit
♥	&hearts;	&#9829;	Black Heart Suit
♦	&diams;	&#9830;	Black Diamond Suit (except Verdana font)

Some cool stuff added by HTML5

Audio and Video

➔ before HTML5, you need to embed a music player or video player

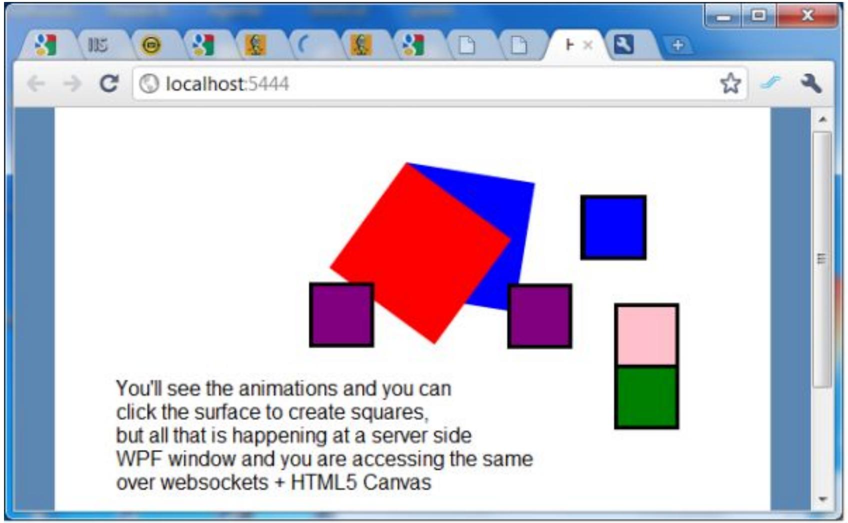
➔ with HTML5 you just do

```
<video>

      <source src=”cat.mp4”></source>

</video>
```

Some cool stuff added by HTML5

Canvas

- ➔ You can put a <canvas></canvas> on your web page and directly draw on with Javascript.



You'll see the animations and you can
click the surface to create squares,
but all that is happening at a server side
WPF window and you are accessing the same
over websockets + HTML5 Canvas

Some cool stuff add by HTML5

Other things

- ➔ Drag and Drop
- ➔ Geolocation
- ➔ WebSocket
- ➔ Storage

More DEMOS: http://html5demos.com/

A good HTML5 mini-book

http://diveintohtml5.info/

Today we learned

- ➔ HTTP, the protocol of the Web
- ➔ Some HTML

Next week

- ➔ More HTML
- ➔ CSS
- ➔ Javascript