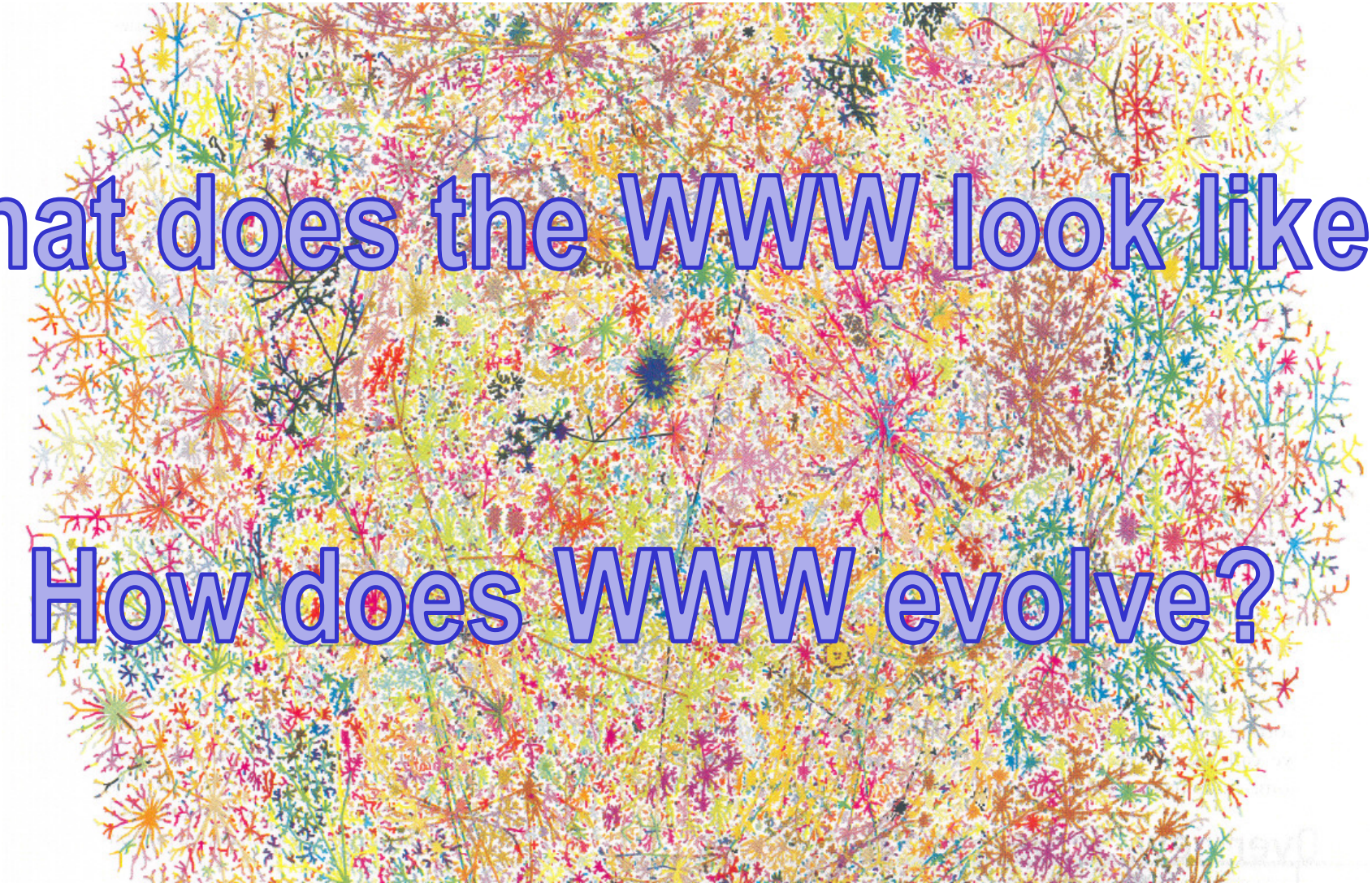
COMP721 Web Development

AUT UNIVERSITY

Week 1: Introduction

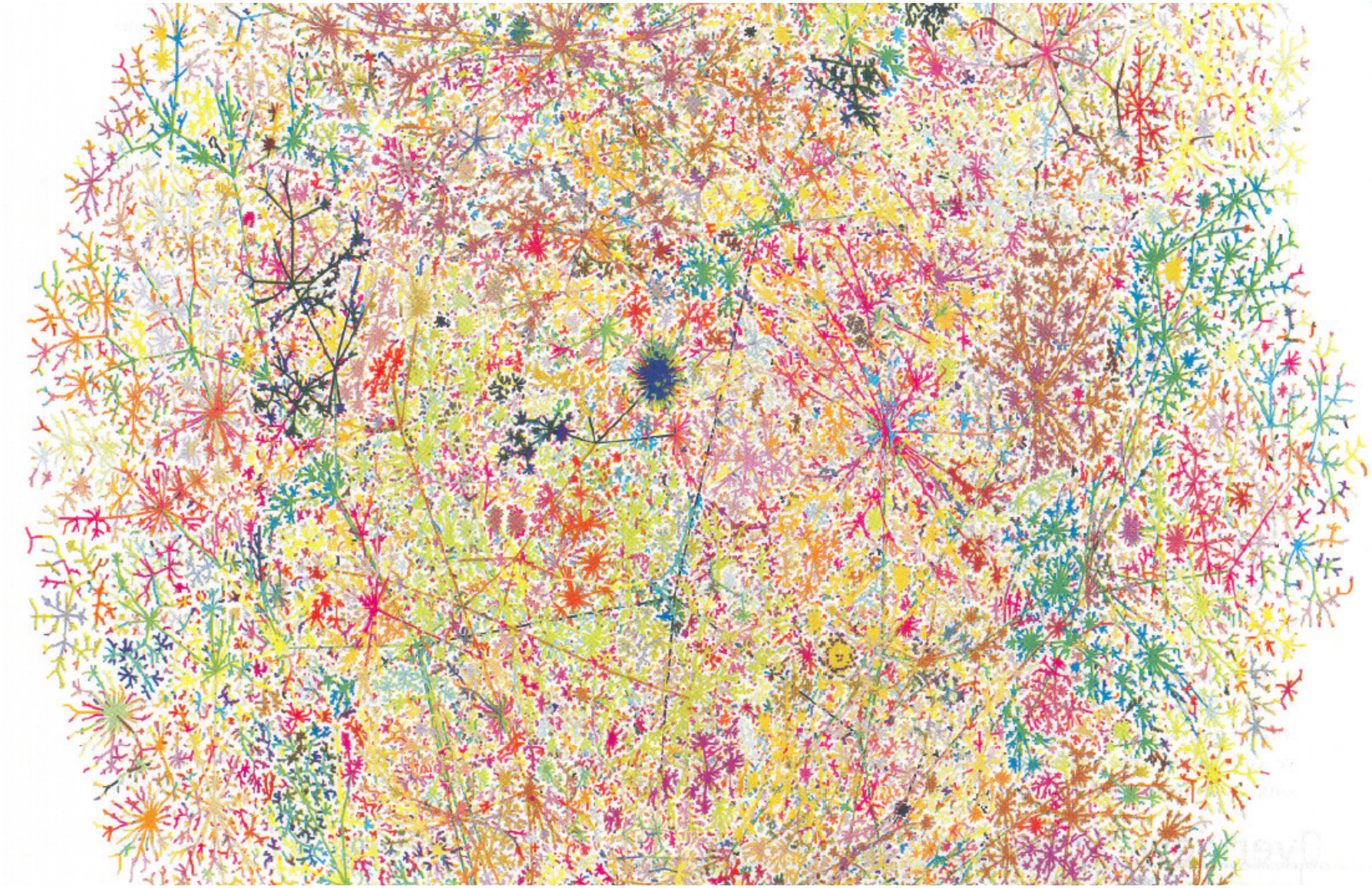


What does the WWW look like?

How does WWW evolve?

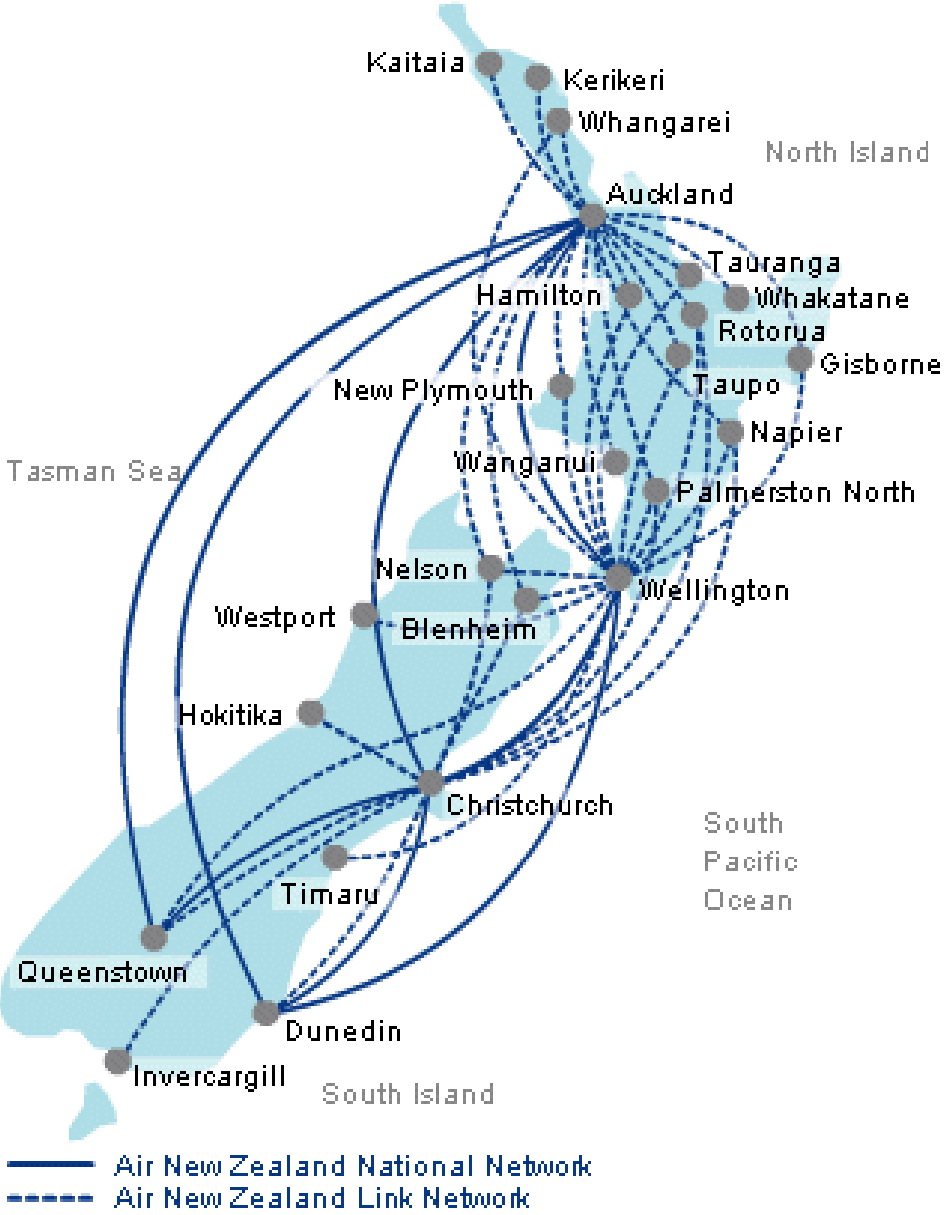
What does the WWW look like?

- **The topology/structure of WWW**
  - Star-shaped
  - Grid-shaped
  - Circle-shaped
  - Tree-shaped
  - A complex one contains all the above



Key property of WWW: a small number of big nodes (hubs) connect to tons of small nodes

Can you find the hubs?



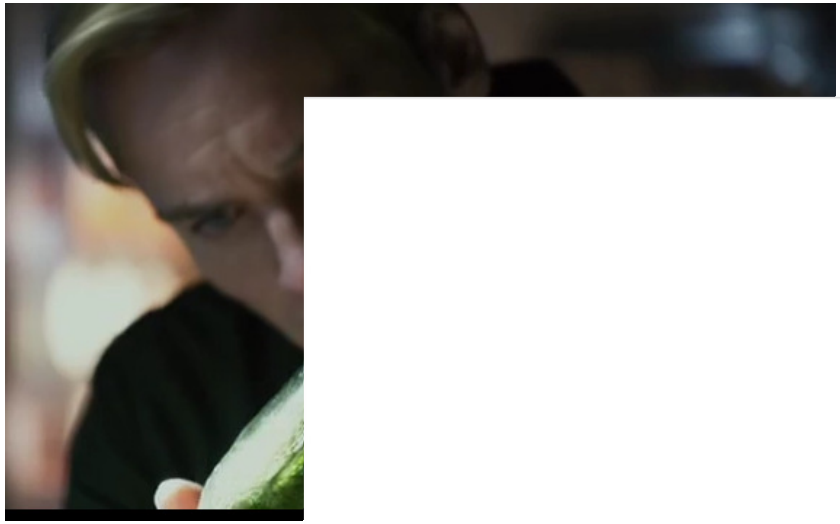
How does WWW evolve?

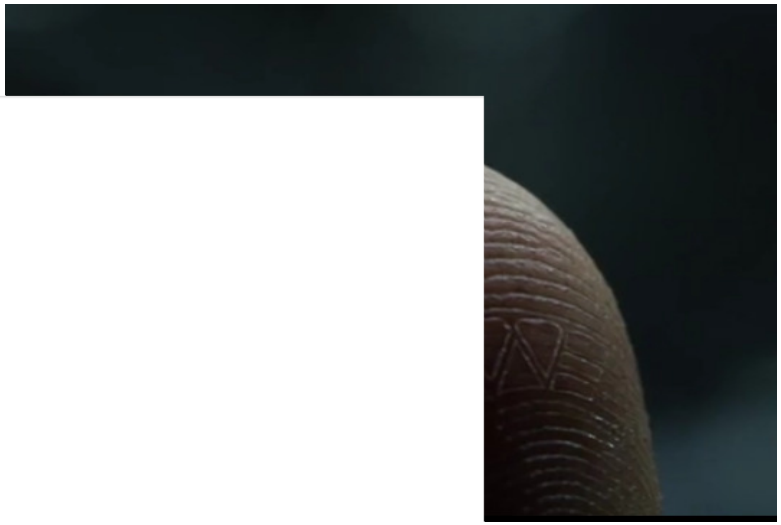
- Invented by Sir Tim Berners-Lee at European Organization for Nuclear Research (CERN) in 1989

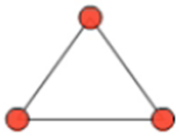
How does WWW evolves?

"Big things have small beginnings"

David (Android in Prometheus)







Agenda

- *About your Lecturer*
- *Web applications: architectures & modern features*
- *Course schedule and assessment*
- *Internet and WWW Definition*
- *HTTP & HTML Essentials*

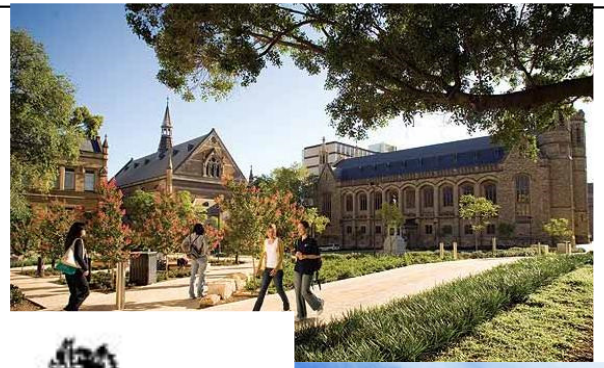
Teaching team

Dr Jian Yu

- Associate Professor
- Main teaching areas: web technologies, web and services computing, operating systems
- Research areas: Internet and Web computing, Machine learning, Recommender systems, complex networks
  - Finalist: AUT Vice Chancellor's Research Excellence Award
  - Room: WT703B
  - Email: jian.yu@aut.ac.nz
  - Phone: 6880

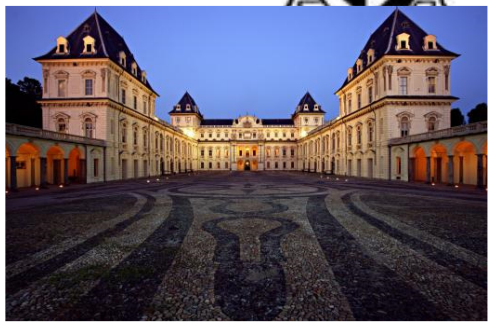
teaching/research experience

- Australia
  - University of Adelaide



  - Swinburne University of Technology

- Italy
  - Politecnico di Torino



  - Fabbrica Italiana Automobili Torino

- China
  - Peking University


INGENIO·ET·LABORE


FIAT

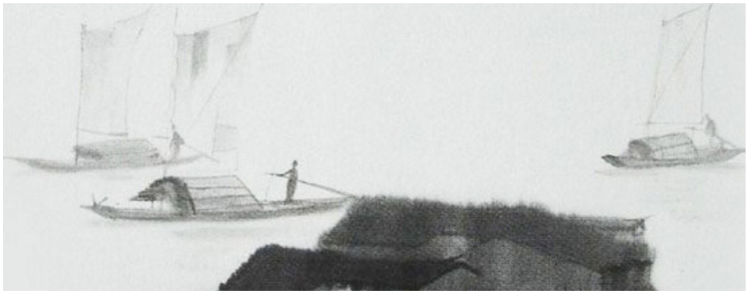

S F

Teaching team

- Mr Lei Zhou (Dia)
  - lei.zhou@autuni.ac.nz
- Mr Alan Zhang
  - yuqi.zhang@autuni.ac.nz
- Ms Nancy Wang
  - nancy.wang@aut.ac.nz

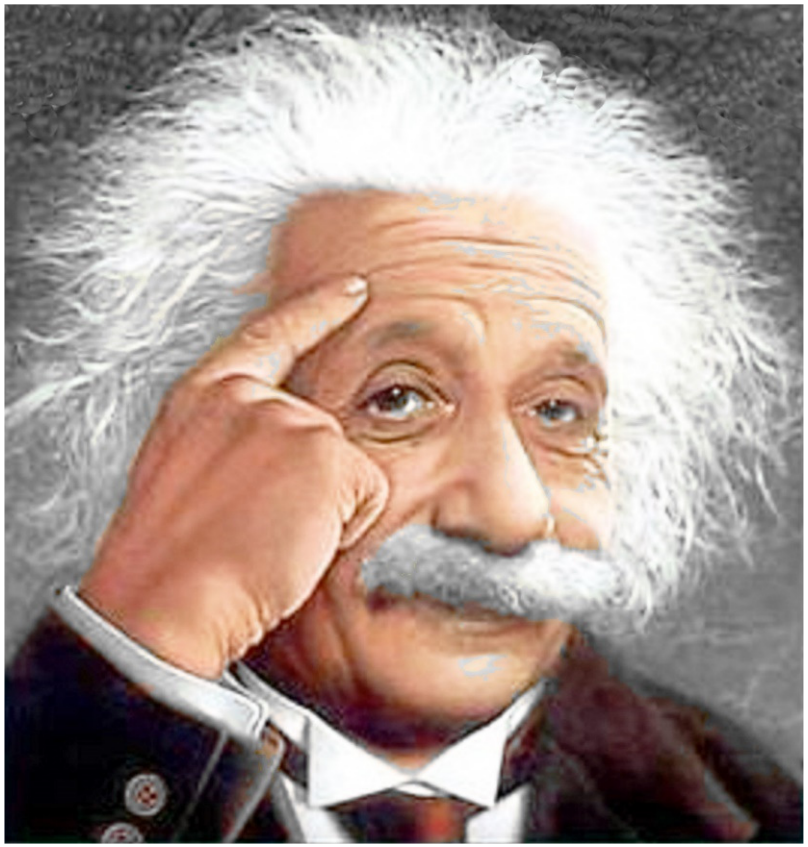
I am your resource

- education is not just about passing grades
  - Independent critical thinking
  - Problem solving skills
  - Life-long learning skills
- benefits to your career
  - letters of reference
  - contact for advice



An important message:

I am always here to help!



Education is not the learning of facts, it's rather the training of the mind to think. - Albert Einstein

Past teaching comments

- *“has strong knowledge of the subject and is very dedicated in making sure the students understood the subject.”*
- *“Providing great insight on this subject area and improving my knowledge”*
- *Was very energetic and explained the lectures quite good that made it clear and easy for me to understand”*

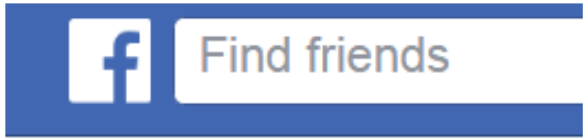
Some exciting web applications

http://www.aut.ac.nz/

https://maps.google.co.nz/

https://www.facebook.com/, Instagram

very interactive



Traditional Web Application architecture

A traditional 3-Tier Architecture of Web applications

Client 1

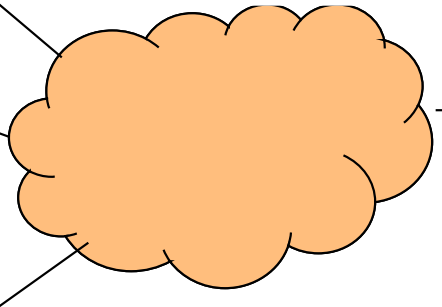
Client 2

...

Client n

Browser (Chrome, Firefox, Edge/IE…)
HTML + CSS + Javascript

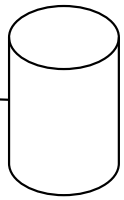
World Wide Web



Server

PHP engine
Java
ASP.NET engine
JSP engine
Ruby, Python…
Node.js

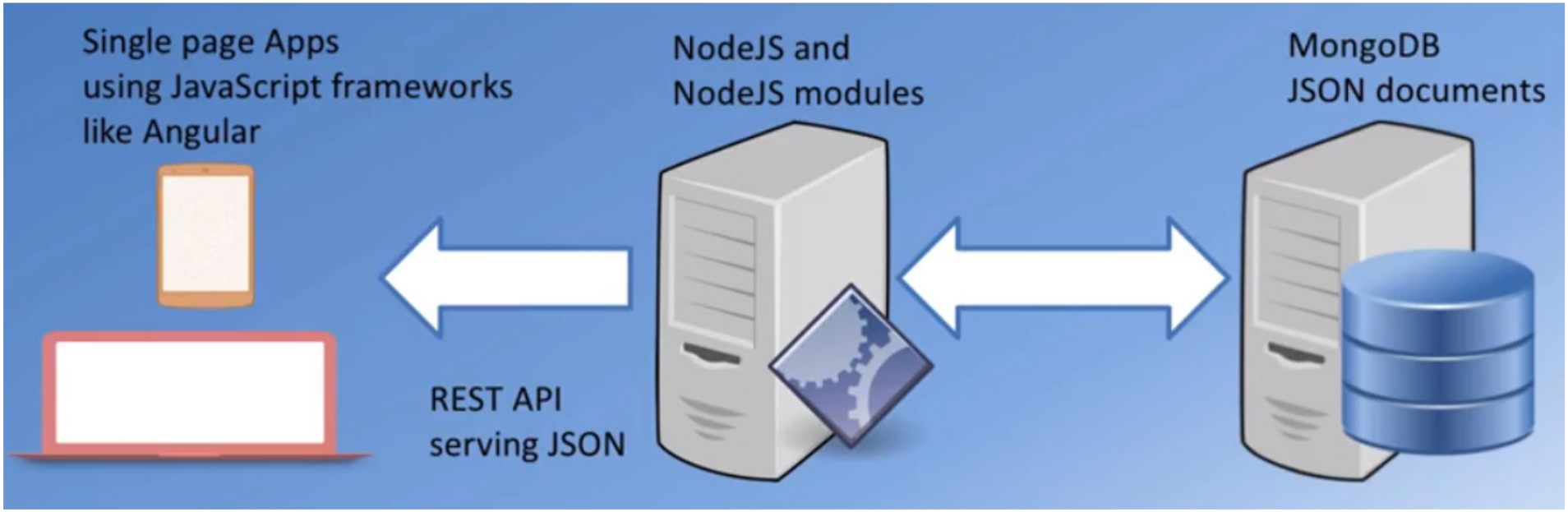
DBMS



MySQL
Oracle
NoSQL:
MongoDB
…

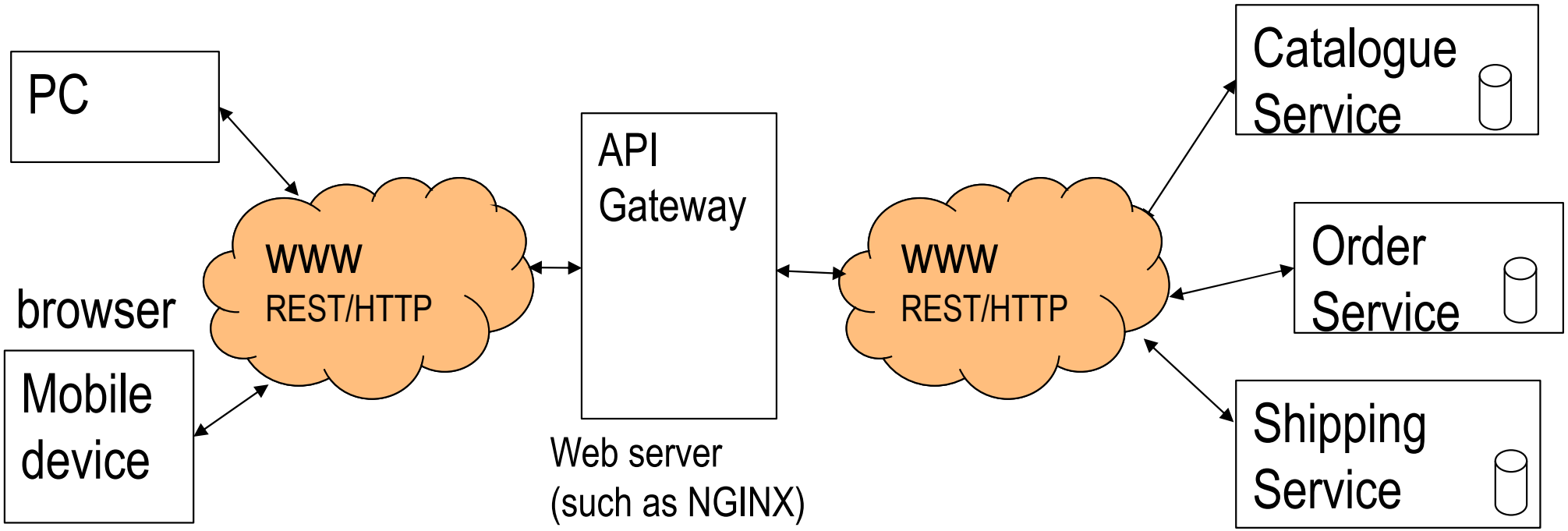
Recent trend 1: JavaScript-based frameworks: JavaScript everywhere

- Full stack JavaScript Development



No-SQL

Recent trend 2: micro-service architecture



REST: Representational State Transfer

Every micro-service is self-contained (e.g., has its own DB) and accessible over the Web

Web applications vs. Mobile apps (native client)

	Web Applications	Mobile Apps
Ubiquity (available any where, anytime)		
Cross platform support		
Maintenance		
Device level access		
User Interface		
Dev Language		

Course Schedule (also on Canvas)

Week	Topic(s)	Assessment
1	Introduction to Web development	
2	CSS, Bootstrap, and PHP Language (Part 1)	
3	HTML5 Forms and PHP Language (Part 2)	Assignment 1 - released on Blackboard
4	PHP (part 3), Database basics, MySQL DB, and MongoDB	
5	Manipulating MySQL databases with PHP	
6	Managing state information	Assignment 1 due
7	Introduction to Ajax	- Online Quiz One - Assignment 2 - released on Blackboard
8	Client-Side Processing – JavaScript, JSON and DOM	
9	Ajax and XML with shopping cart example	
10	Angular Object-Oriented Framework	
11	Web Services, Web APIs	Assignment 2 due
12	Ajax Frameworks (React, jQuery) & Review	Online Quiz Two

COMP721 WebDev Vision

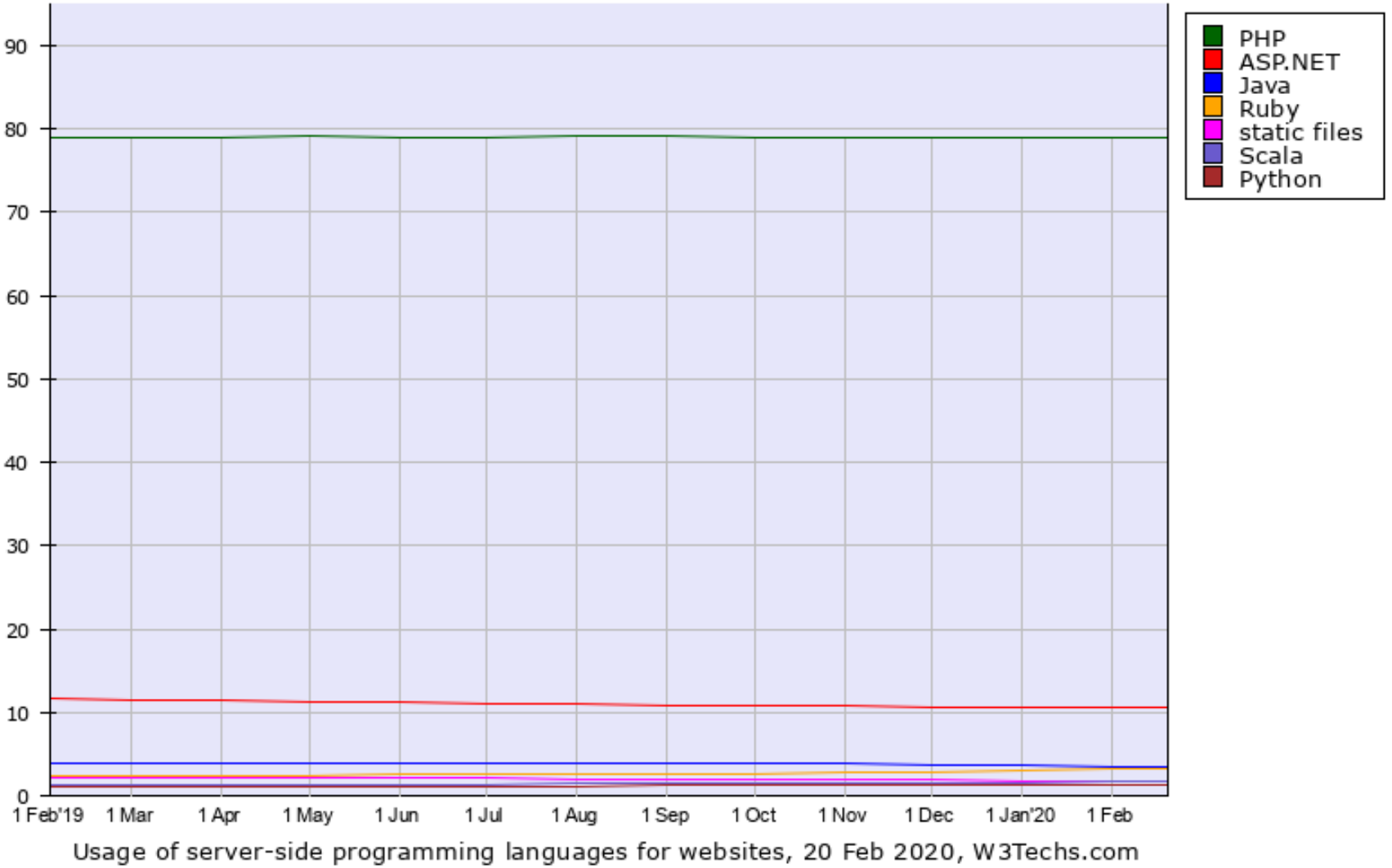
- Footing on the most popular technologies, while introducing the latest trend
  - ☐ historical trends in the usage of server-side languages
  - ☐ **PHP** is the most popular / **dominating** server-side language

Historical trends in the usage statistics of server-side programming languages for websites

This report shows the historical trends in the usage of server-side programming languages since February 2019.

	2019 1 Feb	2019 1 Mar	2019 1 Apr	2019 1 May	2019 1 Jun	2019 1 Jul	2019 1 Aug	2019 1 Sep	2019 1 Oct	2019 1 Nov	2019 1 Dec	2020 1 Jan	2020 1 Feb	2020 20 Feb
PHP	78.9%	78.9%	79.0%	79.1%	79.0%	79.0%	79.1%	79.1%	79.0%	79.0%	78.9%	78.9%	79.0%	78.9%
ASP.NET	11.6%	11.5%	11.4%	11.3%	11.2%	11.1%	11.0%	10.9%	10.9%	10.8%	10.7%	10.6%	10.6%	10.5%
Java	4.0%	4.0%	4.0%	4.0%	4.0%	4.0%	3.9%	3.8%	3.8%	3.8%	3.8%	3.7%	3.4%	3.6%
Ruby	2.4%	2.5%	2.5%	2.5%	2.5%	2.6%	2.6%	2.7%	2.7%	2.8%	2.9%	3.0%	3.1%	3.2%
static files	2.1%	2.1%	2.1%	2.1%	2.1%	2.1%	2.0%	2.0%	2.0%	2.0%	1.9%	1.8%	1.8%	1.7%
Scala	1.3%	1.3%	1.3%	1.3%	1.3%	1.4%	1.4%	1.5%	1.5%	1.6%	1.6%	1.6%	1.6%	1.6%
Python	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%	1.2%	1.2%	1.2%	1.2%	1.3%	1.3%	1.3%	1.3%
JavaScript	0.7%	0.7%	0.7%	0.7%	0.7%	0.7%	0.7%	0.8%	0.8%	0.8%	0.8%	0.8%	0.8%	0.9%
ColdFusion	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.5%	0.4%	0.4%
Perl	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.3%	0.2%	0.2%	0.2%
Erlang	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%	0.1%

https://w3techs.com/technologies/history_overview/programming_language



Usage of server-side programming languages for websites, 20 Feb 2020, W3Techs.com

COMP721 WebDev Vision

- Key content of this course
  - Server-side programming: PHP
  - PHP with database support
  - Web 2.0 technology: Ajax, Javascript, XML and JSON
  - Popular frameworks: Angular, Node.js, and JQuery
- By the end of the course you should know enough to be able to build modern, moderate-complex web applications
- Nota Bene
  - Website design (e.g. font, style, layout) is not our main concern, the focus is on Programming
  - This course gives you the solid foundation of web development (not a training course on specific tools/frameworks)

Assessment Structure

ASSESSMENT

Assessment Event	Individual/Group Task	Weighting %
Assignments (including 1 and 2)	Individual	50%
Lab Works (10 in total)	Individual	20%
Online Problem-Solving Quizs (including 1 and 2)	Individual	30%

Requirements to pass the paper

- minimum 35% pass in each coursework assessment item AND
- to obtain at least 50% overall

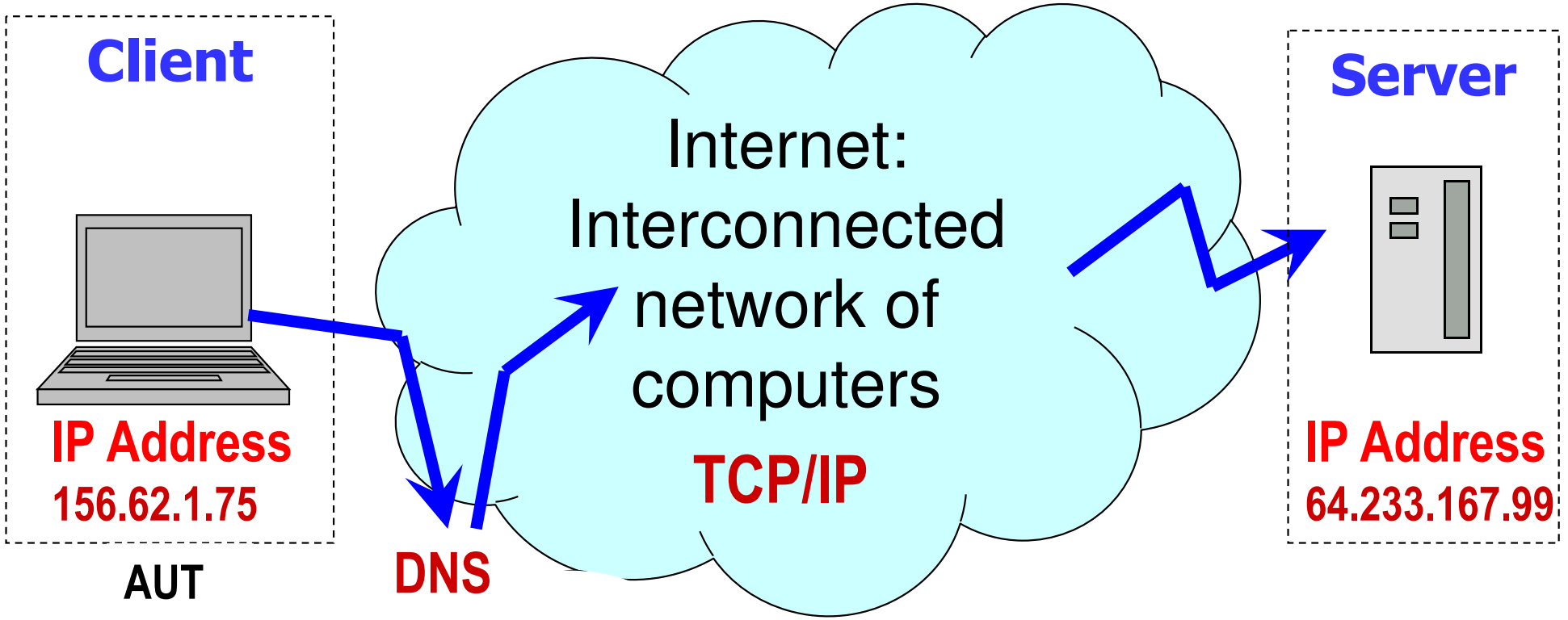
- Lab submission rule:
  - It is recommended that labworks are submitted in the same week or next week
  - If you submit your labwork 3 weeks after its release, 50% penalties apply

Zero tolerance to plagiarism

- Plagiarism, Collusion and Related Forms of Cheating:
  - ***Plagiarism*** = using someone else's work without acknowledgement
  - ***Collusion*** = helping someone else to plagiarise or cheat
  - ***Cheating*** = doing either of the above deliberately
  - Penalties include 0 marks for an assignment - including the person who did the work.
  - You must submit your own work and clearly identify others' appropriate contributions.

The Internet and The World Wide Web

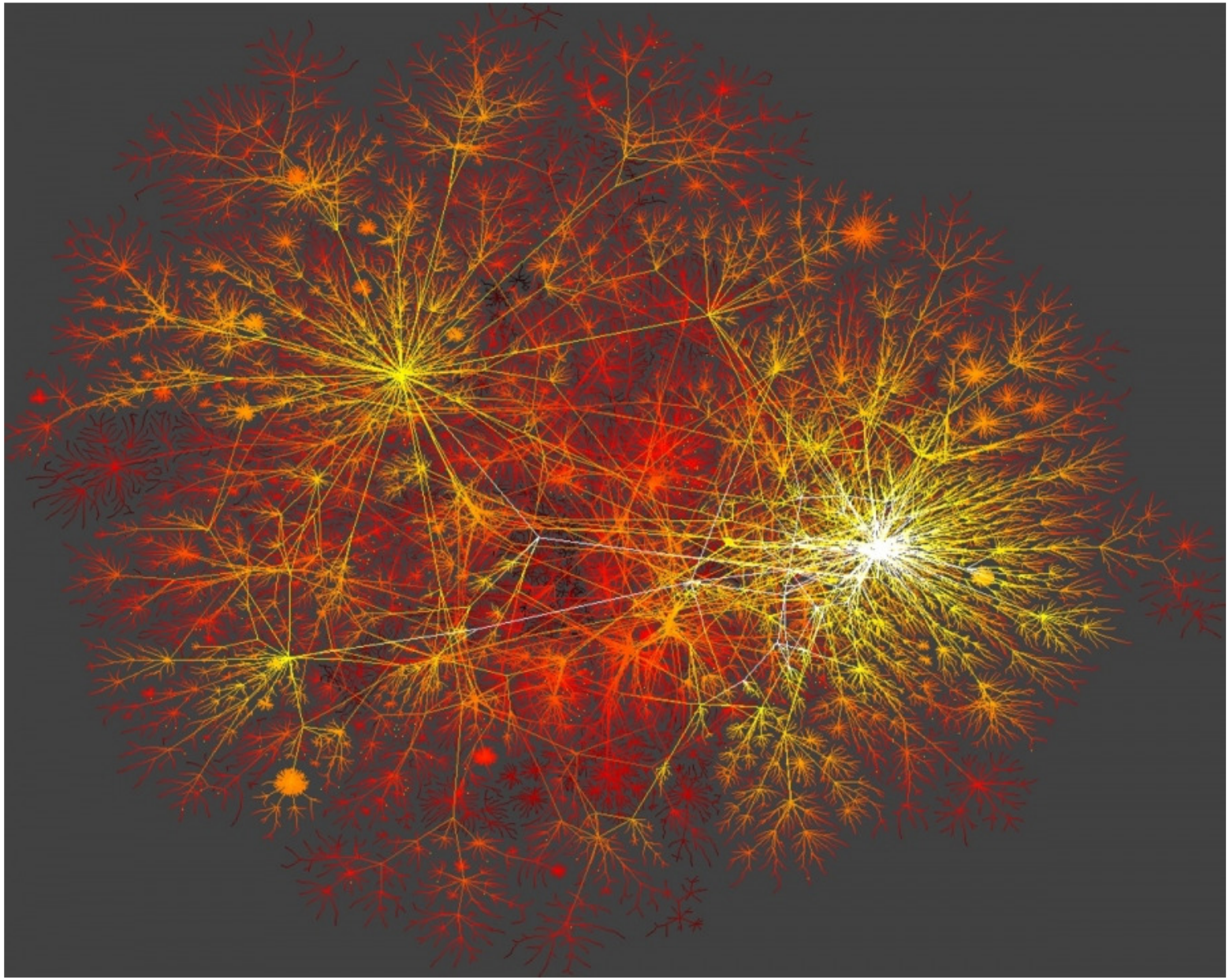
An overview of Internet



TCP *end-to-end reliable message transfer*

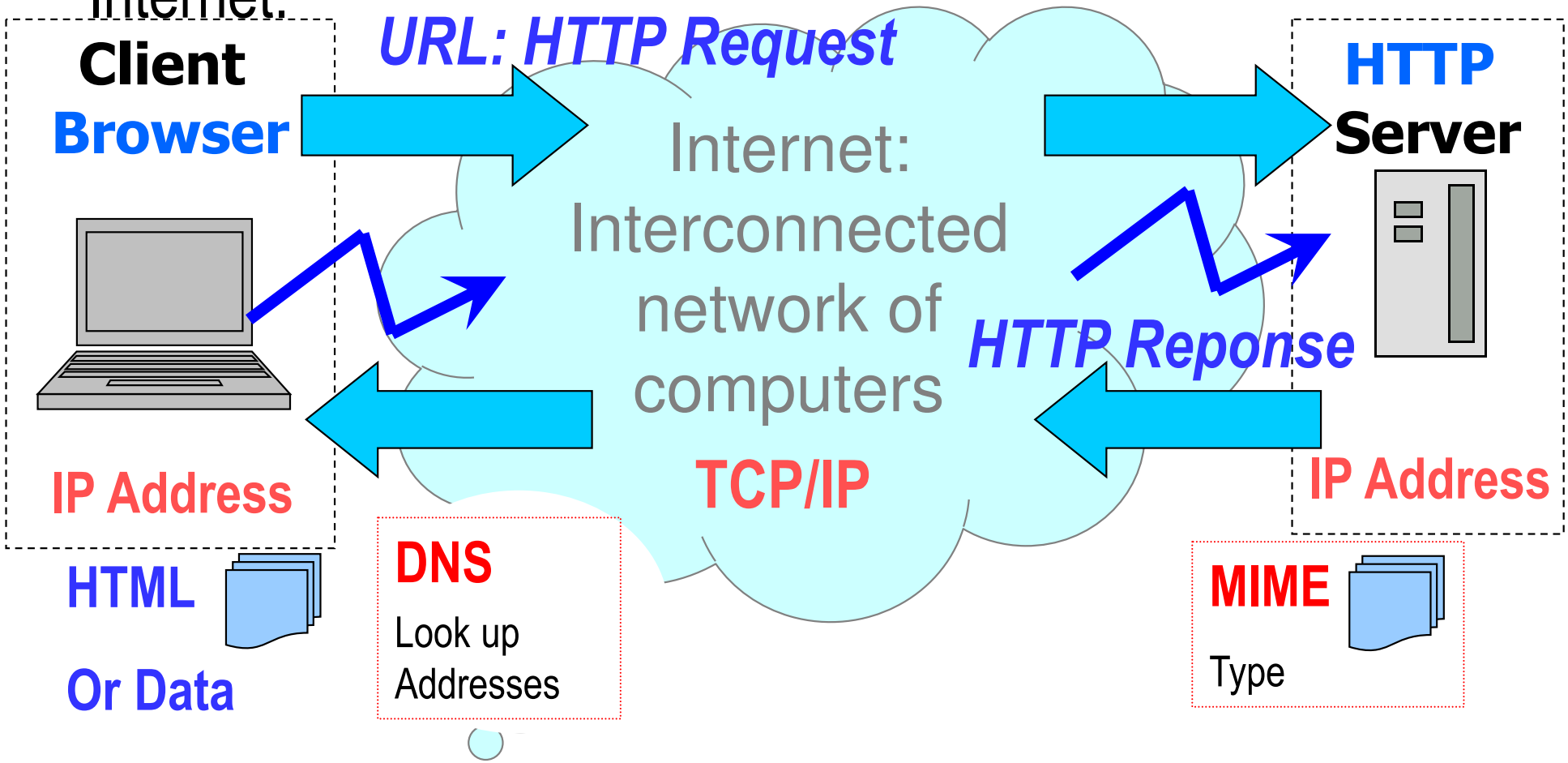
IP *handles the delivery*

The topology of the Internet



The Web – What is it?

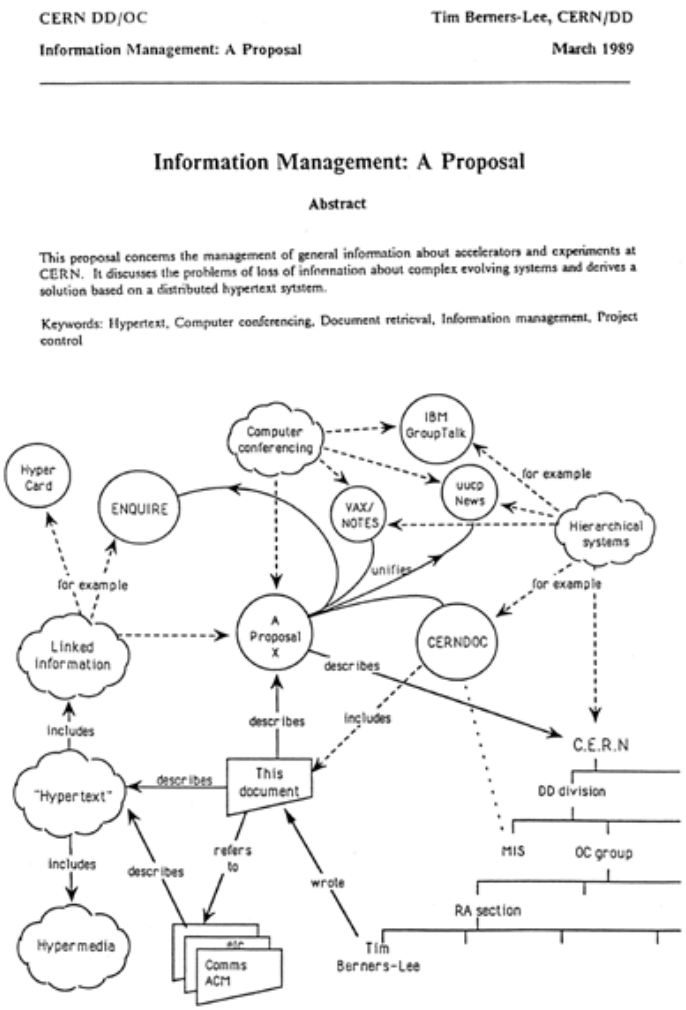
- is a way of accessing information over the medium of the Internet.



- Uses the **HTTP Protocol**

The Web – How did it started?

- In 1990 Sir Tim Berners-Lee (European Organization for Nuclear Research) authored a document outlining fundamentals of the web
  - The ability of links to cross machine boundaries (URIs)
  - "A simple, common protocol for exchanging hypertext documents" (HTTP)
  - A common document mark-up language (HTML)



CERN DD/OC — Tim Berners-Lee, CERN/DD

Information Management: A Proposal — March 1989

Information Management: A Proposal

Abstract

This proposal concerns the management of general information about accelerators and experiments at CERN. It discusses the problems of loss of information about complex evolving systems and derives a solution based on a distributed hypertext sytstem.

Keywords: Hypertext, Computer conferencing, Document retrieval, Information management, Project control

HTTP (Hypertext Transfer Protocol) Essentials

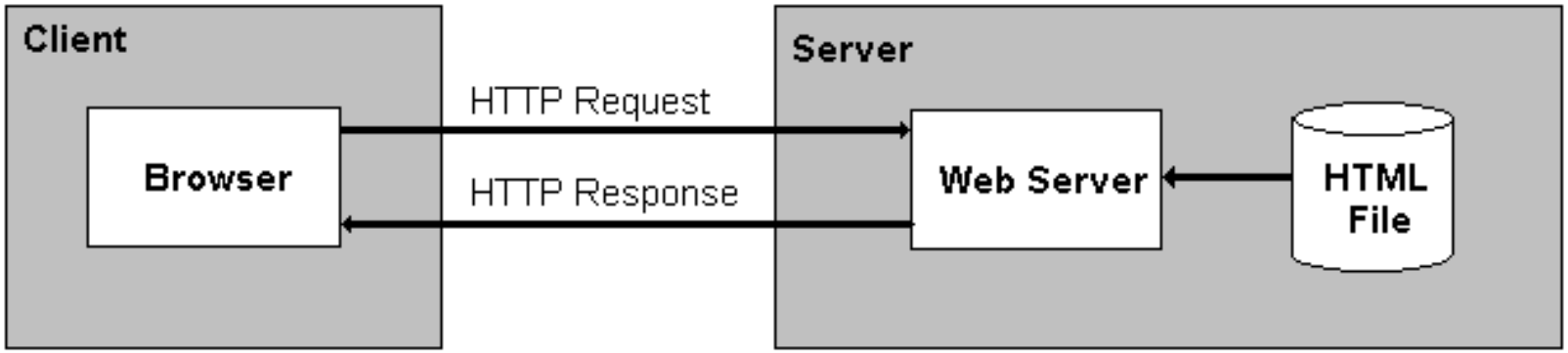
Typical HTTP Methods: GET

- HTTP GET **Look inside: http://websniffer.cc/**
  - A GET request retrieves data from a web server by specifying parameters in the URL portion of the request.
- Typical get request initial line:

```
GET /path/to/file/index.html HTTP/1.1
```

- Second line

```
Host: www.aut.ac.nz
```



An experiment: Send HTTP GET command through TCP

- http://telnet.browseas.com/
- Or from your computer using ‘telnet’ command

View HTTP Request and Response Headers

For more information on HTTP see RFC 2616

HTTP(S)-URL: https://www.aut.ac.nz/

Request type: GET ▼ HTTP version: HTTP/1.1 ▼ User agent: Web-Sniffer ▼

Share This Query

Share this query on a website or a forum by copying the following link:
https://websniffer.cc/?url=https://www.aut.ac.nz/

We have some interesting information about the domain **aut.ac.nz** and the IP addres

- **aut.ac.nz**
- **156.62.238.90**

HTTP Request Header

Connect to **156.62.238.90** on port **443** ... ok

```
GET / HTTP/1.1
User-Agent: WebSniffer/1.0 (+http://websniffer.cc/)
Host: www.aut.ac.nz
Accept: */*
Referer: https://websniffer.cc/
Connection: Close
```

HTTP GET with parameter

- https://www.google.co.nz/?gfe_rd=cr&ei=Gp7XVrKmOcbu8wfUqZW4BQ&gws_rd=ssl#q=AUT
- Query string: key-value pairs separated by '&'
  - gfe_rd=cr
  - ei=Gp7XVrKmOcbu8wfUqZW4BQ
  - gws_rd=ssl#q=AUT

HTTP Response

- Initial line is status line such as
  - `HTTP/1.0 200 OK`
  - `HTTP/1.0 404 Not Found`
- 200 and 404 are examples of **status codes**
- Message body will contain the requested resource if it was found

Typical HTTP Methods: POST

- HTTP POST
  - E.g., login into Blackboard/gmail
- The POST method is used when you want to send some data to the server, for example, file update, form data, etc.
- Sample post request:

```
POST /path/script.cgi HTTP/1.1
Host: www.google.com
User-Agent: HTTPTool/1.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 32

home=Cosby&favourite+flavour=flies
```

Compare GET and POST

	GET	POST
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs

BACK button/Reload	Harmless	Data will be re-submitted (some browser will alert)
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data (for binary data)

HTML Essentials

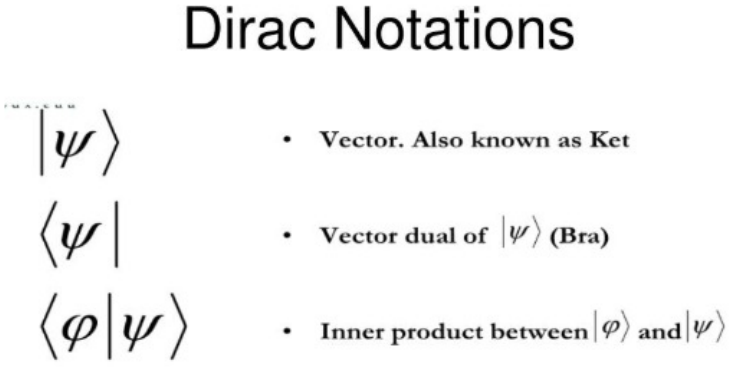
A note on client-side technologies

- A major guiding principle of CS
  - Separation of concerns - Edsger Dijkstra
- Applied in client-side technologies
  - HTML – content
  - CSS – style and appearance
  - JavaScript – action

What is HTML

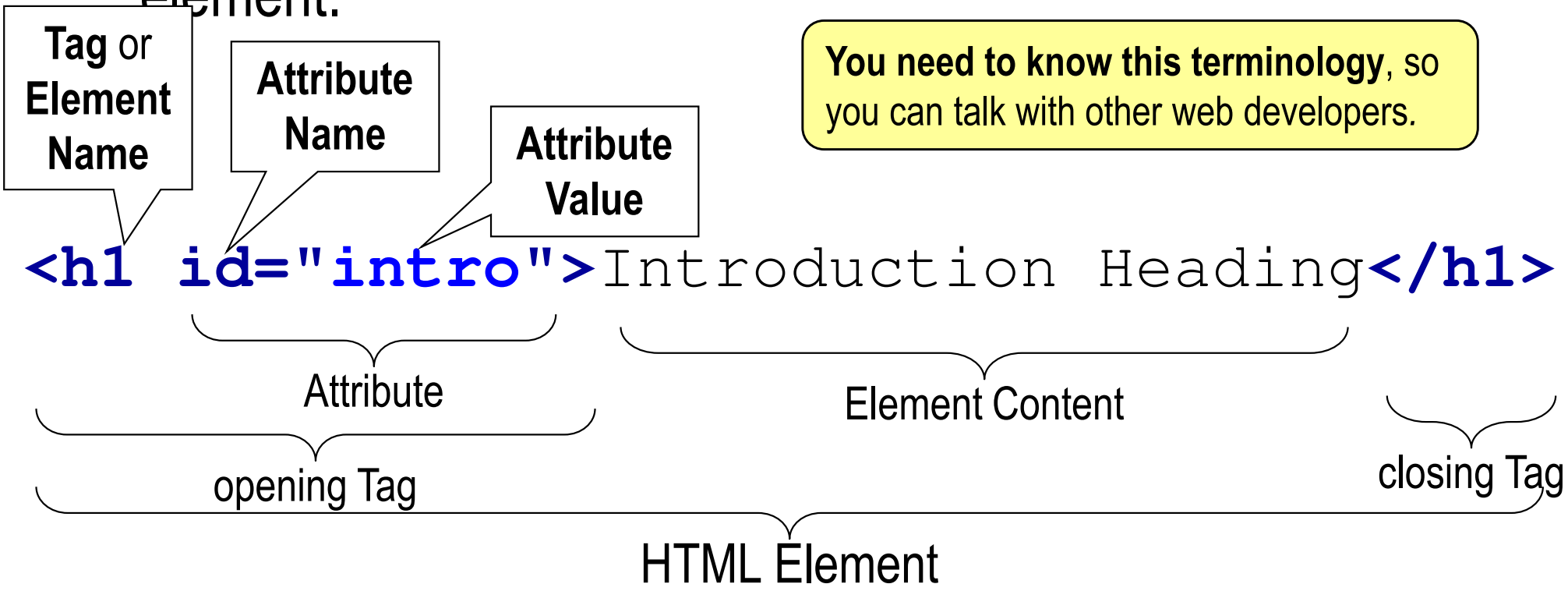
- HTML = **H**yper **T**ext **M**arkup **L**anguage (hyper: beyond)
- Current version: HTML5 (since 2011) – maintained by W3C
- Describes the content & structure of information on web page
  - Not the appearance on screen
- Composed of HTML elements

  -Each element contain opening and closing **tags**
  - Tags are enclosed in brackets (**< >**)
  - The information contained within an element's opening and closing tags is referred to as its **content**
  - A tag pair and the data it contains are referred to as an **element**
  - Element example: Hello



HTML Syntax

- A HTML **element structure** includes: **opening tag**, **tag name**, an attribute name (e.g. **id**) with an **attribute value** (e.g. **"intro"**), the **element content** (the text affected by the tag meaning), and finally the **closing tag** of the element.



HTML Tag & Attributes

- Tagname is case insensitive, i.e. = <P>
  - But to have a good habit, always use lowercase for tagname
- Attributes come in name/value pairs; values always inside double quotation marks
  - Syntax: attribute_name="attribute_value"
  - Example: Hi
- Attribute name and value are case sensitive
  - Always use lowercase for attribute name

HTML Attributes

- Each HTML tag may have different attributes
  - Have a look at w3school for the list of attributes
- But, all HTML tags have the following attributes
  - id: Unique id to identify the element
  - title: Extra information, displayed as tooltip text
  - style: Inline CSS style for the element

    <h1 style="color:blue;text-align:center">This is a header</h1>
  - class: One or more CSS class name for the element

HTML Syntax (continued)

Common HTML elements

HTML Element	Description
	Formats enclosed text in a bold typeface **Use **
<body></body>	Encloses the body of the HTML document
  ~~ ~~	Inserts a line break ***void element. Self closed***
~~ ~~ ***Use CSS***	Centers a paragraph in the middle of a Web page ***Deprecated***
<head></head>	Encloses the page header and contains information about the entire page
<hn></hn>	Indicates heading level elements, where *n* represents a number from 1 to 6
<hr /> ~~<hr>~~	Inserts a horizontal rule ***void element. Self closed***
<html></html>	Begins and ends an HTML document; these are required elements
	Formats enclosed text in an italic typeface **Use **
 <img src="…">	Inserts an image file ***void element. Self closed***
	Identifies enclosed text as a paragraph
~~<u></u>~~ ***Use CSS***	Formats enclosed text as underlined ***Deprecated***

Note: Use best standards / current practices

HTML Syntax (continued)

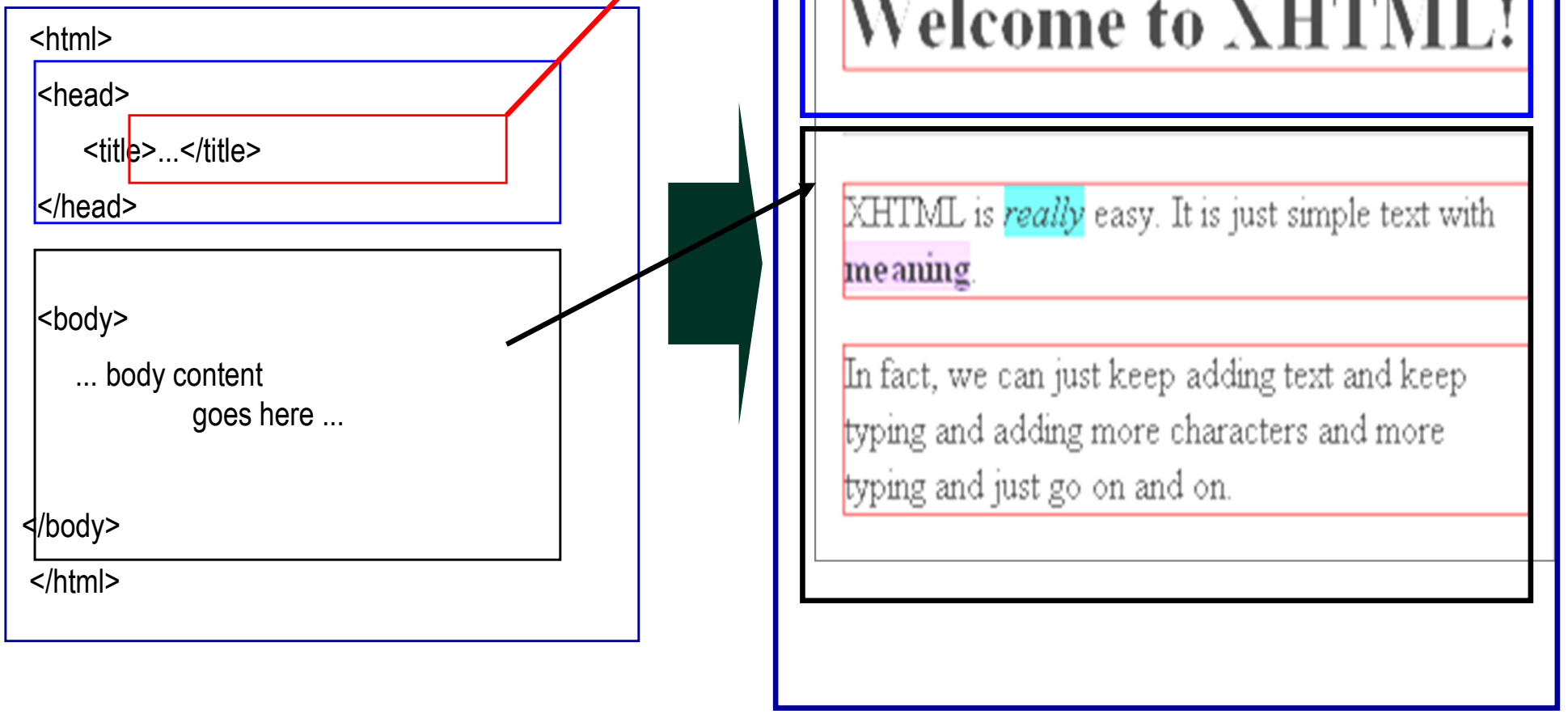
- HTML documents must have a file extension of .html or .htm
- All **HTML** documents must use the **<html>** element as the root element
- A **root element** contains all the other elements in a document
- The **<head>** element contains information that is used by the Web browser
- A <head> element must contain a **<title> element (**content displayed in the browser's title bar and bookmark list)
- The <head> element and the elements it contains are referred to as the **document head**

HTML Syntax (continued)

- The <body> element and the text and elements it contains are referred to as the **document body**
- The process by which a Web browser assembles or formats an HTML document is called **parsing** or **rendering**
- Example:
  This paragraph will appear in boldface in a Web browser
- Parameters used to configure HTML elements are called **attributes**
- Insert line breaks using the paragraph ** ** and line break **
** elements

HTML Syntax (continued)

- The simple basic structure of HTML documents:



Note that only the <body> section/element is graphically rendered to the user on the browser GUI

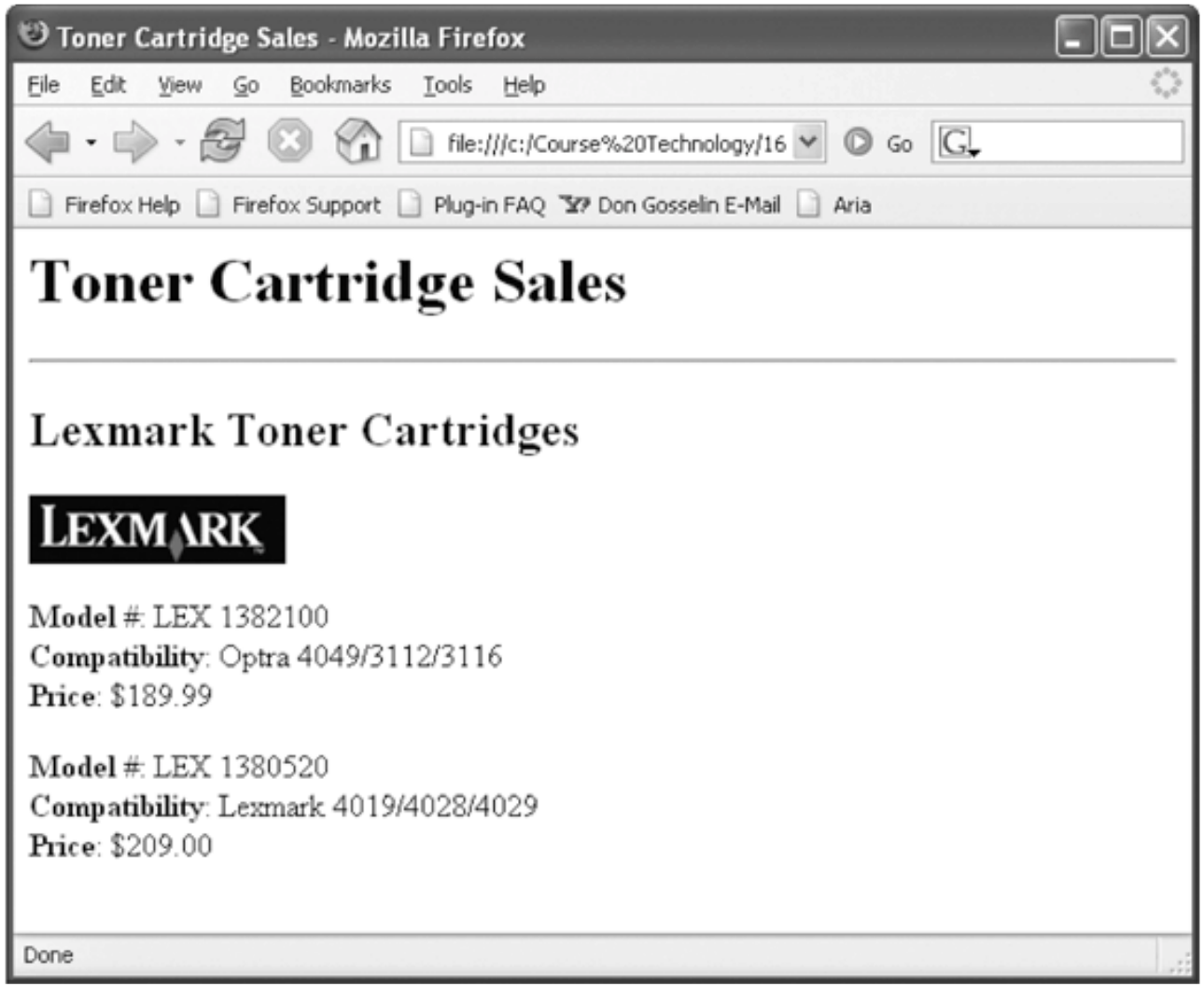
HTML Syntax (continued)

Sample HTML Code

```
<html>
 <head>
   <title>Toner Cartridge Sales</title>
 </head>
 <body>
  <h1>Toner Cartridge Sales</h1>
  <hr />
  <h2>Lexmark Toner Cartridges</h2>
  <img src="lexmark_logo.gif" alt="logo" />
  <p><strong>Model #</strong>:LEX 1382100<br />
   <strong>Compatibility</strong>: Optra 4049/3112/3116<br />
   <strong>Price</strong>: $189.99</p>
  <p><strong>Model #</strong>:LEX 1380520<br>
   <strong>Compatibility</strong>:Lexmark 4019/4028/4029<br />
   <strong>Price</strong>:$209.00</p>
 </body>
</html>
```

Question:
What's the abstract data structure of a HTML document?

HTML Syntax (continued)



A simple HTML document in a Web browser

Creating an HTML Document

- You cannot use a Web browser to create an HTML document
- Can use any editors
- HTML editors, such as Macromedia Dreamweaver, are popular graphical interfaces that create WYSIWYG (what-you-see-is-what-you-*might*-get) Web pages, better to use 'Code View' so you understand the HTML
- Best to use simple Text Editors, such as Crimson Editor, NotePad++ that can give you better control.

Some essential HTML tags

- **Heading**
- Tagname: h1, h2, h3, h4, h5, h6
- Purpose: Title of chapters, sections or subsections
  - Separating major areas of the document
- Example:

```
<html>
        <head>
                <title> My First Web page</title>
        <body>
                <h1>Chapter Title</h1>
                <p>Hello World!</p>
                <h2> Section Title </h2>
                <p> Try to write a long text here </p>
        </body>
</html>
```

Horizontal Rule

- Tagname: hr
- Purpose: Display horizontal line to separate areas

```
<html>
        <head>
                <title> My First Web page</title>
        <body>
                <h1>Chapter Title</h1>
                <p>Hello World!</p>
                <hr/>
                <h2> Section Title </h2>
                <p> Try to write a long text here </p>
        </body>
</html>
```

Hyperlink

- Tagname: a
- Purpose: Create a hyperlink
- Attributes:
  - href: The URL of the linked page
    - Absolute (http://www.google.com) or relative (1/test.html)
  - hreflang, media, rel, target, type
    - Rarely used (Look in the w3schools website)
- Example: please click here to visit google

Image

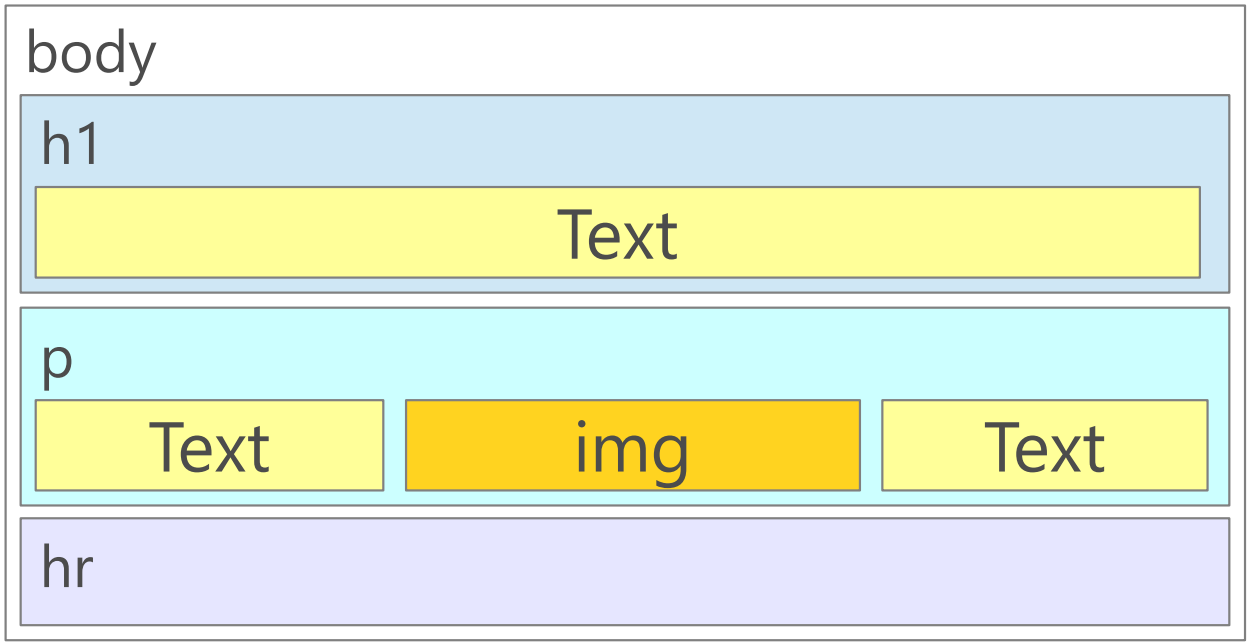

```
<!DOCTYPE html>
<html>
	<head>
		<title>My First Webpage</title>
	</head>
	<body>
		<h1>Chapter Title</h1>
		<p>Hello <img src="flower.jpg"  alt="flower" width="50" height="80"></img> World!</p>
		<hr></hr>
		<h2>Section Title</h2>
		<p>Try to <a href=”http://www.baidu.com”>write</a>
		    a long text here</p>
	</body>
</html>
```

List

- Tagname: ul
  - Purpose: Create a bullet (unordered) list
- Tagname: ol
  - Purpose: Create a number (ordered) list
- Tagname: li
  - Purpose: Create an item in the list
  - Location: Within the ol or ul tags
- All are block tags

Block and Inline

- Block: Elements start and end with a new line
  - h1, hr, p
- Inline: Elements are displayed on the same line
  - a, img



List

- To create bullet list that looks like
  - Apple
  - Orange
  - Banana
- Specify the ul tag
- For each item, create a li tag

```
<ul>
    <li>Apple</li>
    <li>Orange</li>
    <li>Banana</li>
</ul>
```

Table

- To create table that looks like

```
<table border="1">
<tr>
 <th>No.</th>
 <th>Country</th>
 <th>Capital City</th>
</tr>
<tr>
  <td>1</td>
  <td>New Zealand</td>
  <td>Wellington</td>
</tr>
<tr>
  <td>2</td>
  <td>Australia</td>
  <td>Canberra</td>
</tr>
</table>
```

No.	Country	Capital City
1	New Zealand	Wellington
2	Australia	Canberra

HTML5 new tags - <canvas>

- <canvas>
  - used to draw graphics using JavaScript

```
<html>
<body>
<canvas id="mycanvas" width="400"
height="300">draw a rectangle</canvas>
<script>
var can=document.getElementById('mycanvas');
var can2d=can.getContext("2d");
can2d.fillStyle='#0000ff';
can2d.fillRect(25,25,180,100);
</script>
</body>
</html>
```

HTML5 new tags - Media Elements

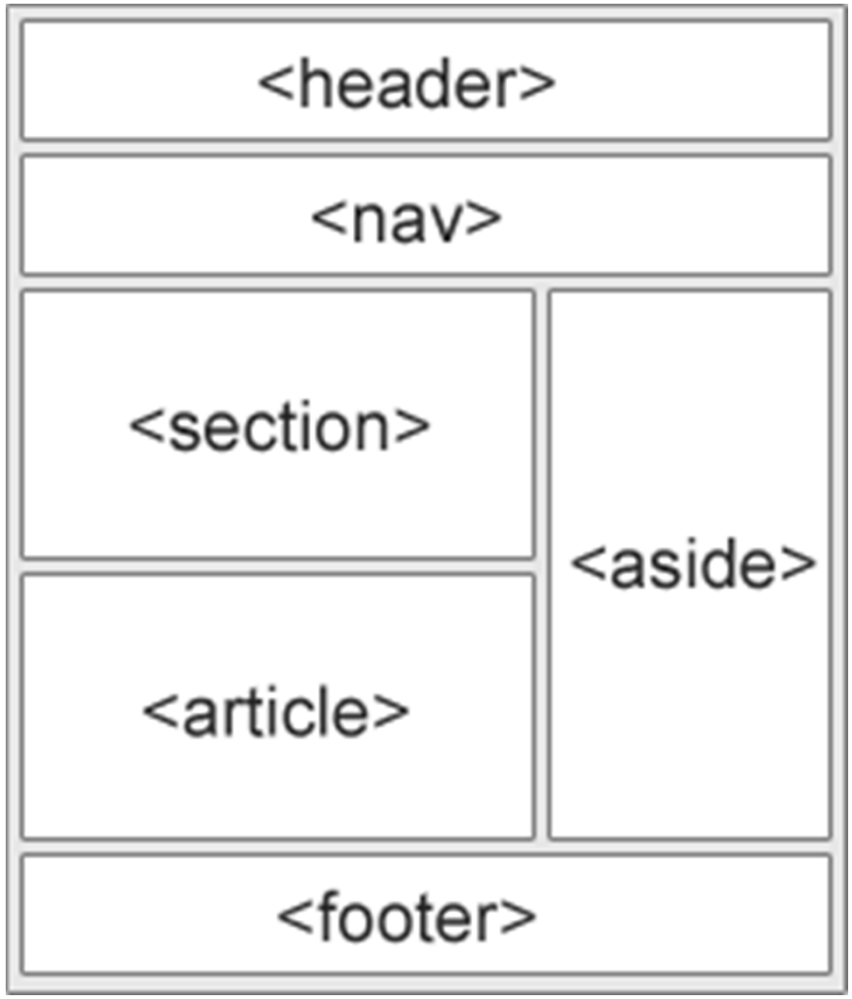
- <audio>, <video>, <embed>, <source>, <track>

```
<html><body>
<video width="400" controls>
  <source
src="https://www.w3schools.com/html/mov_bbb.mp4
" type="video/mp4">
Your browser does not support HTML5 video.
</video>
<p>
Video courtesy of
<a href="https://www.bigbuckbunny.org/"
target="_blank">Big Buck Bunny</a>.
</p></body></html>
```

- <embed>: embed flash swf files, and html snippets
- <track>: specifies text tracks for media elements

HTML5 new tags – semantic tags

- Tags that give meaning (useful to writers)
  - We also find them in “MS Word”



HTML: Syntax References

The W3C HTML Standards / References

http://www.w3.org/

HTML Tutorials

https://www.w3schools.com/html/html5_intro.asp

NB: HTML forms
will be discussed in week 3