CS521

WEB SYSTEMS AND TECHNOLOGIES

SLIDES WEEK [1 TO 6]

MADE BY MAH JABEEN

0321 2711298
mahjabeen97869@gmail.com

Introduction to Web Systems and Technologies

TOPIC 01
MODULE 01

Web Systems and Technologies

Introduction to Web Systems and Technologies

06-06-2024
Cs521
Tech trailblazer
Long
1. semantic tags
2. Change the background color of heading and paragraph
3. Write the ouptput of the program

```
Sum=0
{
For(i=2; i<=6; i++)
 (i%2 !=0) then
+sum= i
}
Consol.log(sun)
}
```

Short
1. Web 3.0 2.0 1.0 k features
2. Write a js program and show current date and current time using the builtin date object
3. Position property and its types

Introduction to Web Systems and Technologies

Topics (Basics)

1. What is web and how does it work?
2. Structuring web contents using HTML
3. Setting visual layout using CSS
4. Making web interactive using JavaScript
5. Web backend programming using PHP and databases

Introduction to Web Systems and Technologies

Topics (Advanced)

6. Making websites secure
7. Introduction to web services
8. Front-end and back-end frameworks
9. Hosting websites
10. Search engine optimization (SEO)

Introduction to Web Systems and Technologies

Textbook

1. "Fundamentals of Web Development", 2nd Edition by Randy Connolly and Ricardo Hoar, 2018, Pearson

Reference websites

1. https://www.w3schools.com/
2. https://developer.mozilla.org/en-US/
3. https://reactjs.org/

Introduction to Web Systems and Technologies

Credits

- Material for the slides used in this course is provided by *Pearson Education*

Reference websites

1. https://www.w3schools.com/
2. https://developer.mozilla.org/en-US/
3. https://reactjs.org/

Web Systems and Technologies

Module 02

Web Systems and Technologies

TOPIC 02

Ecosystem of Web

Ecosystem of Web

A Complicated Ecosystem

Ecosystem of Web

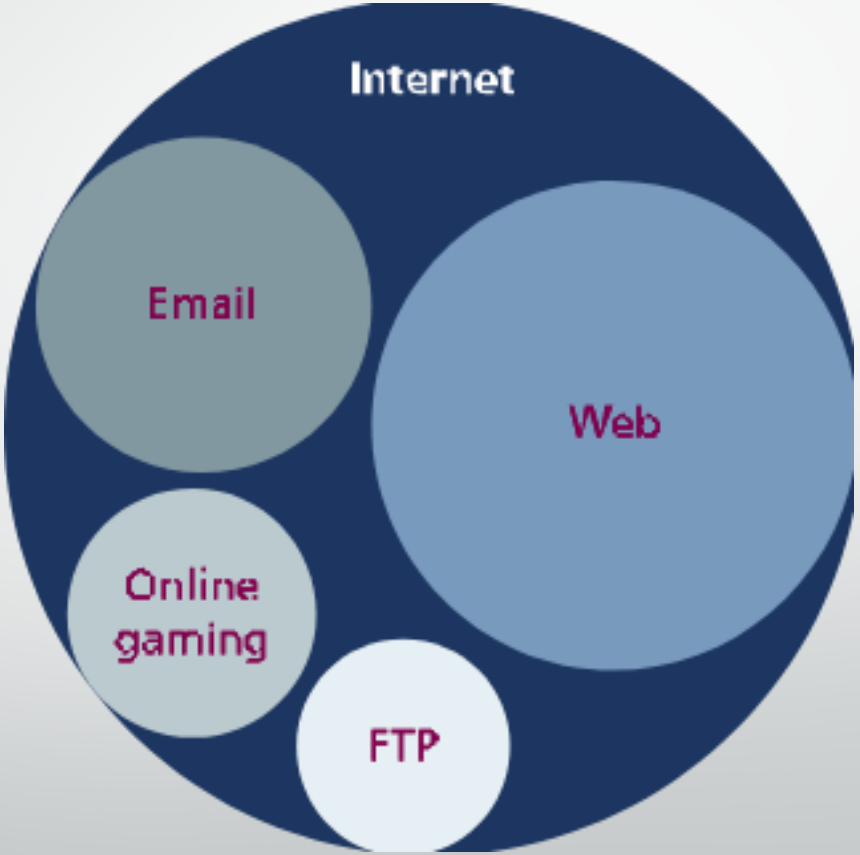
A Short History of the Internet

- Telephone Network
- Packet Networks
  - ARPANET (1969)
  - X.25 (1974)
  - USENET (1979)
  - TCP/IP (1983) ← INTERNET

first network
kn sa tha

Ecosystem of Web

Internet vs. (World Wide) Web



Ecosystem of Web

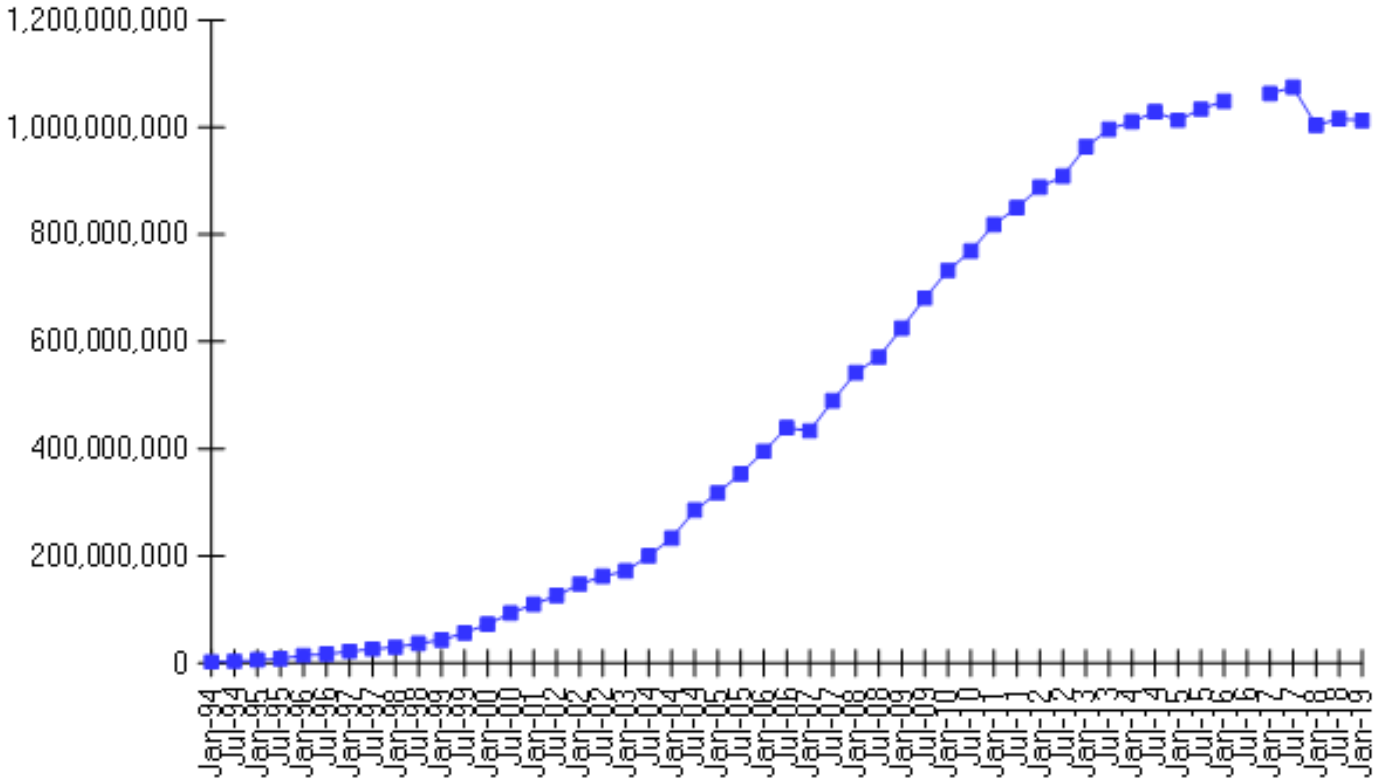
The Birth of the Web (1990)

- URLS
- HTTP
- SERVERS
- BROWSERS
- HTML

Ecosystem of Web

Growth of Internet

- Number of Internet Hosts



Source: Internet Systems Consortium (www.isc.org)

Ecosystem of Web

Web Apps vs. Desktop Apps

- Advantages
  - Accessible from any Internet-enabled computer
  - Cross platform and browser compatibility
  - Easier to roll out program updates
  - Fewer security concerns about local storage

Ecosystem of Web

Web Apps vs. Desktop Apps

- Disadvantages
  - Requires internet
  - Security concerns about sensitive private data being transmitted over the Internet
  - Concerns over the storage, licensing, and use of uploaded data
  - Problems with certain websites not having an identical appearance across all browsers
  - Restrictions on access to operating system resources
  - May need plugins for functioning

Web Systems and Technologies

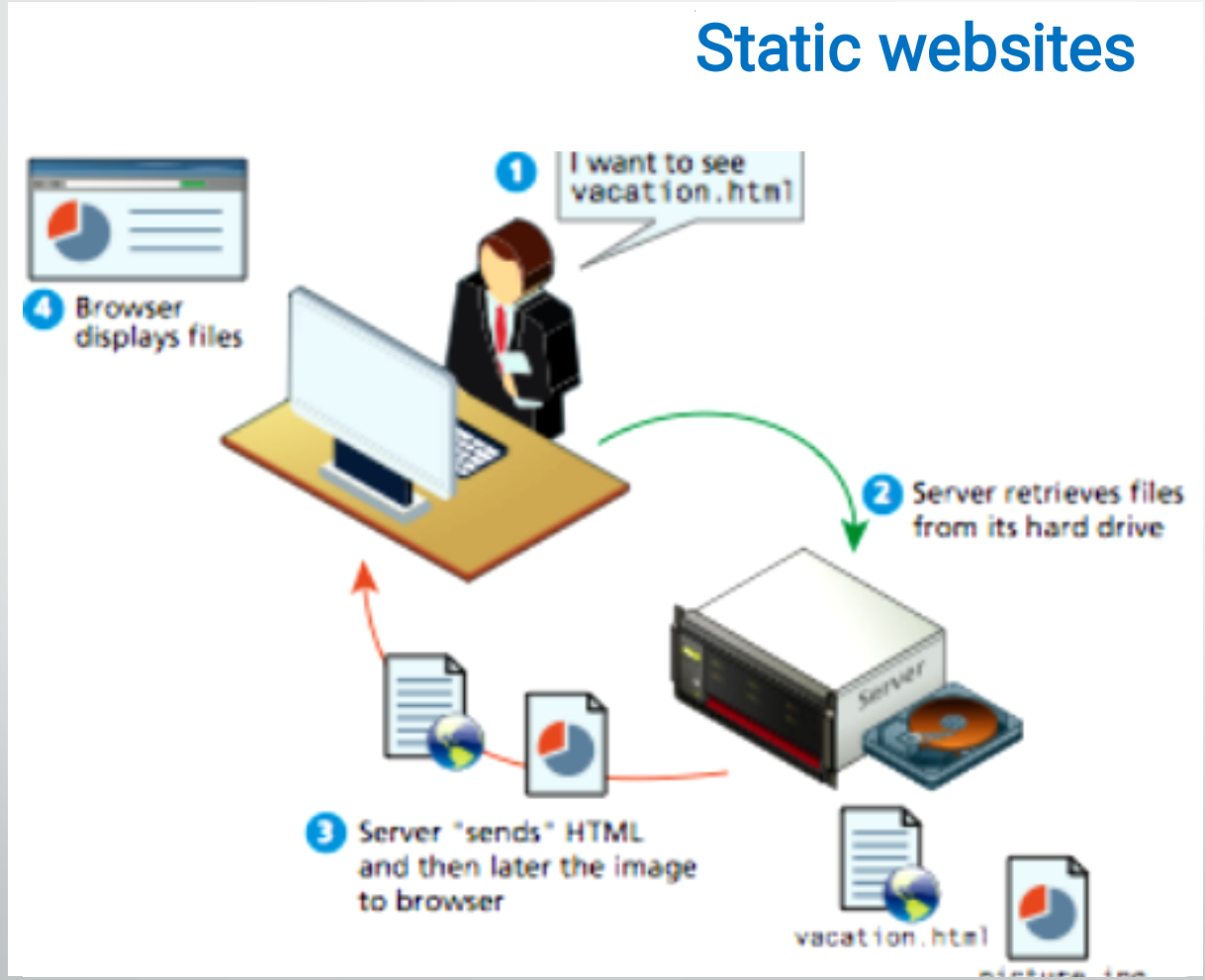
TOPIC 03

MODULE 03

Web Systems and Technologies

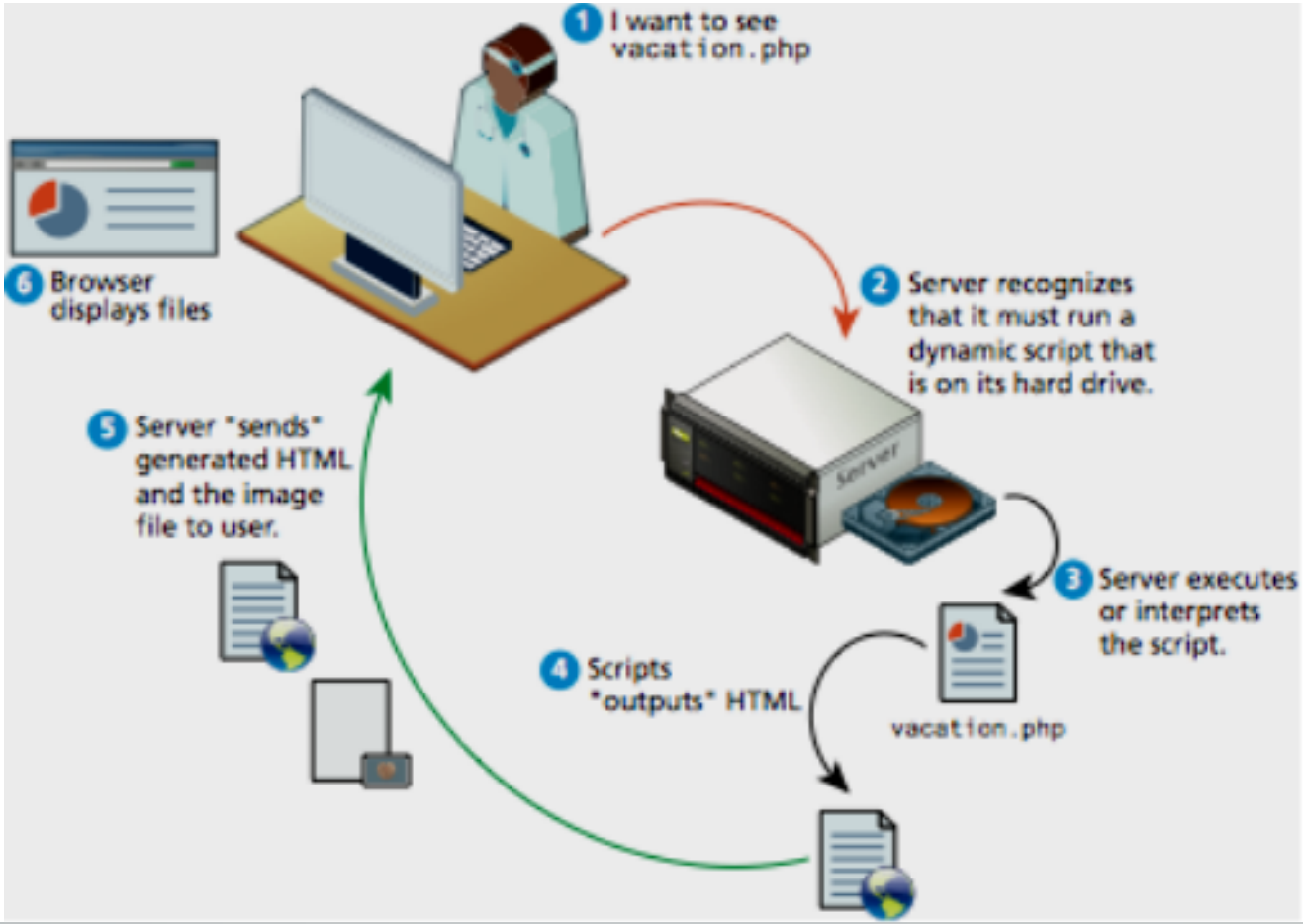
Static vs Dynamic Websites

Static vs Dynamic Websites



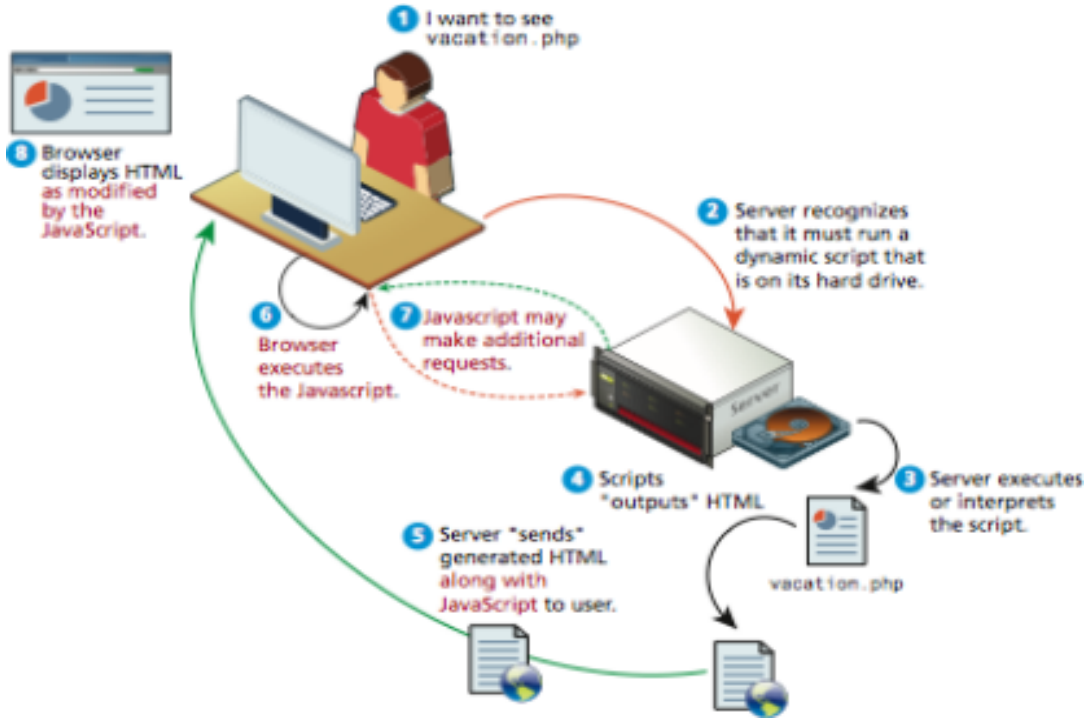
Static vs Dynamic Websites

(Traditional) Dynamic Websites



Static vs Dynamic Websites

(Modern) Dynamic Websites



Web Systems and Technologies

TOPIC 04

MODULE 04

Web Systems and Technologies

Brief History of Web

Brief history of Web

features of web short qusetion

Web 1.0

- Contents generated by publishers
- Company generated companies
- Static websites
- Dynamic websites

Brief history of Web

Web 2.0

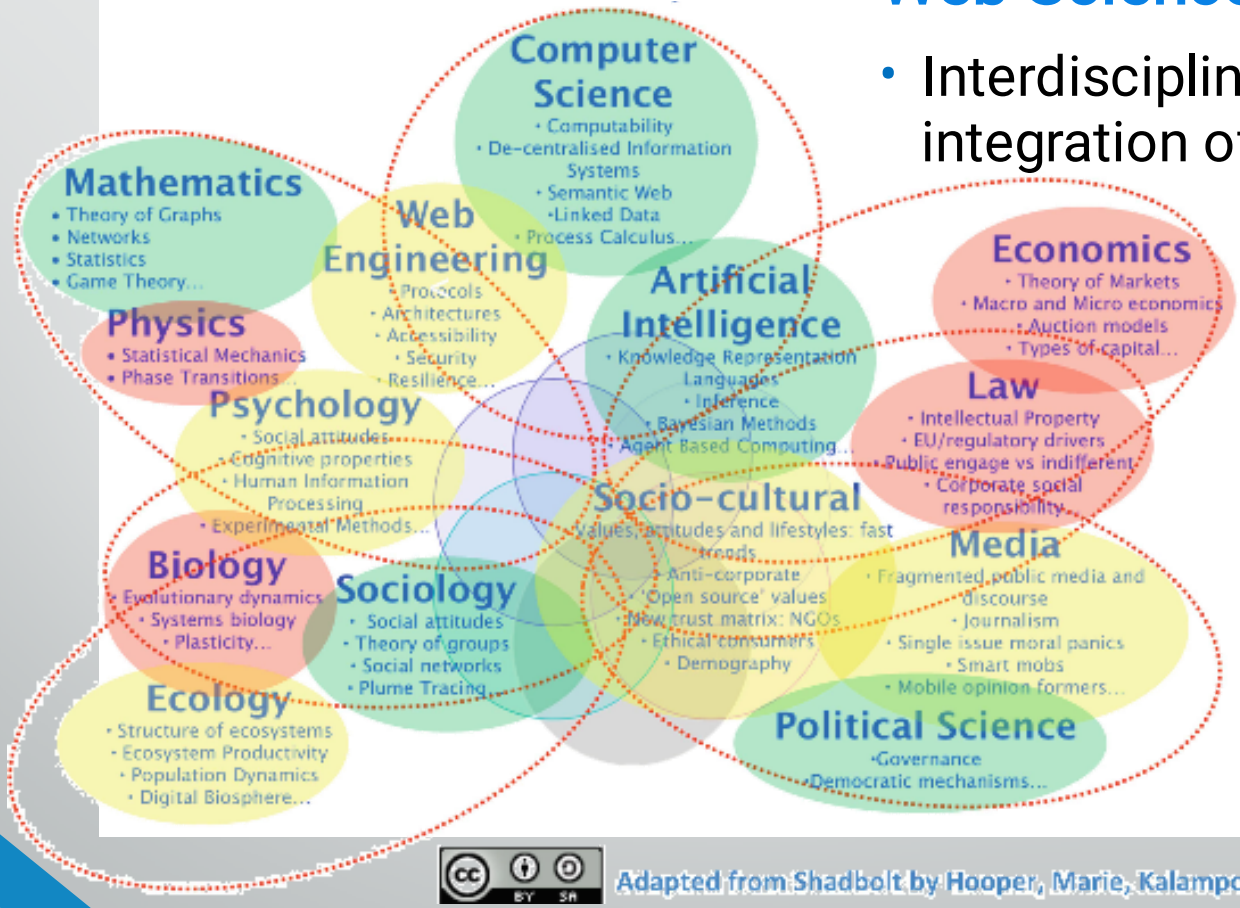
- User Generated Contents (UGC)
- Dynamic websites

Brief history of Web

Web 3.0

- Machine understandable contents
- Semantic web

Brief history of Web



Web Science

- Interdisciplinary study of sociotechnical integration of the world wide web

Web Systems and Technologies

TOPIC 05

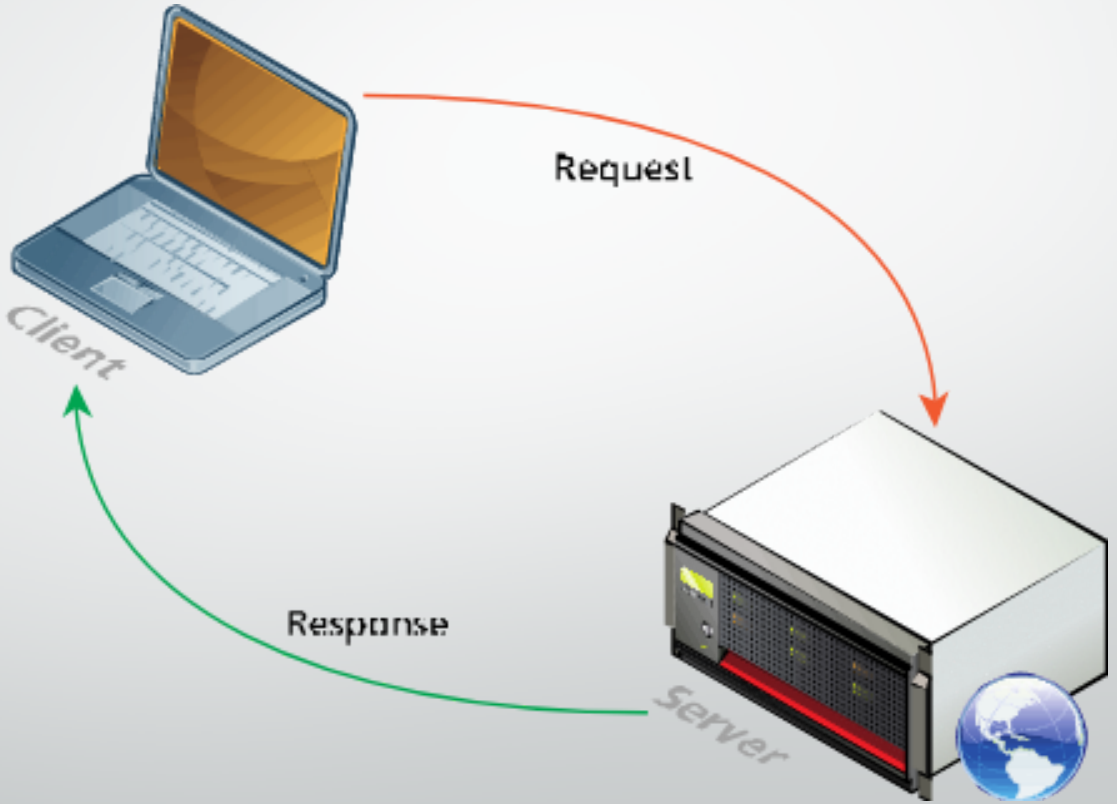
MODULE 05

Web Systems and Technologies

Client Server Model

Client Server Model

The request-response Loop

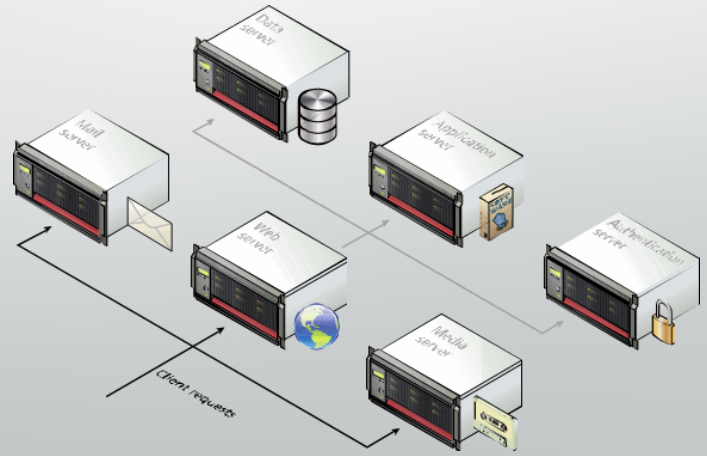


Client Server Model

Types of Servers

- Web Servers
- Application Servers
- Database Servers
- Mail Servers
- Media Servers
- Authentication Servers
- …

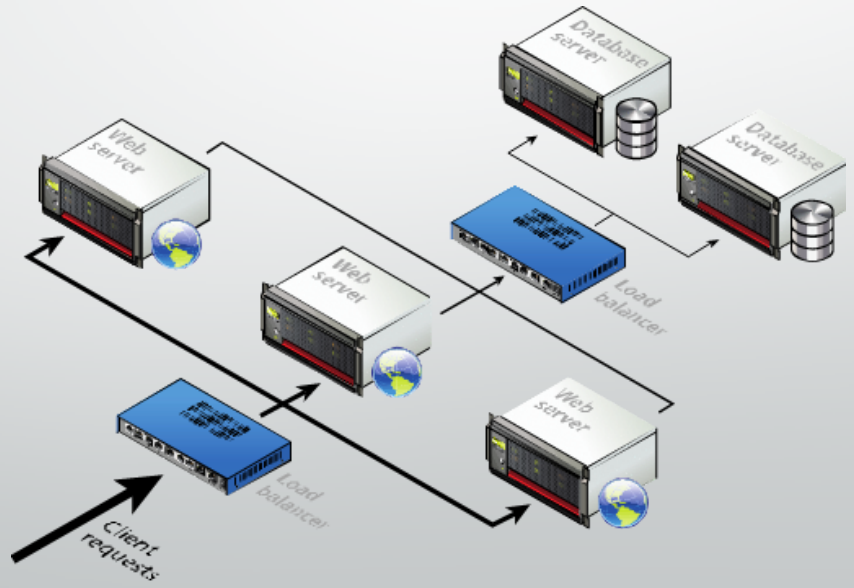
which type of server we use for authentication



Client Server Model

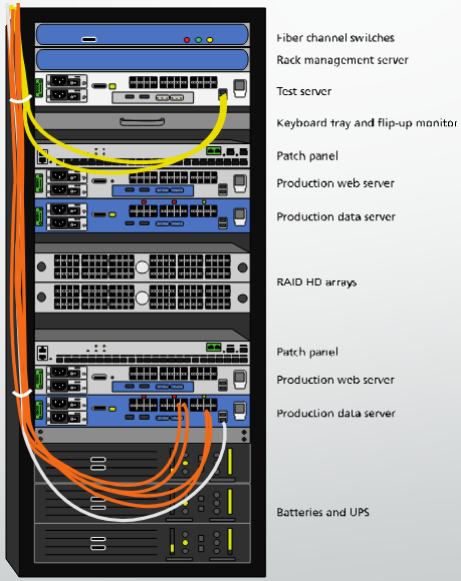
Server Farms

- Large websites receives millions of requests per minute
  - Cannot be handled by a single server
- Server farms are used for scalability



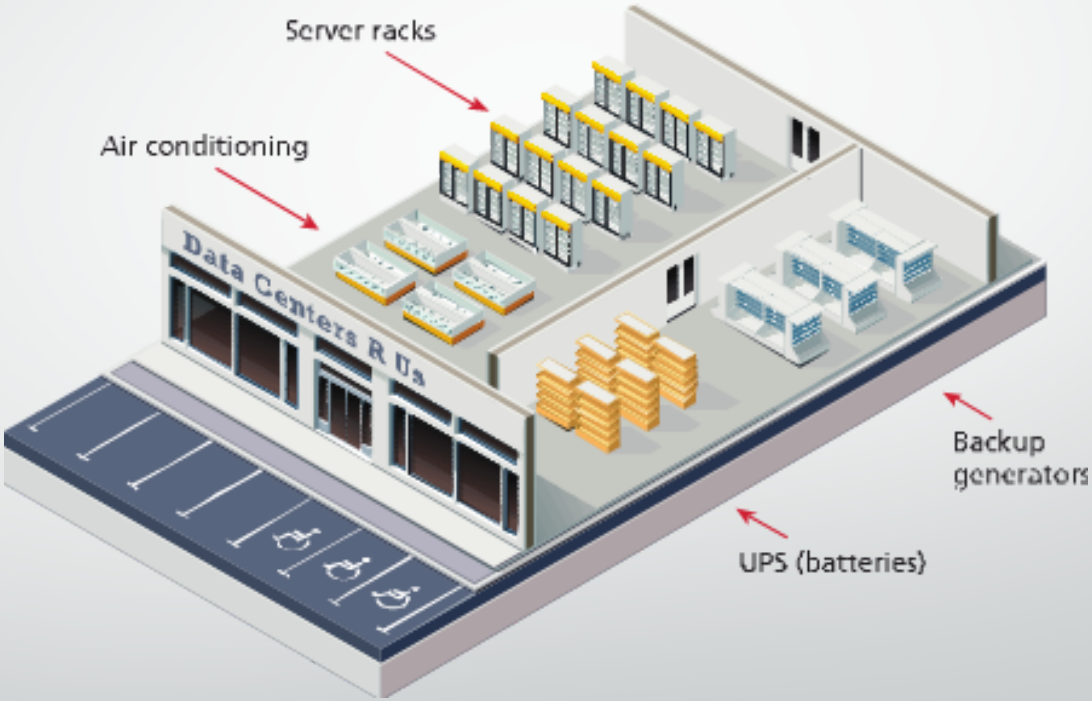
Client Server Model

Server Racks



Client Server Model

Data Centers

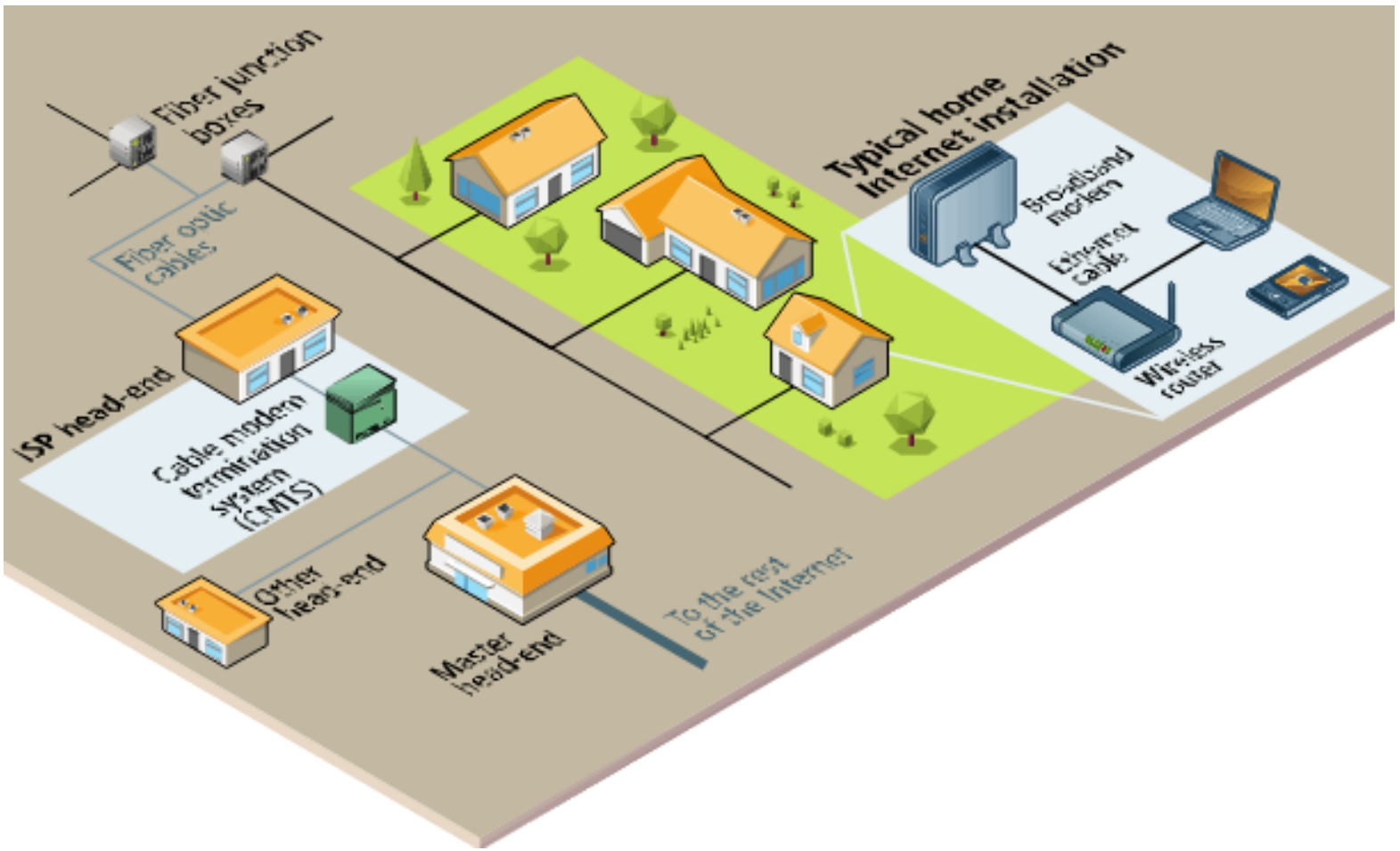


Web System and Tecnologies CS 521

TOPIC 06

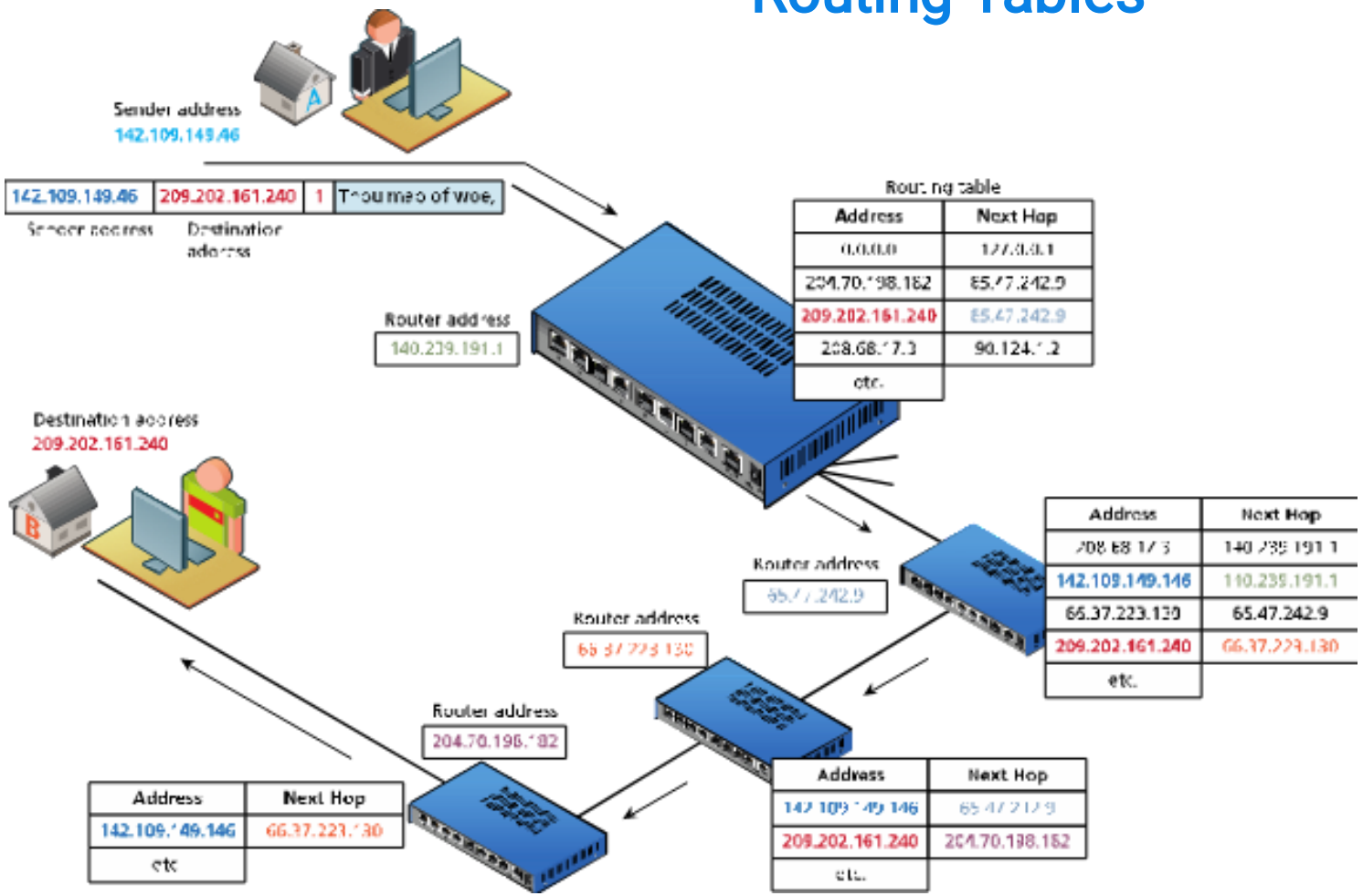
How Internet Works?

Internet from Home to ISP



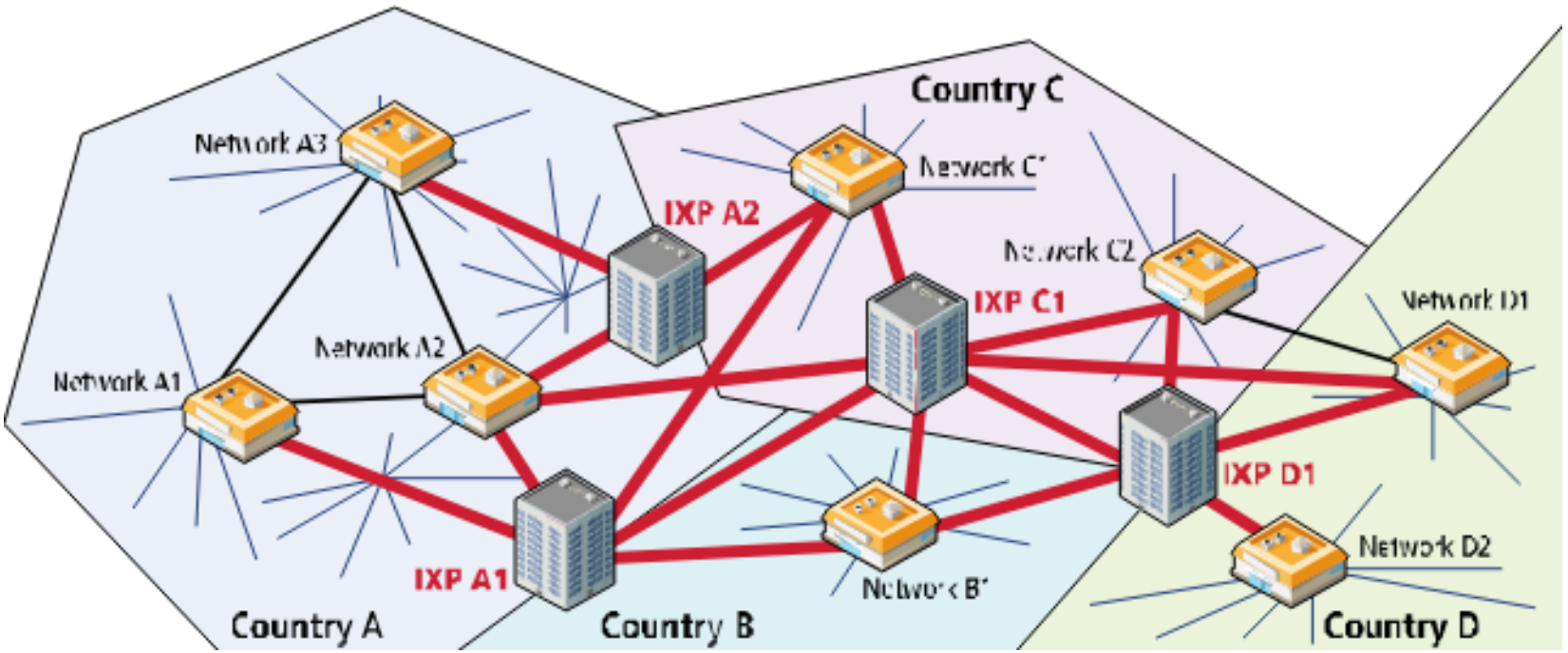
How Internet Works?

Routing Tables



How Internet Works?

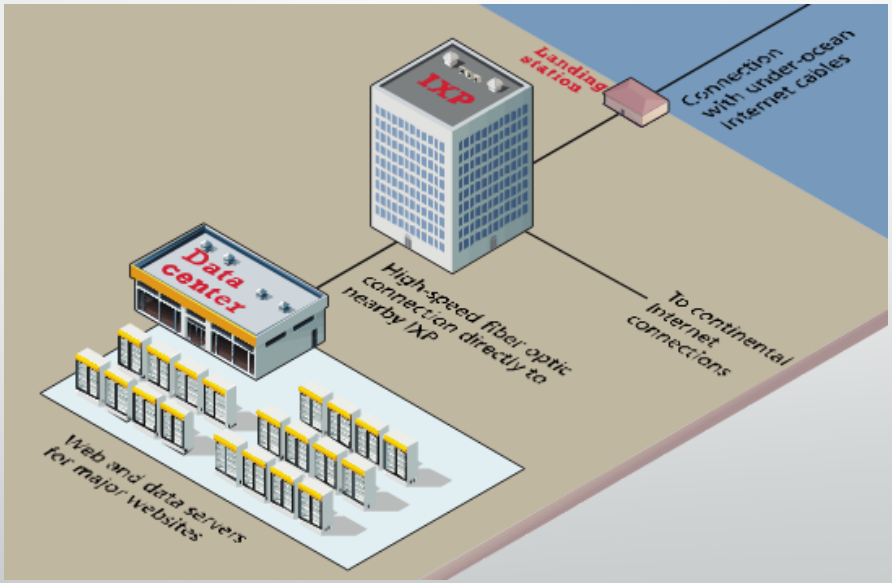
From ISP to the World



How Internet Works?

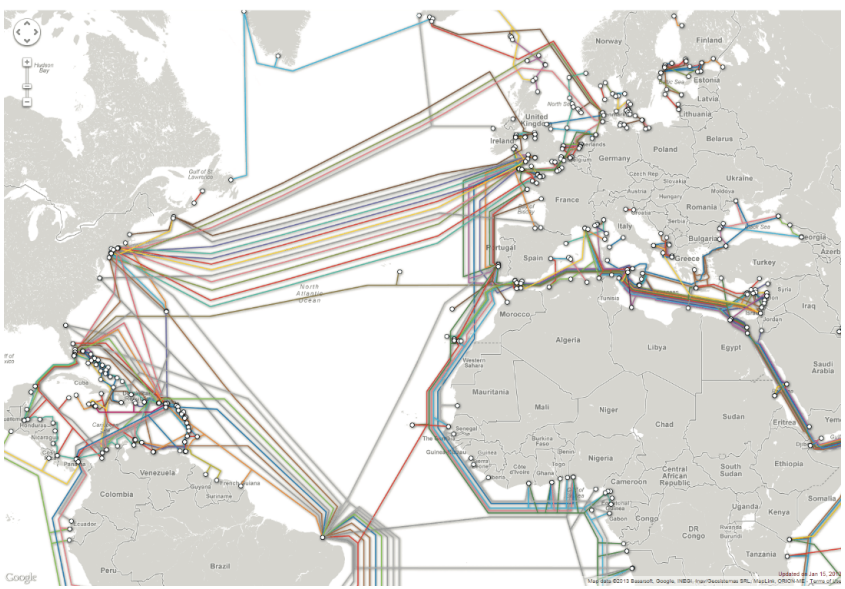
IXP

- IXP – Internet Access Point – Dedicated infrastructure to make the internet more connected.



How Internet Works?

Under-ocean Connectivity



The End

Web System and Tecnologies CS 521

Topic 07

Internet Protocols

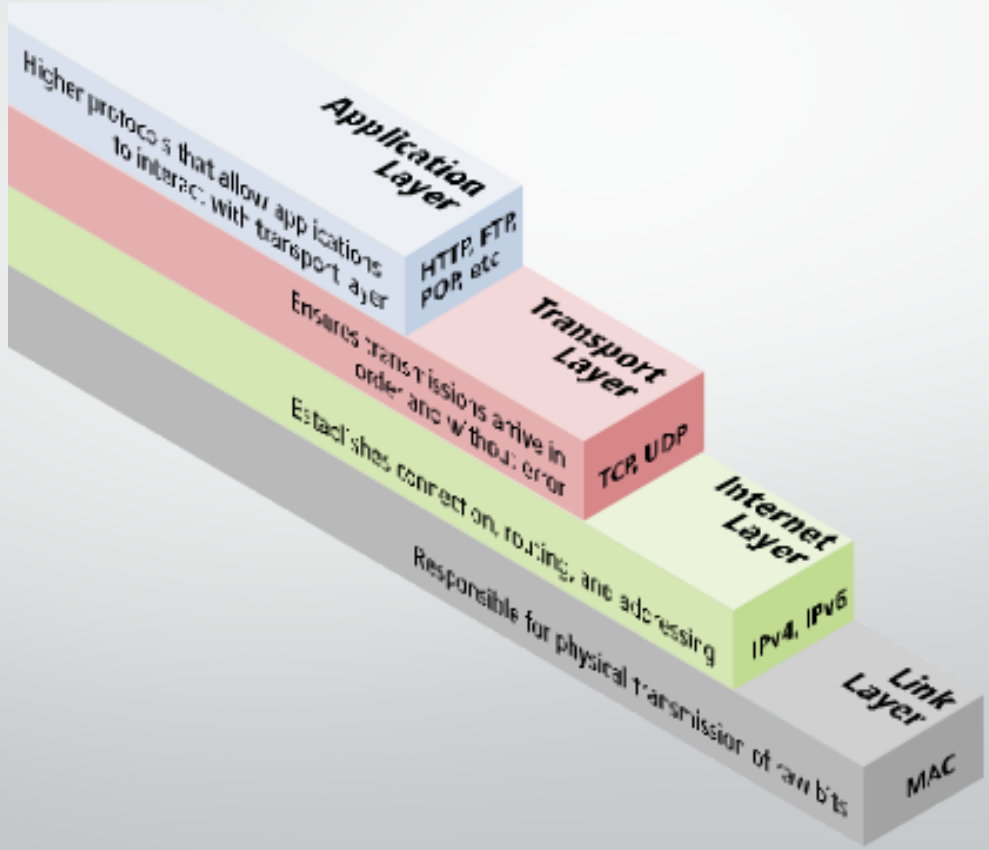
TCP/IP

- These protocols have been implemented in every operating system, and make fast web development possible.
- Networking is it’s own entire discipline.
- Web developer needs general awareness of what the suite of Internet protocols does

Internet Protocols

Layered Architecture:

look at it carefully you will get the idea of whole architecture.



Internet Protocols

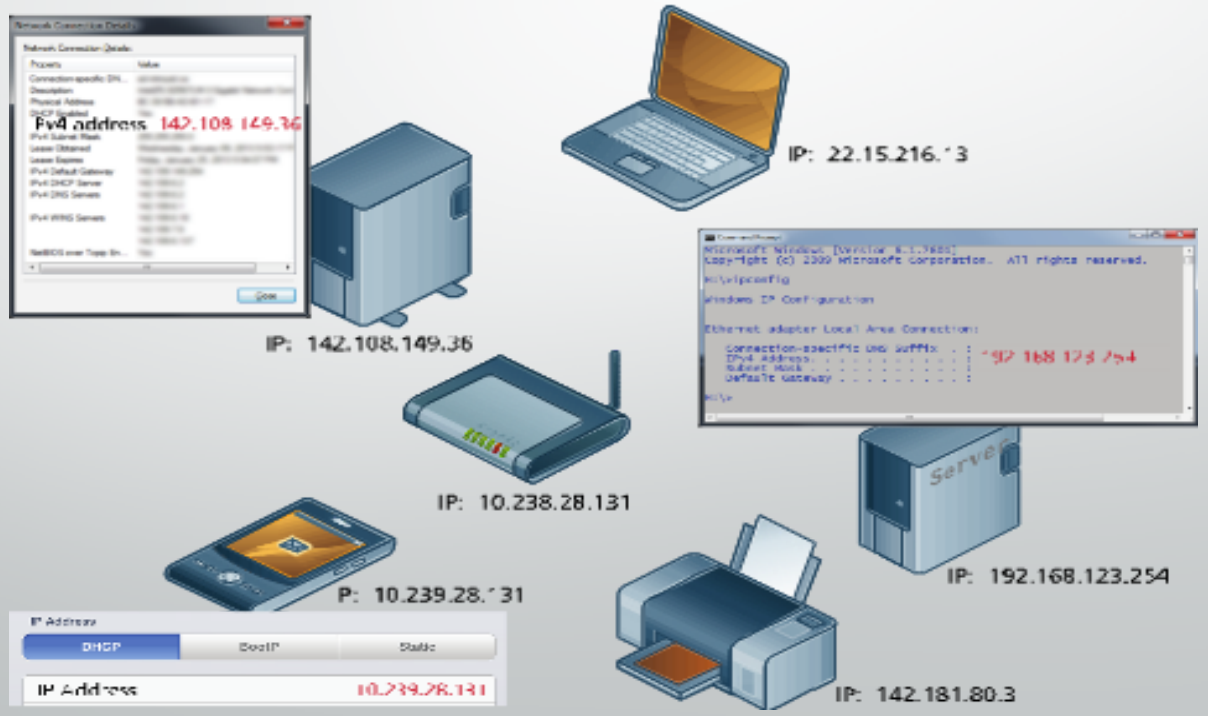
Link Layer

- Physical transmission of data across media (both wired and wireless) and establishing logical links.
- Handles issues like packet creation, transmission, reception, error detection, collisions, line sharing, and more.

Internet Protocols

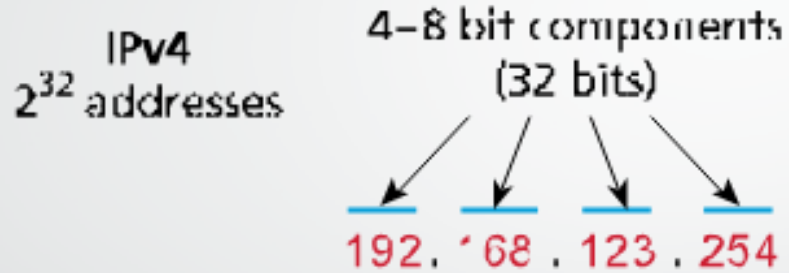
Internet Layer:

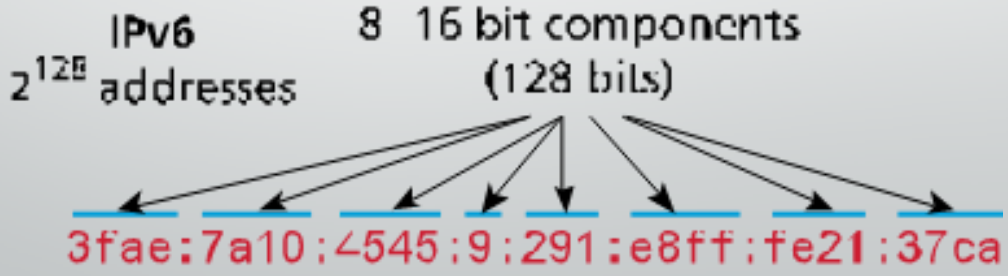
- The Internet layer provides “best effort” communication.
- Makes use of IP addresses



Internet Protocols

IP-Address:

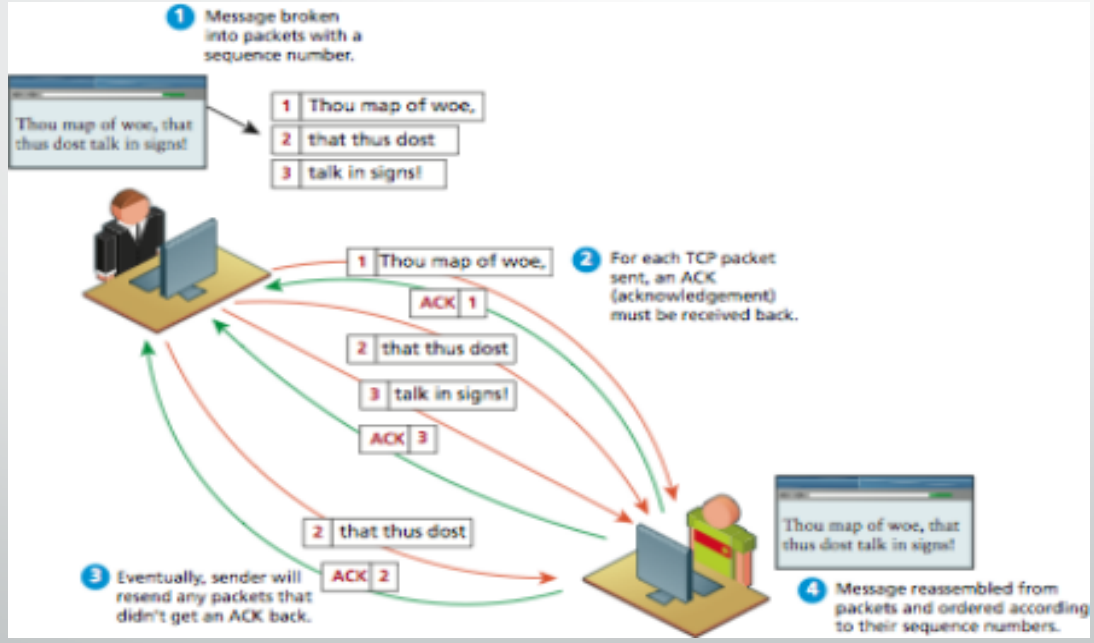




Internet Protocols

Transport Layer:

- Ensures transmissions arrive in order and without error



Internet Protocols

Application Layer Protocols:

- HTTP. The Hypertext Transfer Protocol is used for web communication.
- SSH. The Secure Shell Protocol allows remote command-line connections to servers.

Internet Protocols

Application Layer Protocols:

- FTP. The File Transfer Protocol is used for transferring files between computers.
- POP/IMAP/SMTP. Email-related protocols for transferring and storing email.
- DNS. The Domain Name System protocol used for resolving domain names to IP addresses.

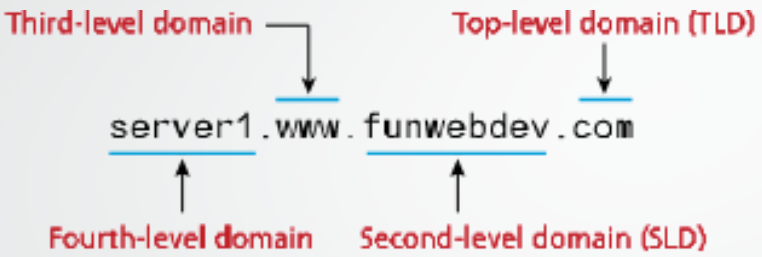
The End

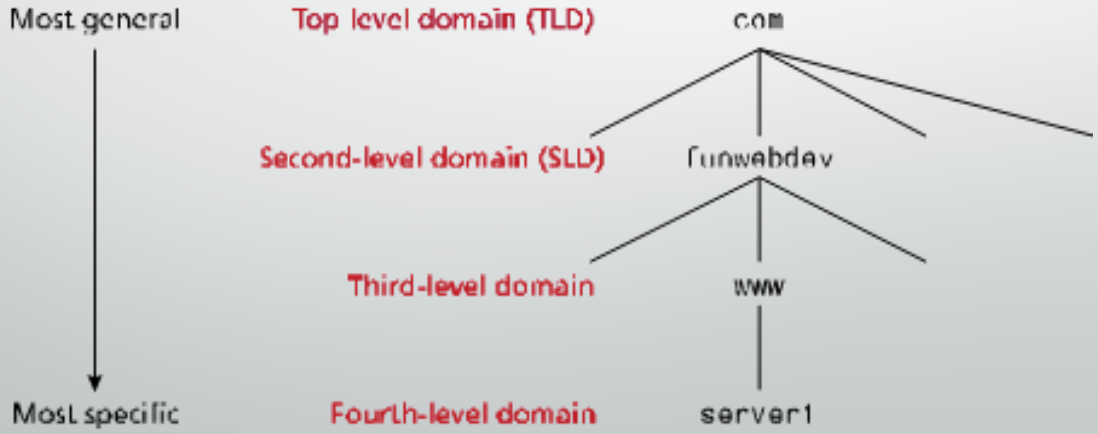
Web System and Tecnologies CS 521

TOPIC 08

Domain Name System

Name Levels in a Domain:





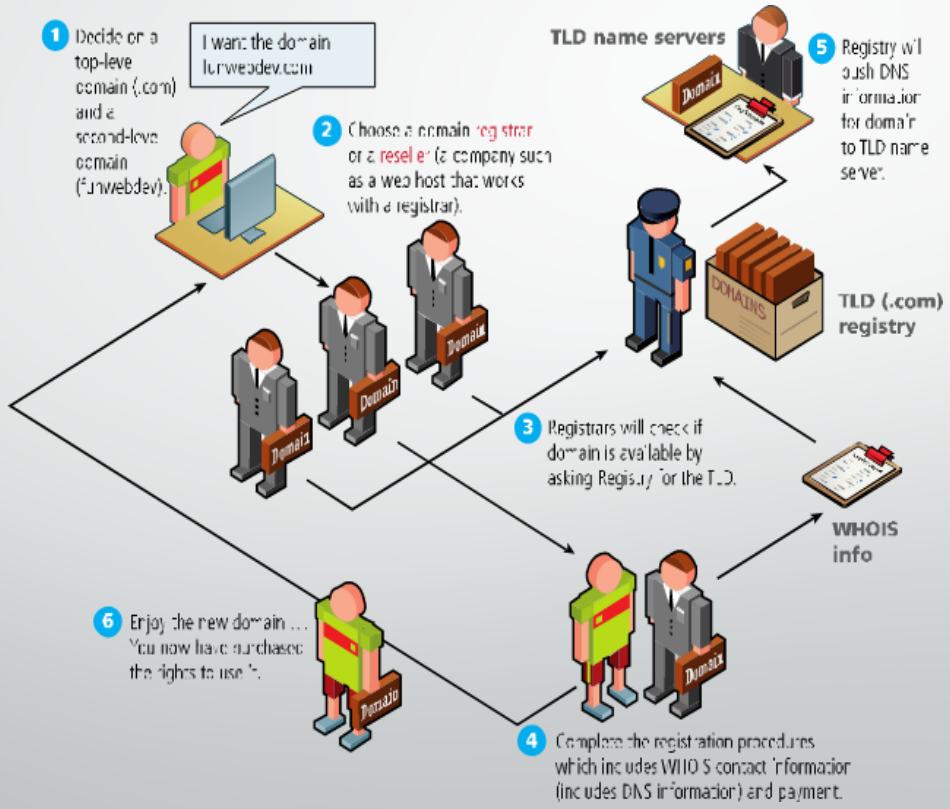
Domain Name System

Types of Top Level Domains:

- Generic top-level domain (gTLD)
  - Unrestricted. TLDs include .com , .net, .org, and .info.
  - Sponsored. TLDs including .gov , .mil, .edu, and others.
  - New TLDs.
- Country code top-level domain (ccTLD)
  - TLDs include .pk , .ca , .uk , .au. , …
  - Internationalized Domain Names

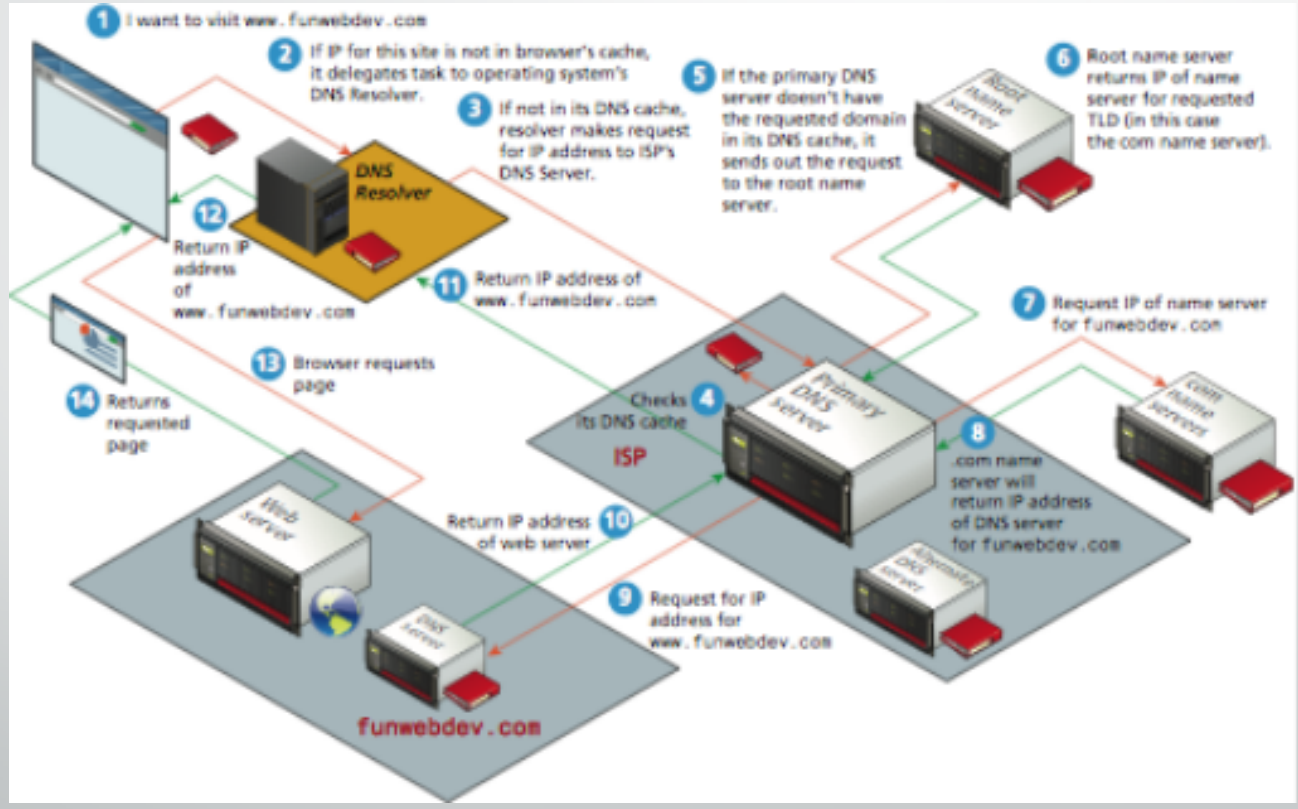
Domain Name System

Domain Registration:



Domain Name System

Address Resolution:



The End

Web System and Technologies CS 521

TOPIC 09

Uniform Resource Locators

Components of a URL:

- Protocol
- Domain
- Path
- Query String
- Fragment

which one is not component of url. component name was given .

Uniform Resource Locators

Protocols:

for file transfer which protocol is use.

- **ftp**://example.com/abc.txt → sends out an FTP request on port 21, while
- **http**://example.com/abc.txt → transmits an HTTP request on port 80

Uniform Resource Locators

Domain:

The domain identifies the server from which we are requesting resources.

- Since the DNS system is case insensitive, this part of the URL is case insensitive.
- Alternatively, an IP address can be used for the domain

Uniform Resource Locators

Port :

- The optional port attribute allows us to specify connections to ports other than the defaults
- Add a colon after the domain, then specify an integer port number.

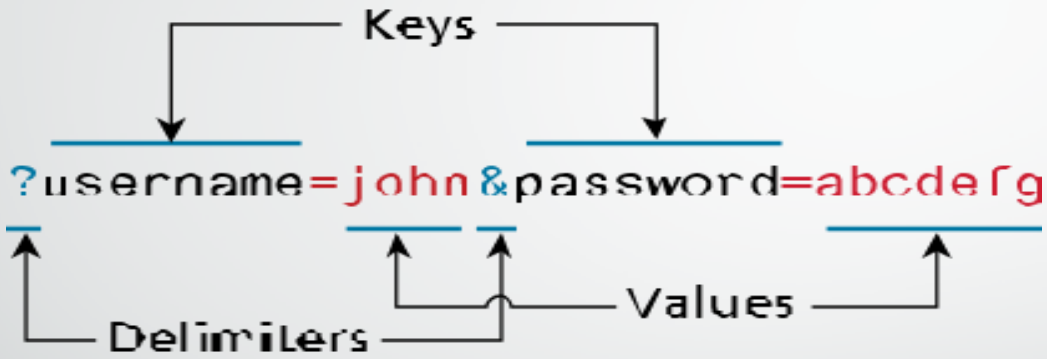
Uniform Resource Locators

Path :

- Familiar concept to anyone who has ever used a computer file system.
- The root of a web server corresponds to a folder somewhere on that server.
  - On many Linux servers that path is /var/www/html/
  - On Windows IIS machines it is often c:/inetpub/wwwroot/
- The path is optional
  - If not provided, server decides which file to send

Uniform Resource Locators

Query String :

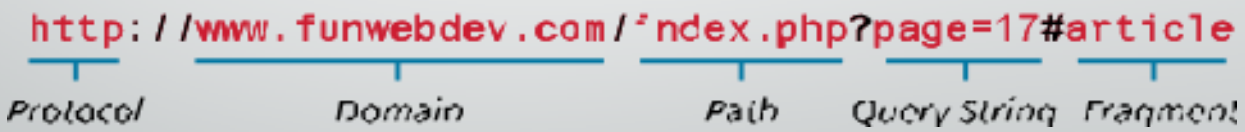


Uniform Resource Locators

Fragment :

- A way of requesting a portion of a page.

- Browsers will see the fragment in the URL, seek out the tag anchor in the HTML, and scroll the website to it



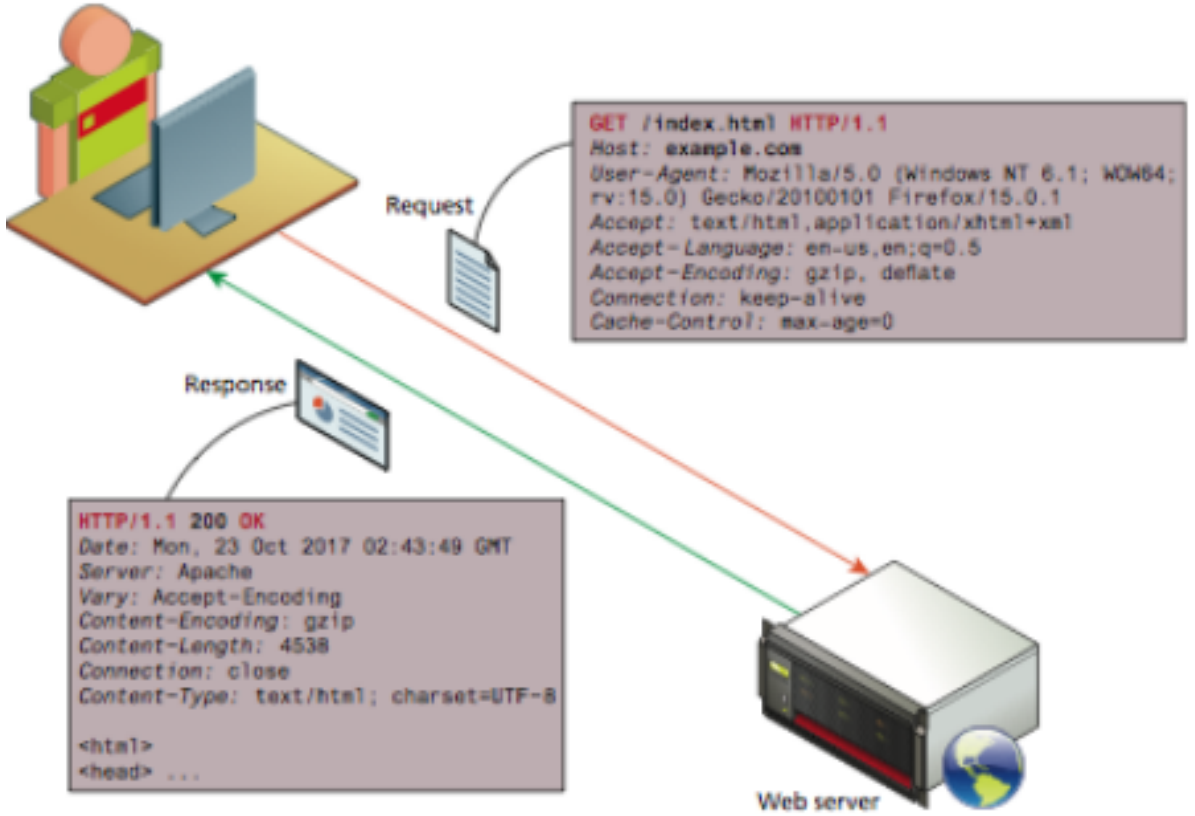
The End

Web System and Tecnologies CS 521

TOPIC 10

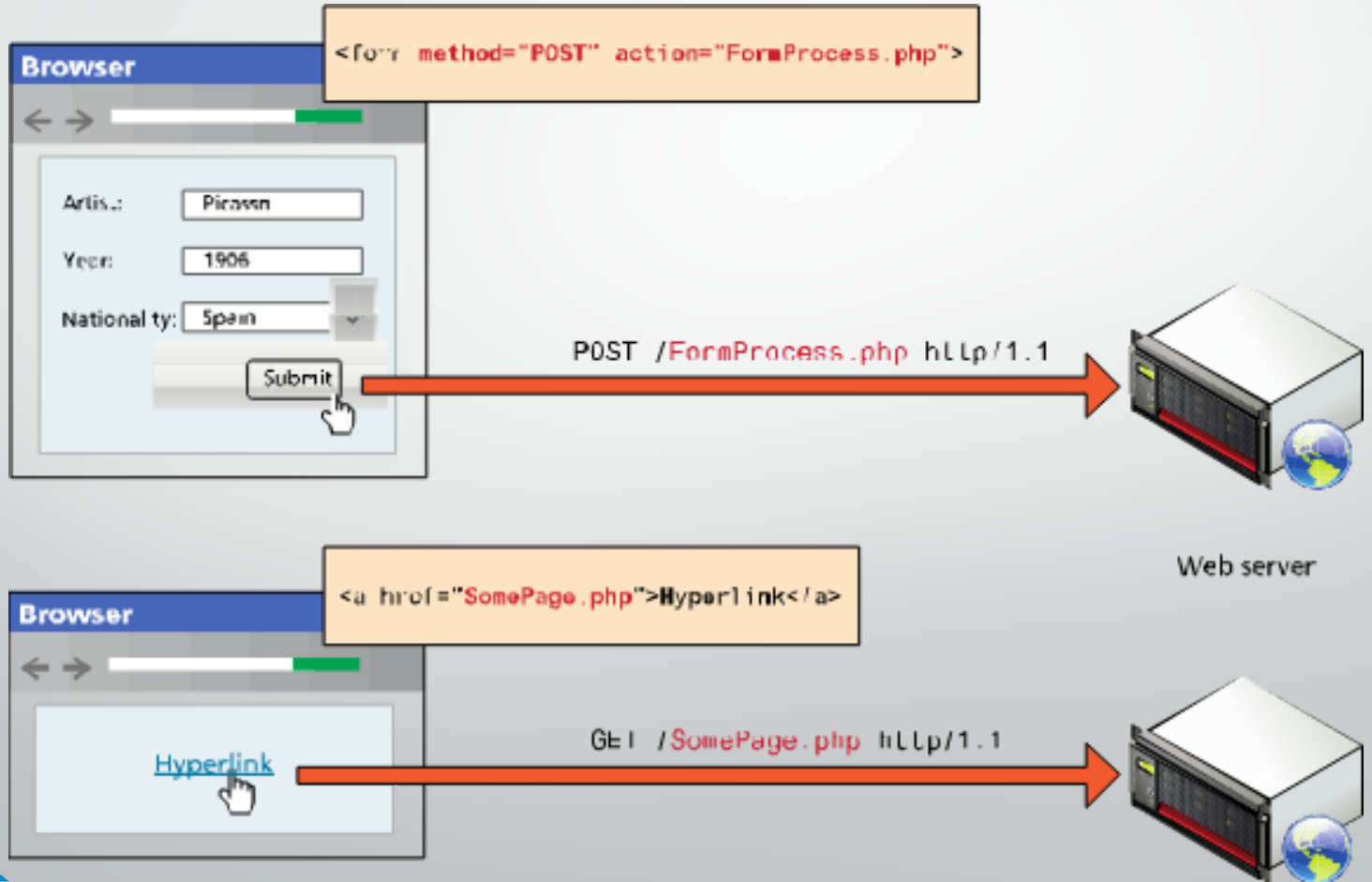
Hypertext Transfer Protocol (HTTP)

Headers



Hypertext Transfer Protocol (HTTP)

Request Methods



Hypertext Transfer Protocol (HTTP)

Response codes

- 200 codes are for successful responses,
- 300 are for redirection-related responses,
- 400 codes are client errors,
- 500 codes are server errors.

Hypertext Transfer Protocol (HTTP)

Response codes – Examples

- 200: OK
- 301: Moved Permanently
- 304: Not Modified
- 307: Temporary redirect
- 400: Bad Request
- 401: Unauthorized
- 404: Not found
- 414: Request URI too long
- 500: Internal server error

The End

Web System and Tecnologies CS 521

TOPIC 11

Introduction to Web Servers

Web servers

- A **web server** is nothing more than a computer that responds to HTTP requests.
- Real-world web servers are often more powerful than your own desktop computer
- Webservers must choose an **application stack** to run a website. This application stack will include an
  - Operating System
  - Web Server Software
  - A Database
  - Scripting Language for Dynamic Requests

Introduction to Web Servers

Application Stacks

- Most commonly used LAMP
  - **L**inux operating system,
  - **A**pache web server,
  - **M**ySQL database, and
  - **P**HP scripting language

- Other stacks include WAMP, WISA, MEAN, …

Introduction to Web Servers

Operating systems

- Linux
- Mac
- Windows

Introduction to Web Servers

Webserver Software

- Apache
- Nginx
- IIS

which one is webserver option given thy and apache was correct answer.

Introduction to Web Servers

Database Software

- MySQL
- PostgreSQL
- Sqlite
- Oracle
- IBM DB2
- Microsoft SQL Server
- MongoDB

Introduction to Web Servers

Scripting Languages

- PHP
- ASP.NET
- Python
- Node.js

The End

Web System and Tecnologies CS 521

TOPIC 12

Architecture of Web-based System

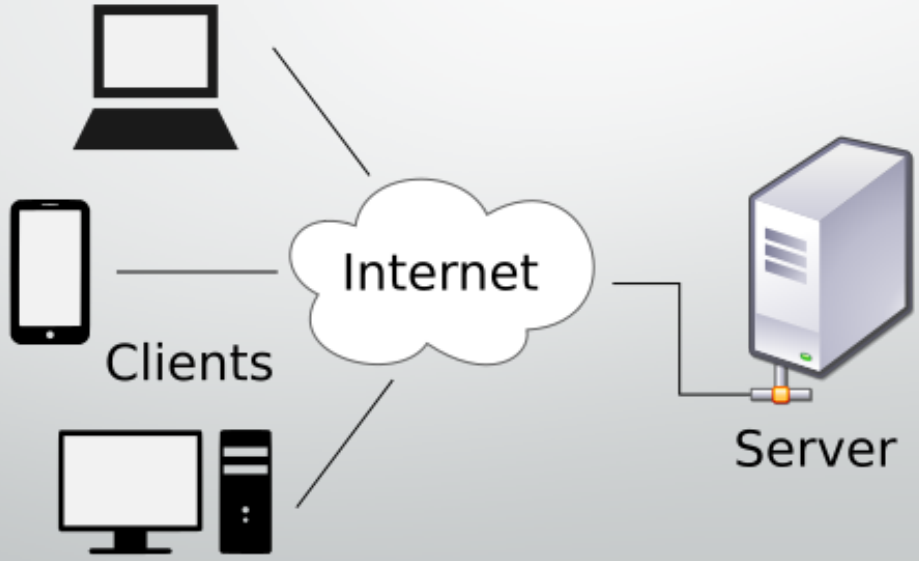
Multi-tier Architecture

- Web development can be realized at multiple tiers
- 2-tier
- 3-tier

Architecture of Web-based System

2-tier Architecture

- Tier 1 – Web Browser
- Tier 2 – Web server

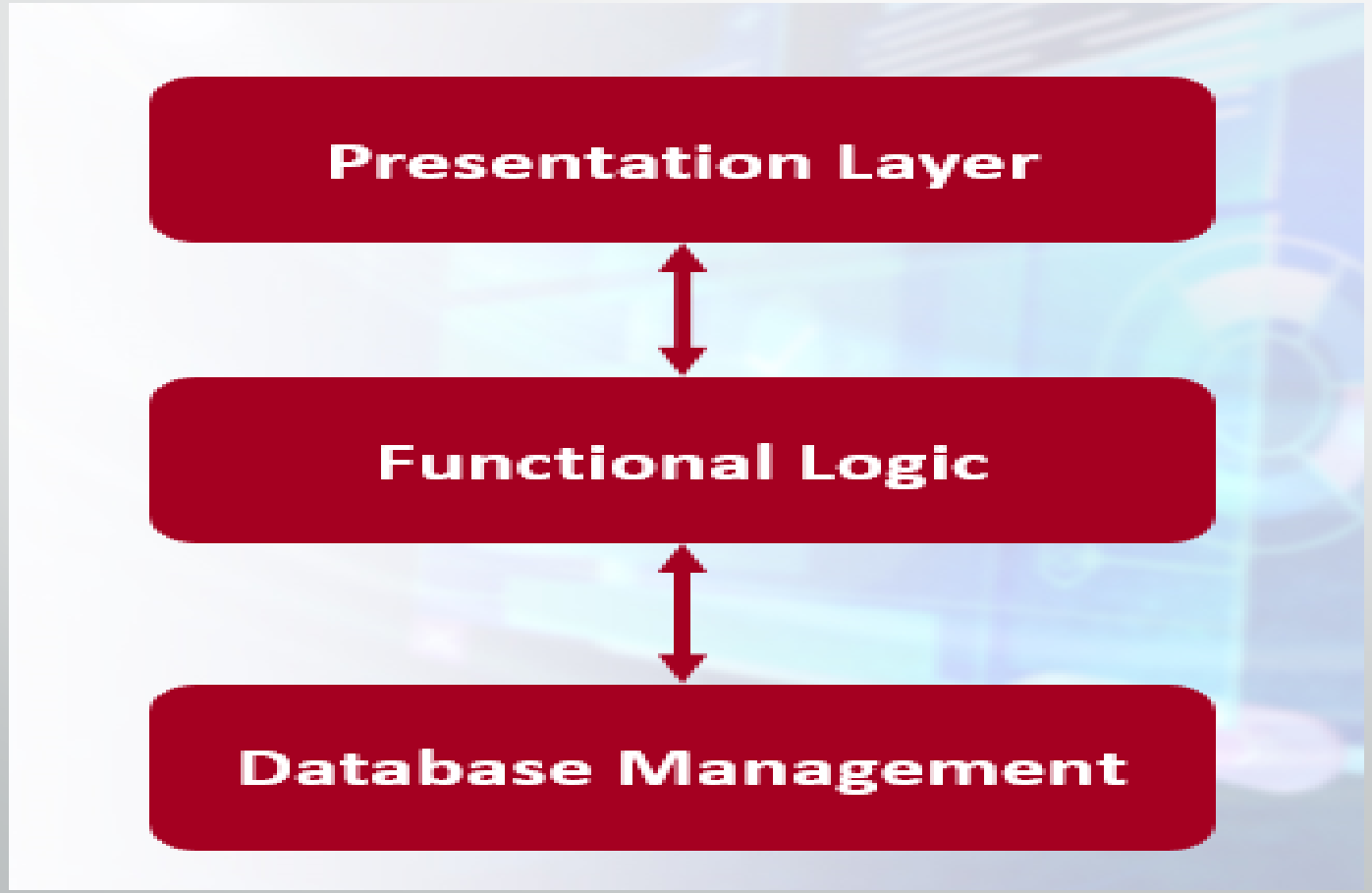


Credit: David Vignoni

https://en.wikipedia.org/wiki/File:Client-server-model.svg

Architecture of Web-based System

3-tier Architecture



The End

Web System and Tecnologies CS 521

TOPIC 13

Introduction to HTML

HTML

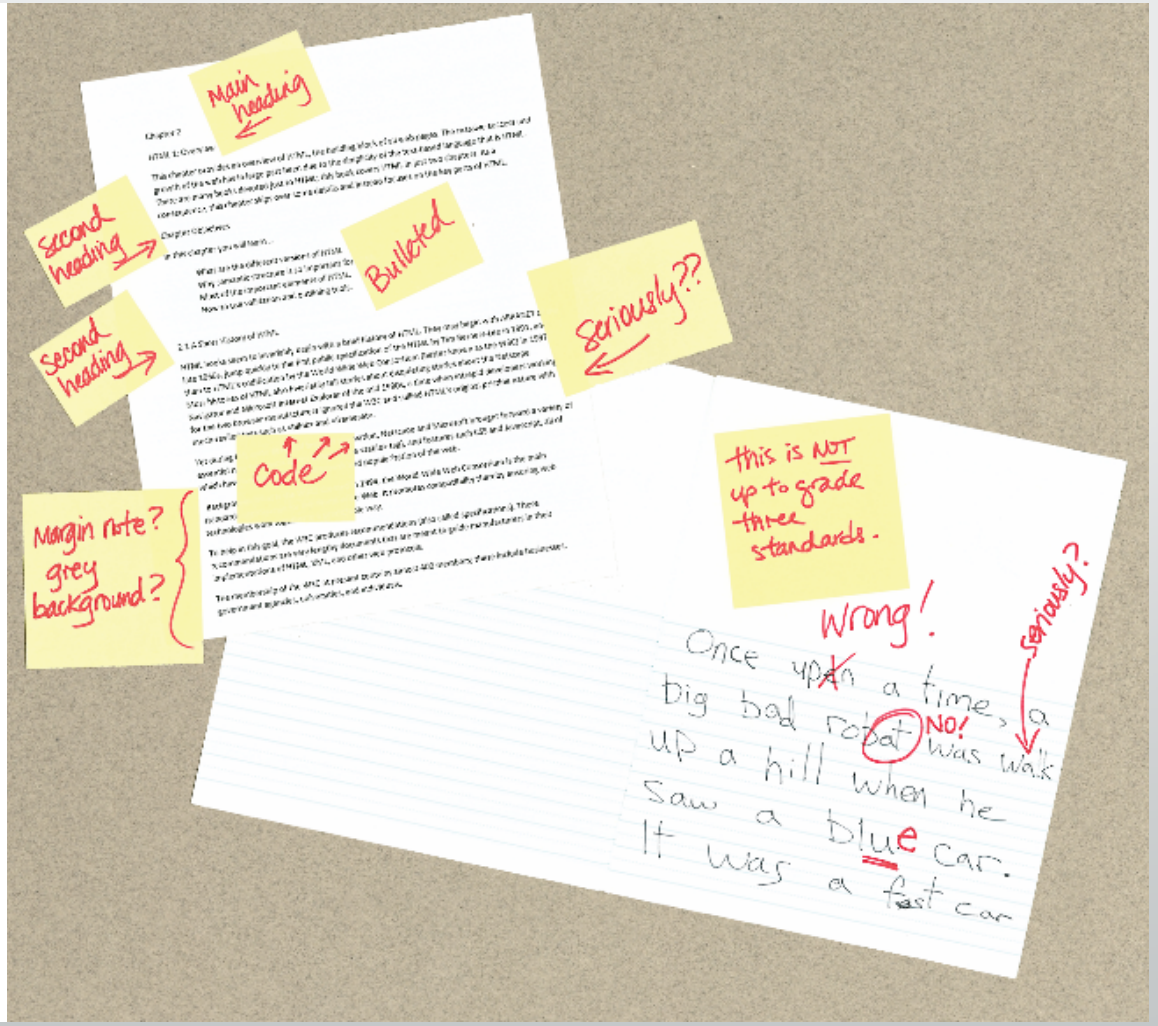
- Hypertext Transfer **Markup** Language

- **markup** is a way to indicate information about the content that is distinct from the content

- HTML has been through many versions and branches, the details of which might matter if you ever see old HTML code

Introduction to HTML

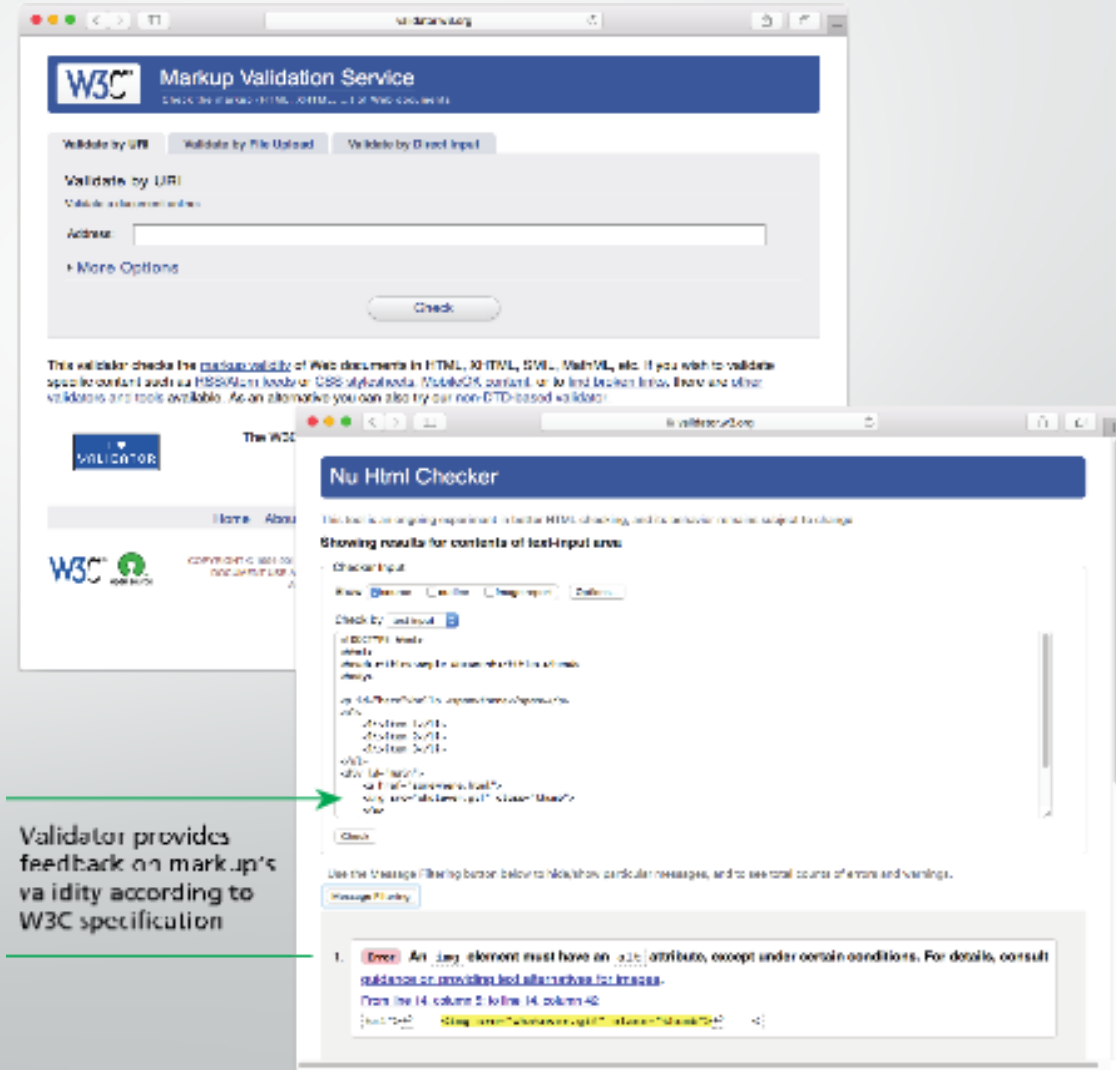
Markup



Introduction to HTML

(X)HTML and Validation

https://validator.w3.org/



Introduction to HTML

HTML 5

- Widely implemented in modern browsers
- The current W3C recommendation for web development

Introduction to HTML

Elements and Attributes

- **HTML documents** are composed of textual content and **HTML elements**
- **HTML element** encompasses
  - the **element name** within angle brackets (i.e., the **tag**) and
  - HTML elements can also contain **attributes, the content** within the tag.

Introduction to HTML

An Empty Element

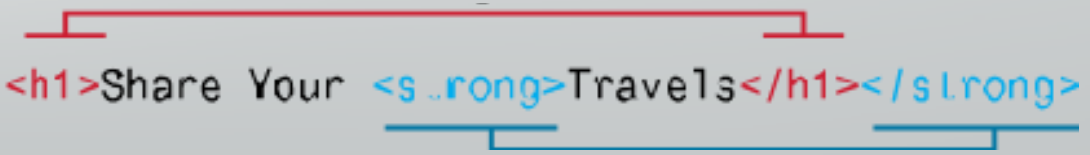
- An **empty element** does not contain any text content; instead, it is an instruction to the browser to do something.
- In XHTML, empty elements had to be terminated by a trailing slash.
- In HTML5, the trailing slash in empty elements is optional

Introduction to HTML

Nesting elements

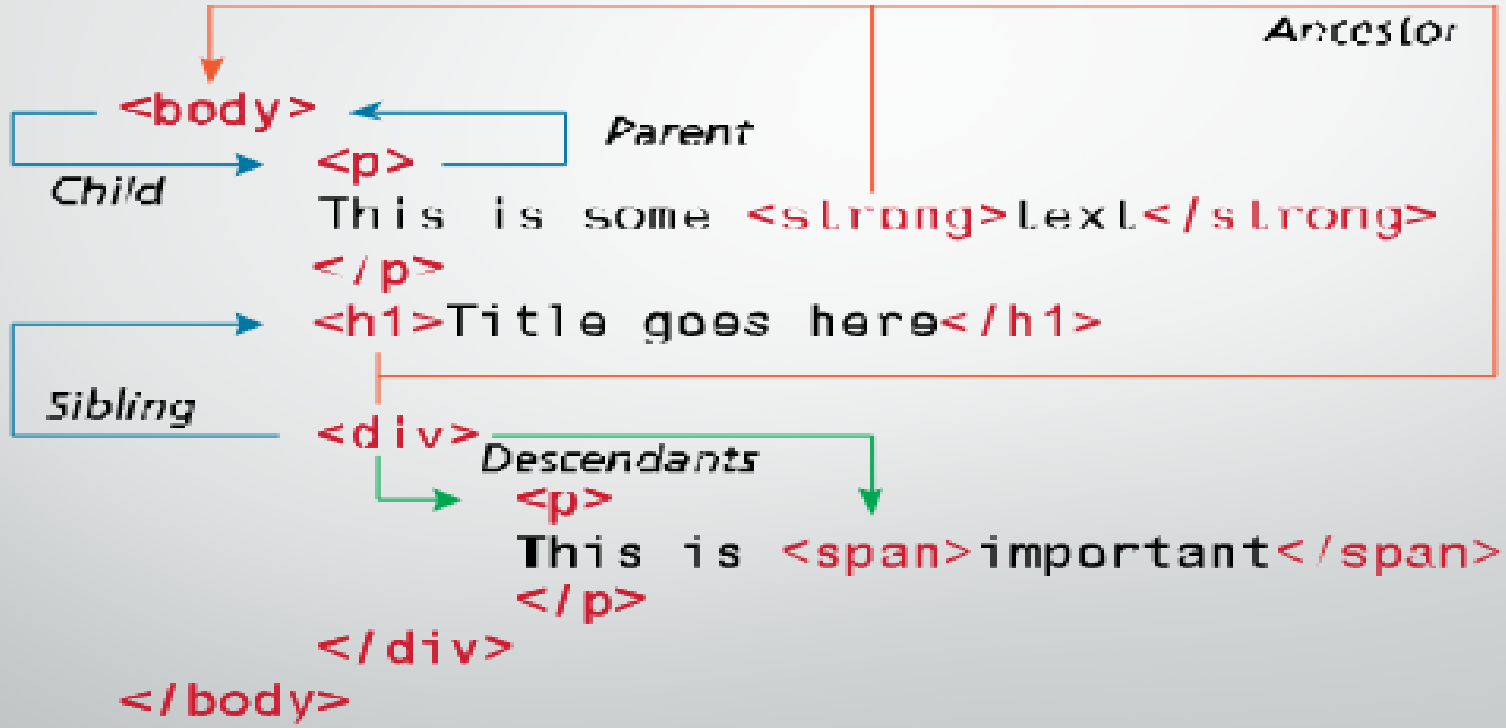
- Correct nesting
  - Nested element must be closed first

- Incorrect nesting
  - Outer element is closed first



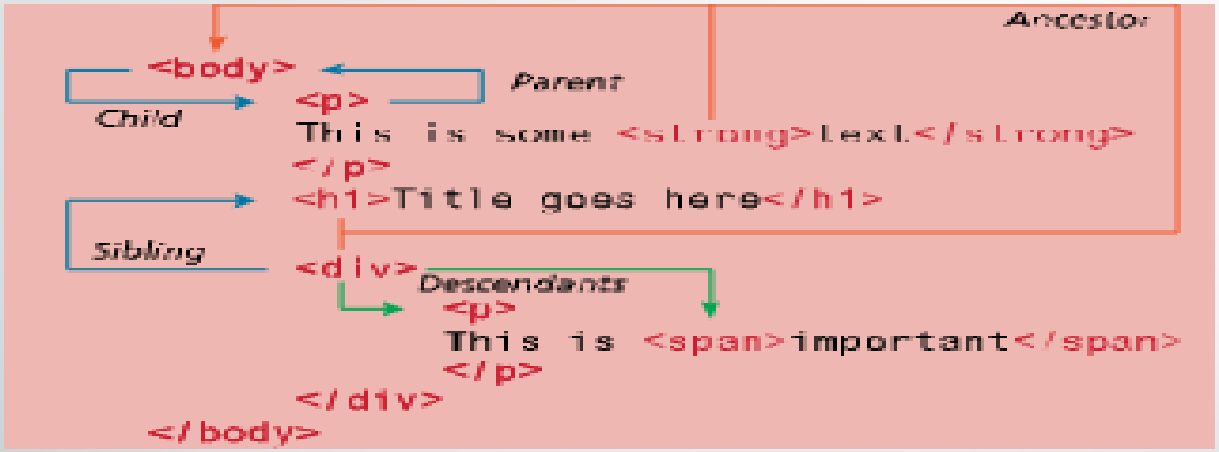
Introduction to HTML

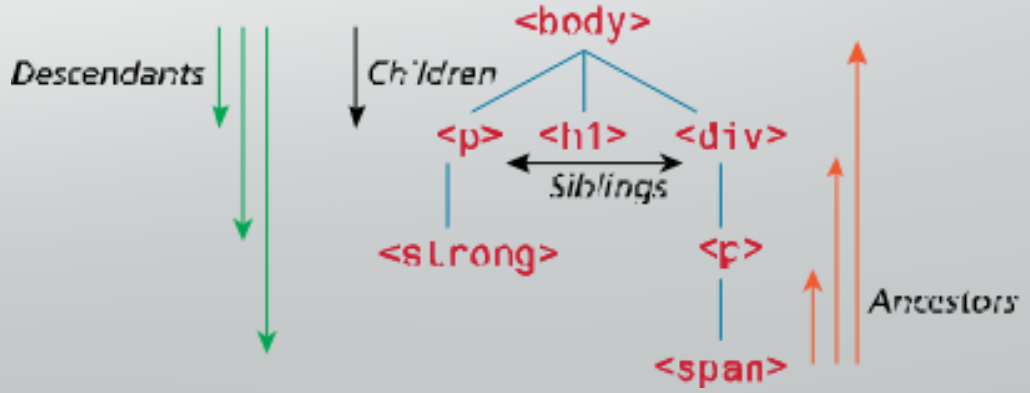
Nesting elements



Introduction to HTML

Tree of Nested Elements





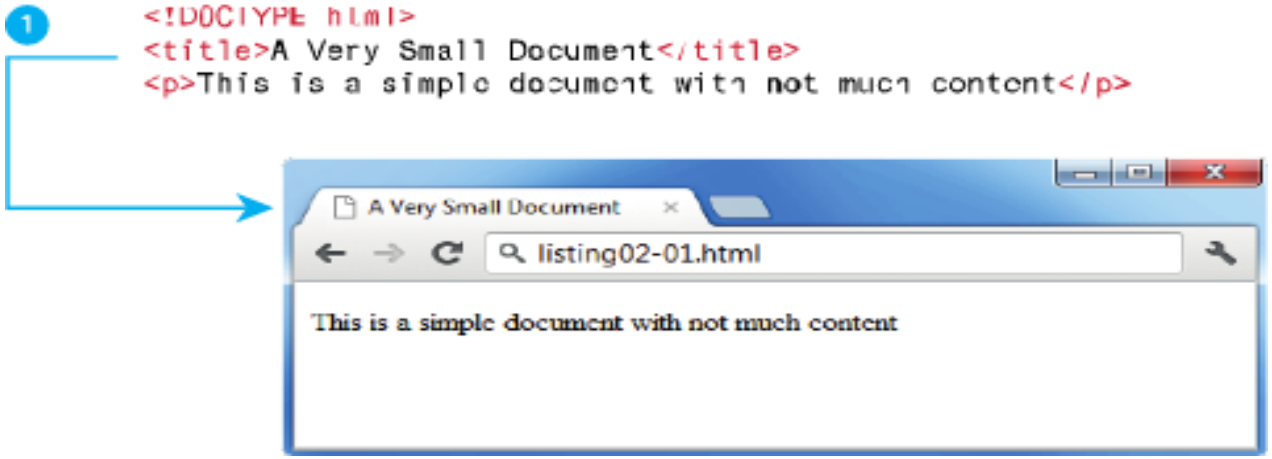
The End

Web System and Tecnologies CS 521

TOPIC 14

Common HTML Tags

A Simple Example



Common HTML Tags

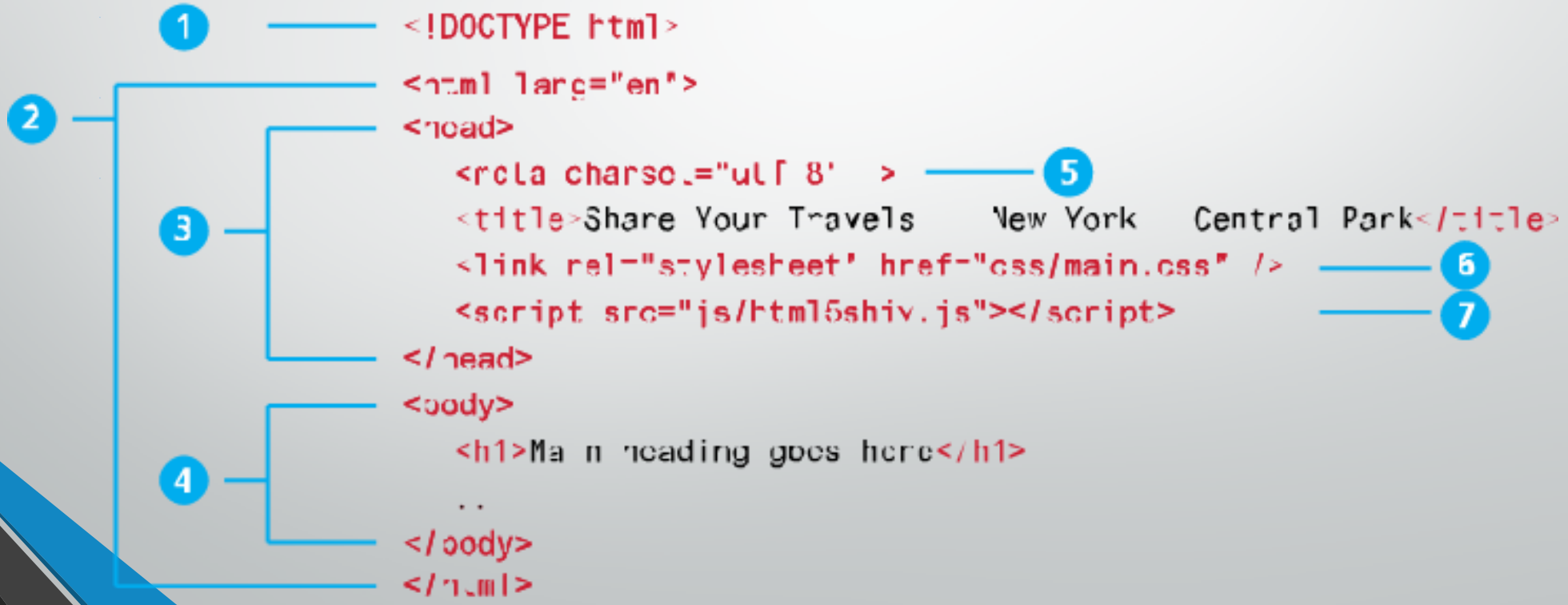
DOCTYPE

- DOCTYPE Short for **Document Type Definition** tells the browser what type of document it is about to process
- Written at the start of the HTML document

<!DOCTYPE html>

Common HTML Tags

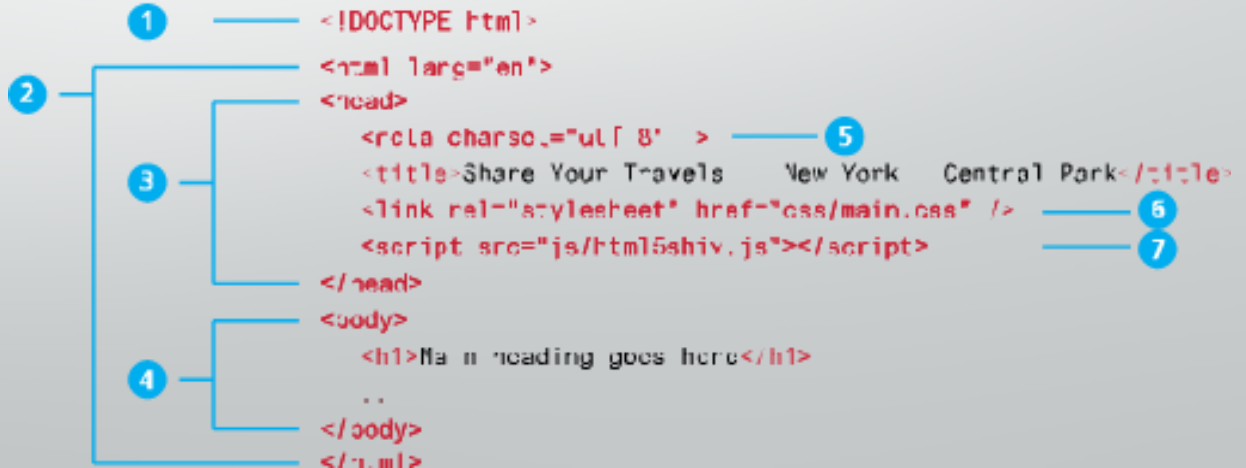
Another Example



Common HTML Tags

Head and Body Tags

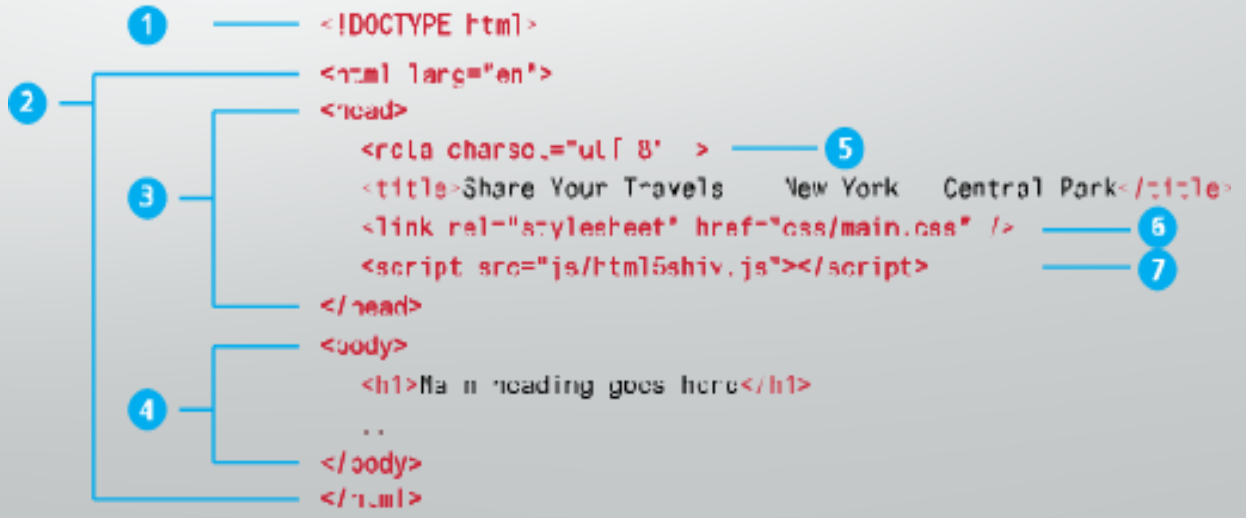
- HTML5 does not require the use of the **<html> , <head>** , and **<body>** elements (most developers still use them)
- **<html>** contains all the other HTML elements in the document (Item 2)
- **<head>** contains descriptive elements about the document, such (title, style sheets, JavaScript files etc.) (Item 3)
- **<body>** contains content to be displayed by the browser (Item 4)



Common HTML Tags

Meta, Link, and Script Tags

- The **<meta>** element (Item 5) declares that the character encoding for the document is UTF-8.
- Item 6 specifies an external CSS style sheet file with **<link>** that is used with this document.
- Item 7 references an external JavaScript file using **<script>**



Common HTML Tags

Another Example

```
<body>
    <h1>Share Your Travels</h1>
    <h2>New York - Central Park</h2>
    <p>Photo by Randy Connolly</p>
    <p>This photo of Conservatory Pond in
        <a href="http://www.centralpark.com/">Central Park</a>
        New York City was taken on October 22, 2016 with a
        <strong>Canon EOS 30D</strong> camera.
    </p>
    <img src="images/central-park.jpg" alt="Central Park" />

    <h3>Reviews</h3>
    <div>
        <p>By Ricardo on <time>2016-05-23</time></p>
        <p>Easy on the HDR buddy.</p>
    </div>
    <hr>
    <div>
        <p>By Susan on <time>2016-11-18</time></p>
        <p>I love Central Park.</p>
    </div>

    <p><small>Copyright &copy; 2017 Share Your Travels</small></p>
</body>
```

1 2 3 4 5 6 7 8 9 10

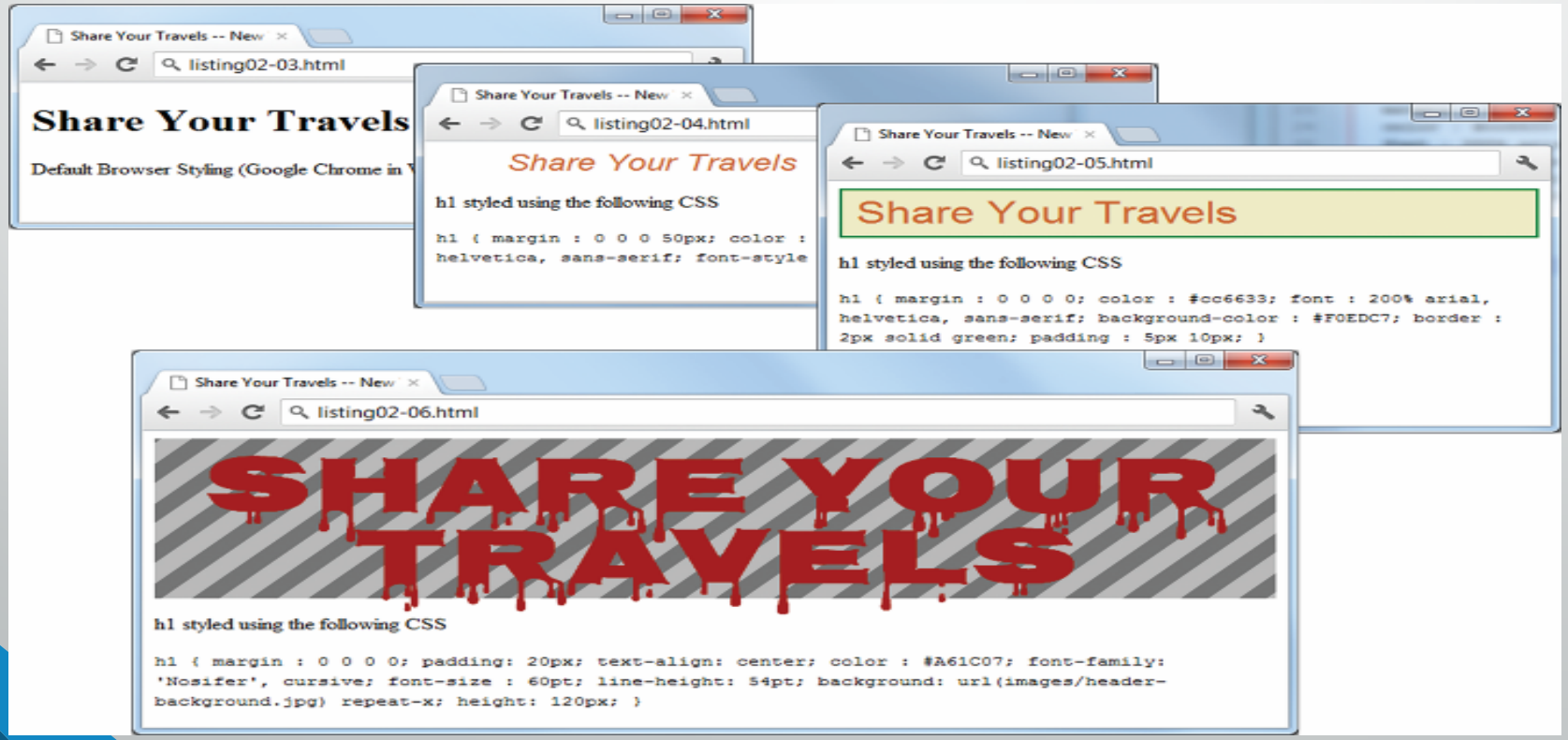
Common HTML Tags

Rendered Document

Common HTML Tags

HTML is for structure!

- CSS is used for styling



Common HTML Tags

how many tags are in the heading

Headings

- HTML provides six levels of headings (h1 - h6)
- Headings are also used by the browser to create a document outline for the page
- Don't use different headings for setting font size (will be covered in CSS)

Common HTML Tags

Paragraphs and Divisions

for logical devision what we use correct answer div.

- **<p>** tag is a container for text and other HTML elements
- ** ** also a container element and is used to create a logical grouping of content

Common HTML Tags

Inline Text Elements

- Do not disrupt the flow
  - <a>
  - <abbr>
  -
  - <cite>
  - <code>
  -
  - <mark>
  - <small>
  -
  -
  - <time>
  - …

… <inline element> …

Common HTML Tags

Block Elements

- Disrupt the existing flow and take their own space
  -
  - <h1>-<h6>
  - <hr>
  -
  -
  -
  - <p>
  - <table>
  - …

…

<block element>

…

Common HTML Tags

Images

Specifies the URL of the image to display (note: uses standard relative referencing).

Text in title attribute will be displayed in a pop up tool tip when user moves mouse over image.

```
<img src="images/central park.jpg" alt="Central Park" title="Central Park" width="80" height="40" />
```

Text in alt attribute provides a brief description of image's content for users who are unable to see it.

Specifies the width and height of image in pixels

Common HTML Tags

Character Entities

- These are special characters for symbols for which there is either no easy way to type them via a keyboard
- or which have a reserved meaning in HTML (like "<")

Entity	Description
	Nonbreakable space
&lt;	<
&gt;	>
&copy;	©
&trade;	™

The End

Web Systems and Technologies

TOPIC 15

Links

Links

Links (Anchor tags)

- Links are an essential feature of all web pages
- Links use the <a> element (the “a” stands for anchor).

Links

Examples

Link to external site

```
<a href="http://www.centralpark.com">Central Park</a>
```

Link to resource on external site

```
<a href="http://www.centralpark.com/logo.gif">Central Park</a>
```

Link to another page on same site as this page

```
<a href="index.html">Home</a>
```

Link to another place on the same page

```
<a href="#top">Go to Top of Document</a>
...
<a name="top">
```

Defines anchor for a link to another place on same page

Links

Examples

Link to specific place on another page

```
<a href="productX.html#reviews">Reviews for product X</a>
```

Link to email

```
<a href="mailto:person@somewhere.com">Someone</a>
```

Link to JavaScript function

```
<a href="javascript:OpenAnnoyingPopup();">See This</a>
```

Link to telephone (automatically dials the number when user clicks on it using a smartphone browser)

```
<a href="tel:+18009220579">Call toll free (800) 922-0579</a>
```

Links

Relative Referencing

Relative Link Type	Example
Same directory	<a href="example.html">
Child Directory	<a href="images/logo.gif">
Grandchild/Descendant Directory	<a href="css/images/background.gif">
Parent/Ancestor Directory	<a href="../about.html"> <a href="../../about.html”>
Sibling Directory	<a href="../images/about.html">
Root Reference	<a href="/images/background.gif">

<a href="about.html> which type of link is this .

Web Systems and Technologies

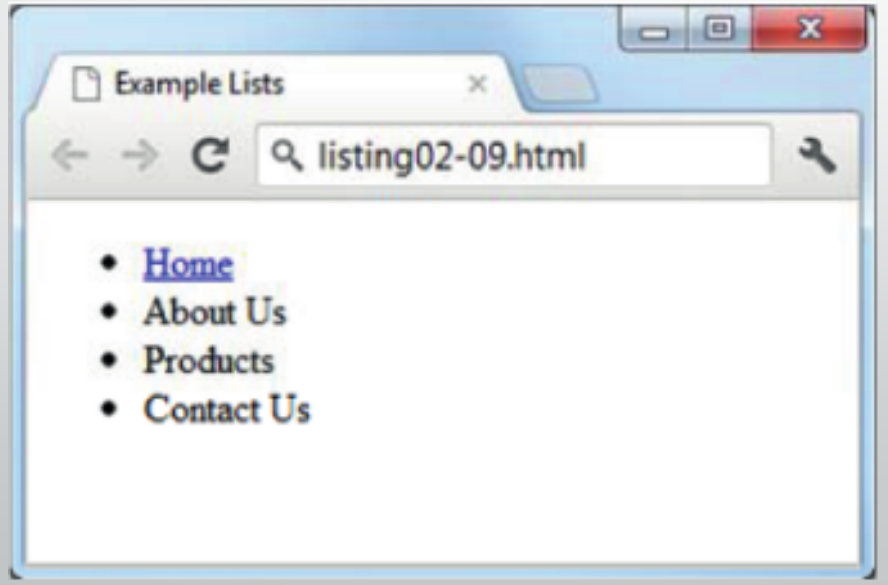
TOPIC 16

Lists

Lists

Unordered Lists

```
<ul>
    <li <a href 'index.html">Home</a></li>
    <li About Us< li
    <li Products< li
    <li Contact Us</li
</ul>
```



Lists

Ordered Lists

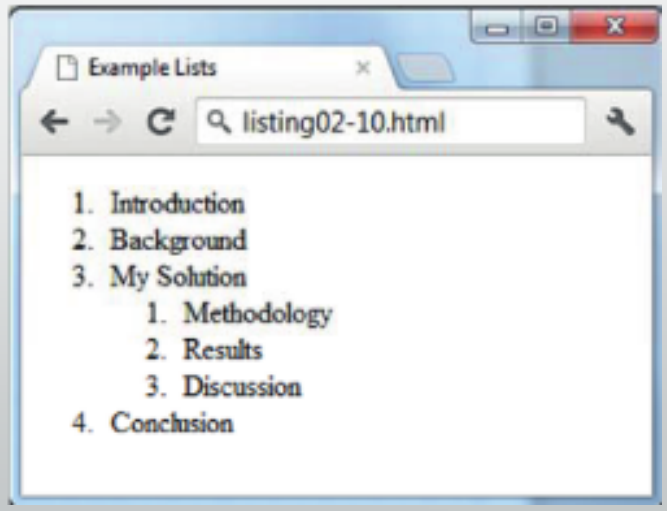
```
<ol>
    <li>Introduction</li>
    <li>Background</li>
    <li>My Solution</li>
    <li>Conclusion</li>
</ol>
```

1. Introduction
2. Background
3. My Solution
4. Conclusion

Lists

Nested Lists

```
<ol>
   <li>Introduction</li>
   <li>Background</li>
   <li>My Solution</li>
   <li>
      <ol>
         <li>Methodology< li>
         <li>Results< li>
         <li>Discussion< li>
      < ol>
   < li>
   <li>Conclusion</li>
< ol>
```



Lists

Description Lists

- Used for definitions and descriptions

```
<dl>
    <dt>HTML</dt>
    <dd>Hyper Text Markup Language</dd>
    <dt>CSS</dt>
    <dd>Cascading Style Sheet</dd>
</dl>
```

HTML
    Hyper Text Markup Language
CSS
    Cascading Style Sheet

END

Web Systems and Technologies

TOPIC 17

long question.

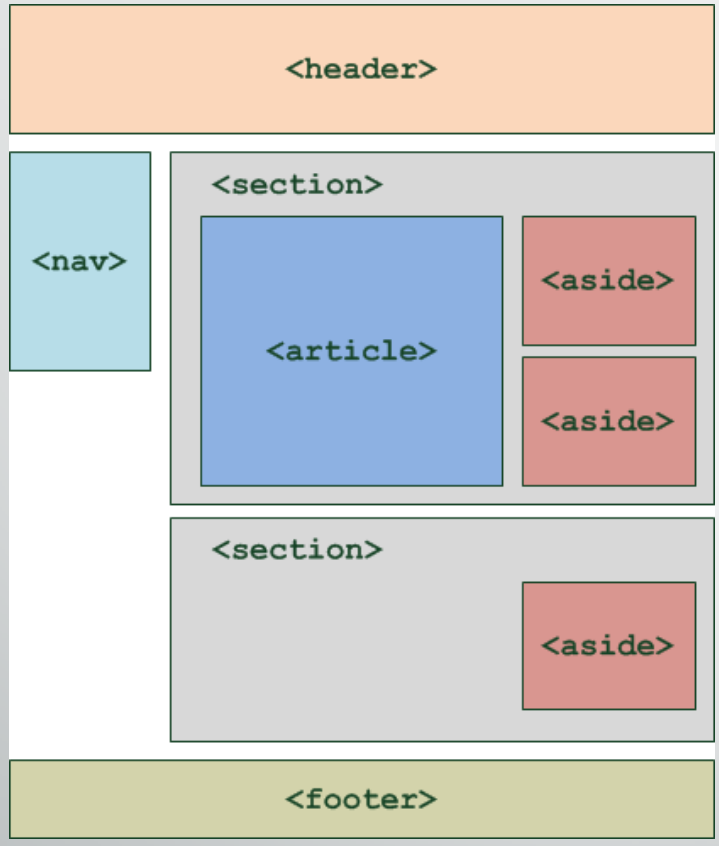
Semantics in HTML 5

Semantics in HTML 5

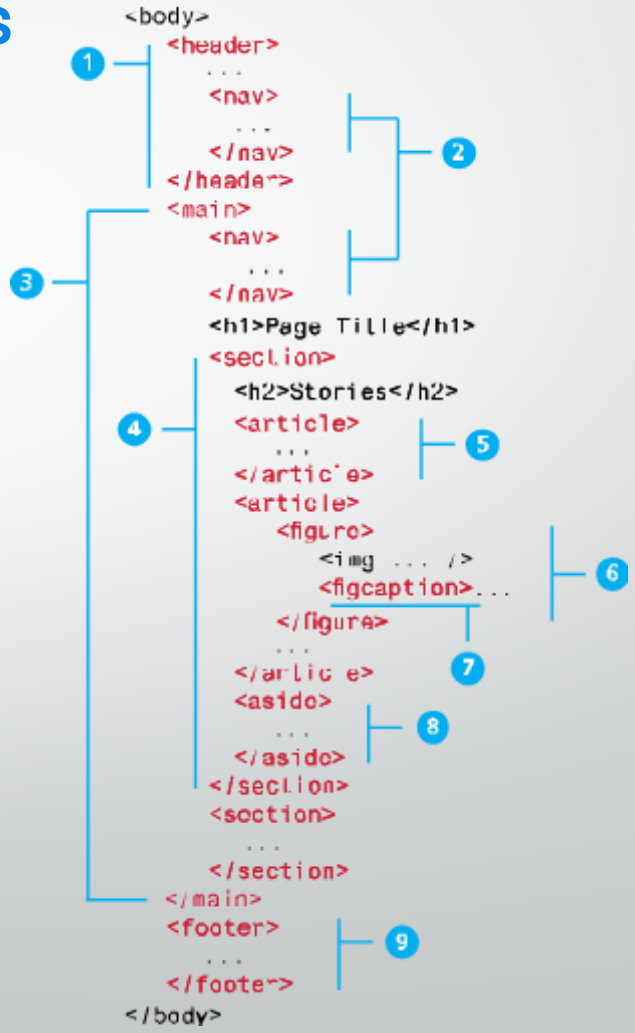
Semantic Markup

- Semantic tags are used for organizing contents and labeling contents (adding meaning)
- Do not change the structure of the document
- Make HTML code
  - maintainable
  - accessible

Semantics in HTML 5

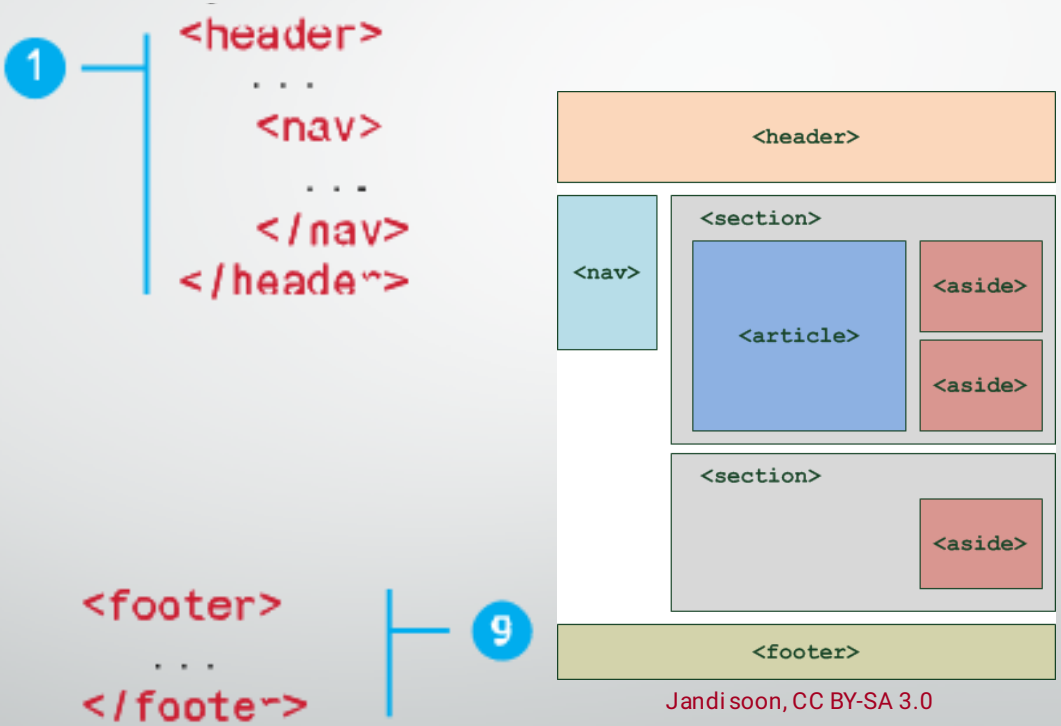


Semantic tags



Semantics in HTML 5

Header and Footer



Semantics in HTML 5

Header

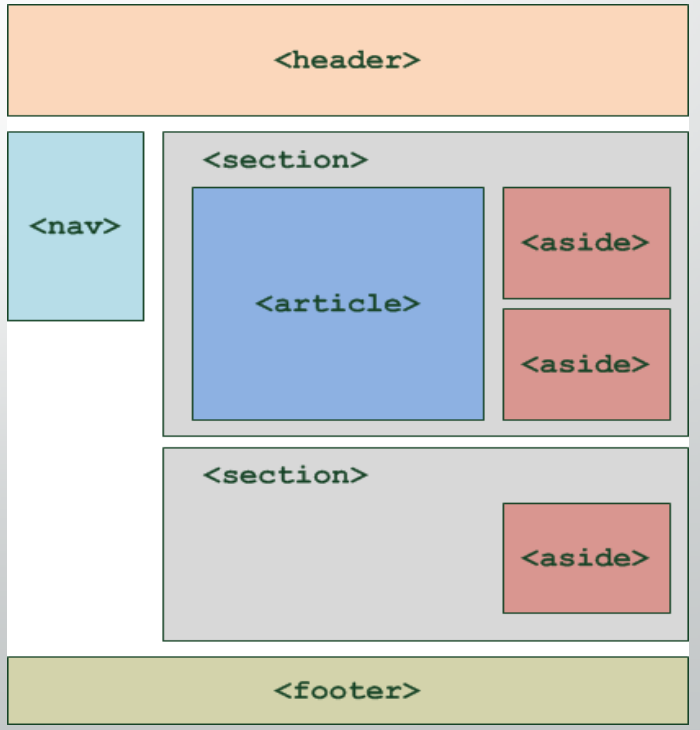
- Intended to usually contain the section's heading (an h1– h6 element), but not required
- The header element can also be used to wrap a section's table of contents, a search form, or any relevant logos.

```
<header>
    <img src="logo.gif" alt="logo" />
    <h1>Fundamentals of Web Development</h1>
    ...
</header>
```

Semantics in HTML 5

Navigation (nav)

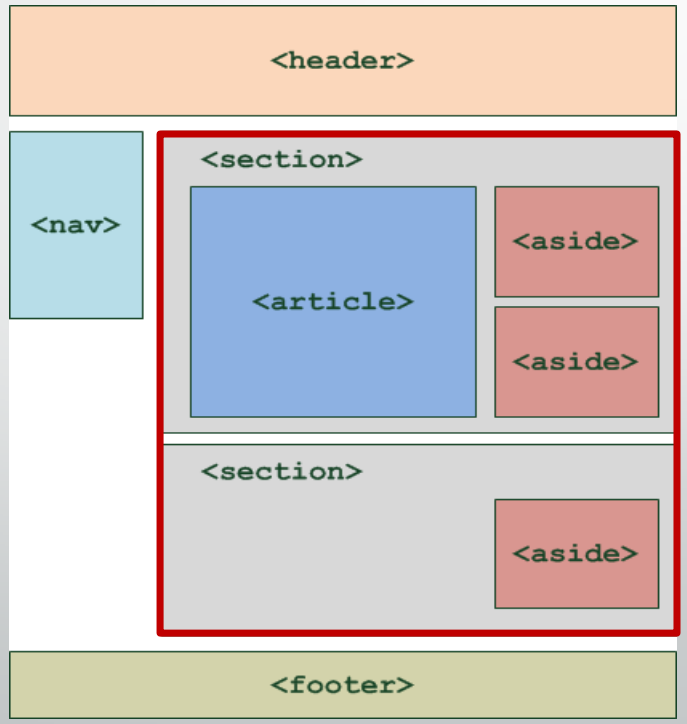
```
<header>
    <img src="logo.gif" alt="logo" />
    <h1>Fundamentals of Web Development</h1>
    <nav>
        <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="about.html">About Us</a></li>
            <li><a href="browse.html">Browse</a></li>
        </ul>
    </nav>
</header>
```



Semantics in HTML 5

Main

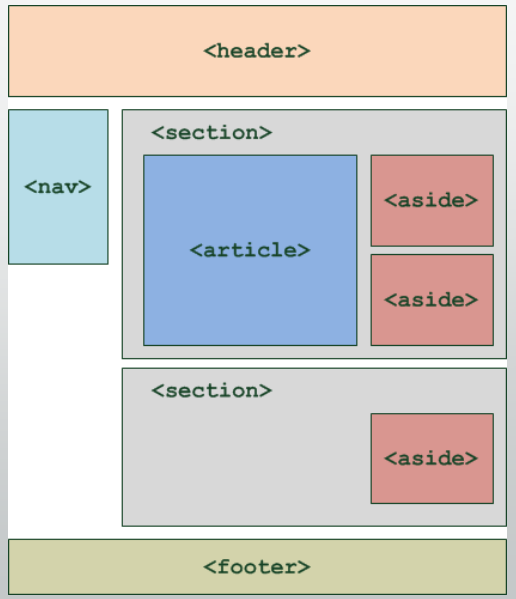
- **<main>** is meant to contain the main unique content of the document.



Semantics in HTML 5

Section and Article

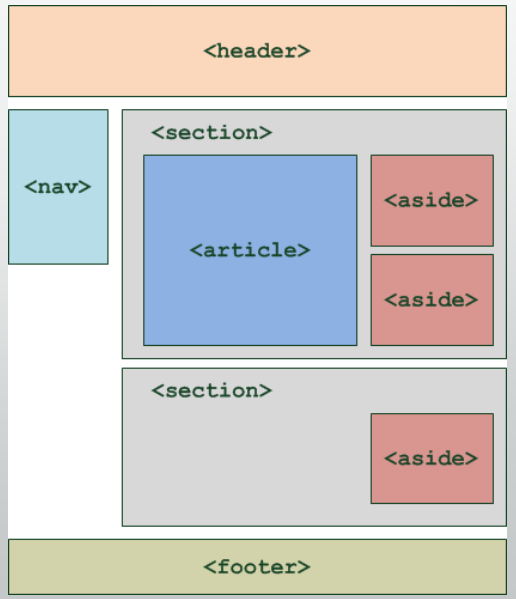
- **<section>** is a much broader element, while the
- **<article>** element is to be used for blocks of content that could potentially be read or consumed independently of the other content on the page



Semantics in HTML 5

Aside

- The **<aside>** element can be used for sidebars, pull quotes, groups of advertising images, or any other grouping of nonessential elements



Semantics in HTML 5

Figures and Figure Captions

Figure could be moved to a different location in document.

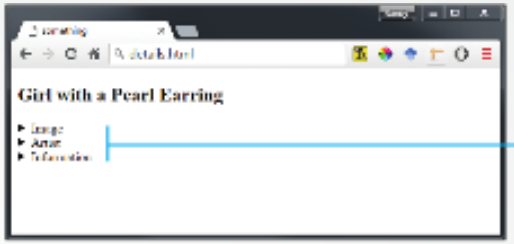
...

But it has to exist in the document (i.e., the figure isn't optional).

```
<p>This photo was taken on October 22, 2011 with a Canon EOS 30D camera </p>
<figure>
  <img src="images/central-park.jpg" alt="Central Park"  /><br/>
  <figcaption>Conservatory Pond in Central Park</figcaption>
</figure>
<p>
It was a wonderfully beautiful autumn Sunday, with strong sunlight and
expressive clouds. I was very fortunate that my one day in New York was
blessed with such weather!
</p>
```

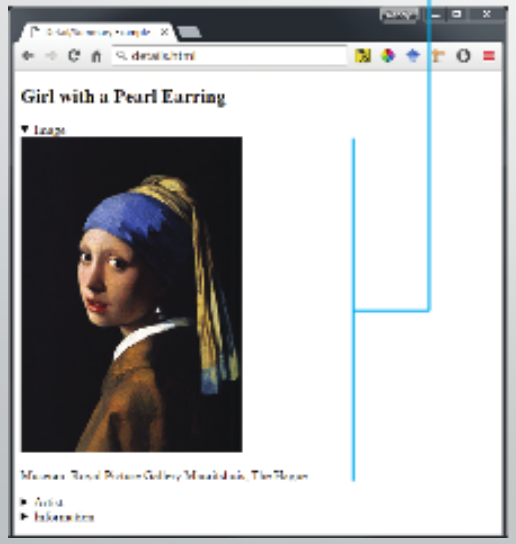
Semantics in HTML 5

Details and Summary




```
<body>
   <h2>Girl with a Pear" Earring</h2>
   <details>
     <summary>Image</summary>
     <img src="images/106020.jpg"><br>
     <p>Museum: Roya" Picture Ga"lery Mauritshuis ...
   </details>
   <details>
     <summary>Artist</summary>
     <p><strong>Jan Vermeer</strong> was a Dutch ...
   </details>
   <details>
     <summary>Information</summary>
     <p>
        Date: 1665<br>
        Medium: Oil on Canvas
     </p>
  </details>
</body>
```

Clicking on the summary label reveals the rest of the content with the <details> container



Web Systems and Technologies
TOPIC 18

Tools Insight

Tools Insights

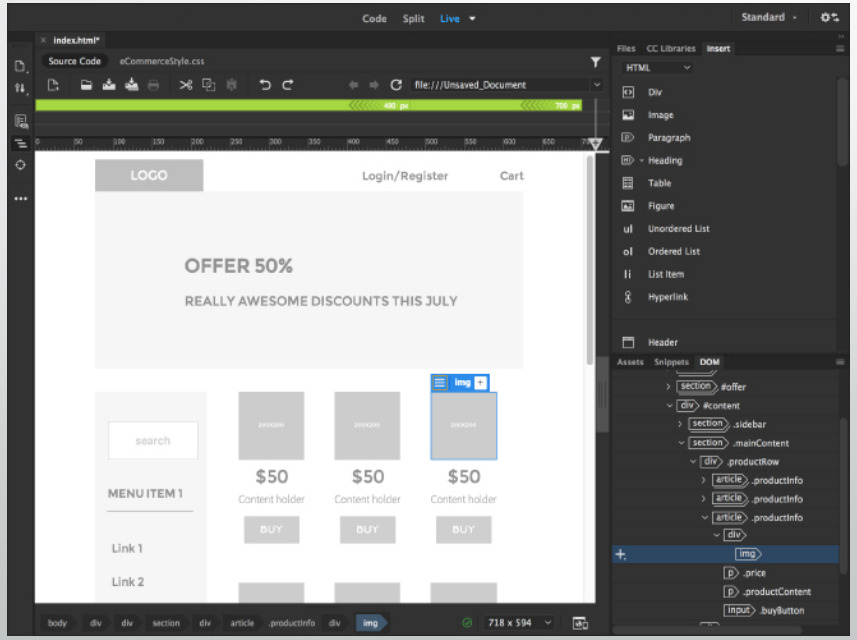
A variety of tools available for web development

- WYSIWYG
- Code editors
- IDEs
- Cloud based environments
- Code playgrounds

Tools Insights

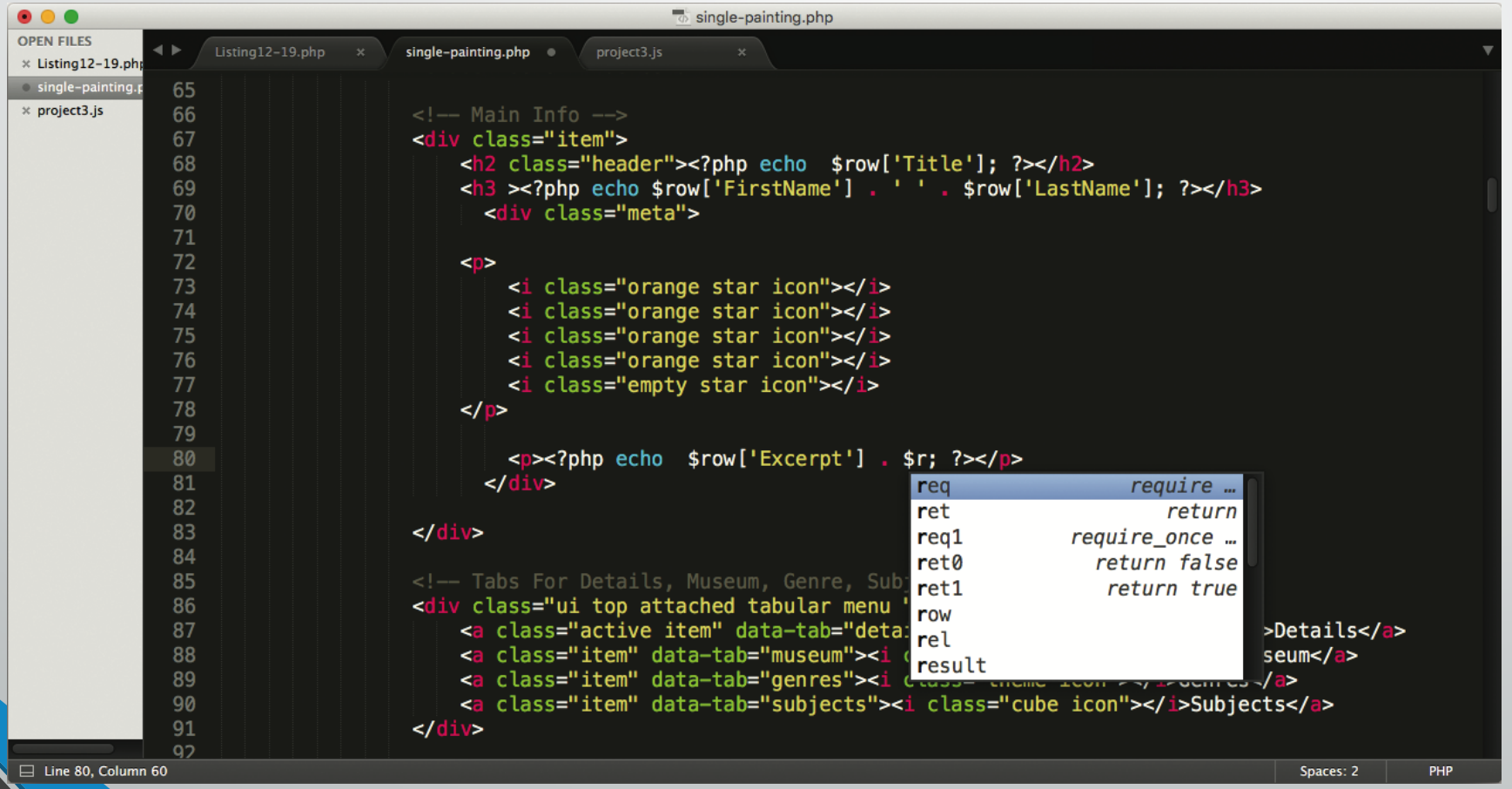
WYSIWYG Editors

- Dreamweaver
- Google Web Designer
- Seamonkey composer



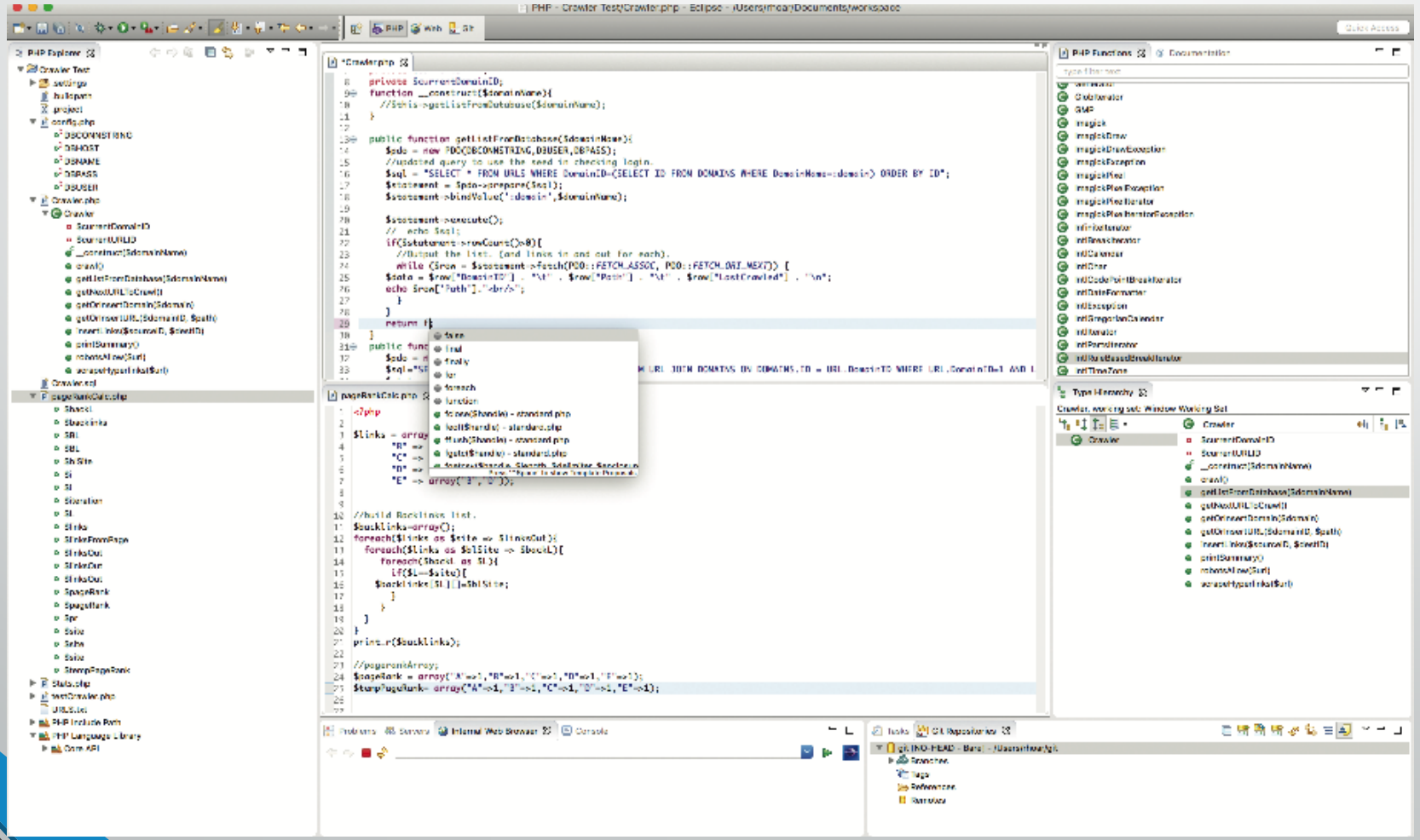
Tools Insights

Code editors



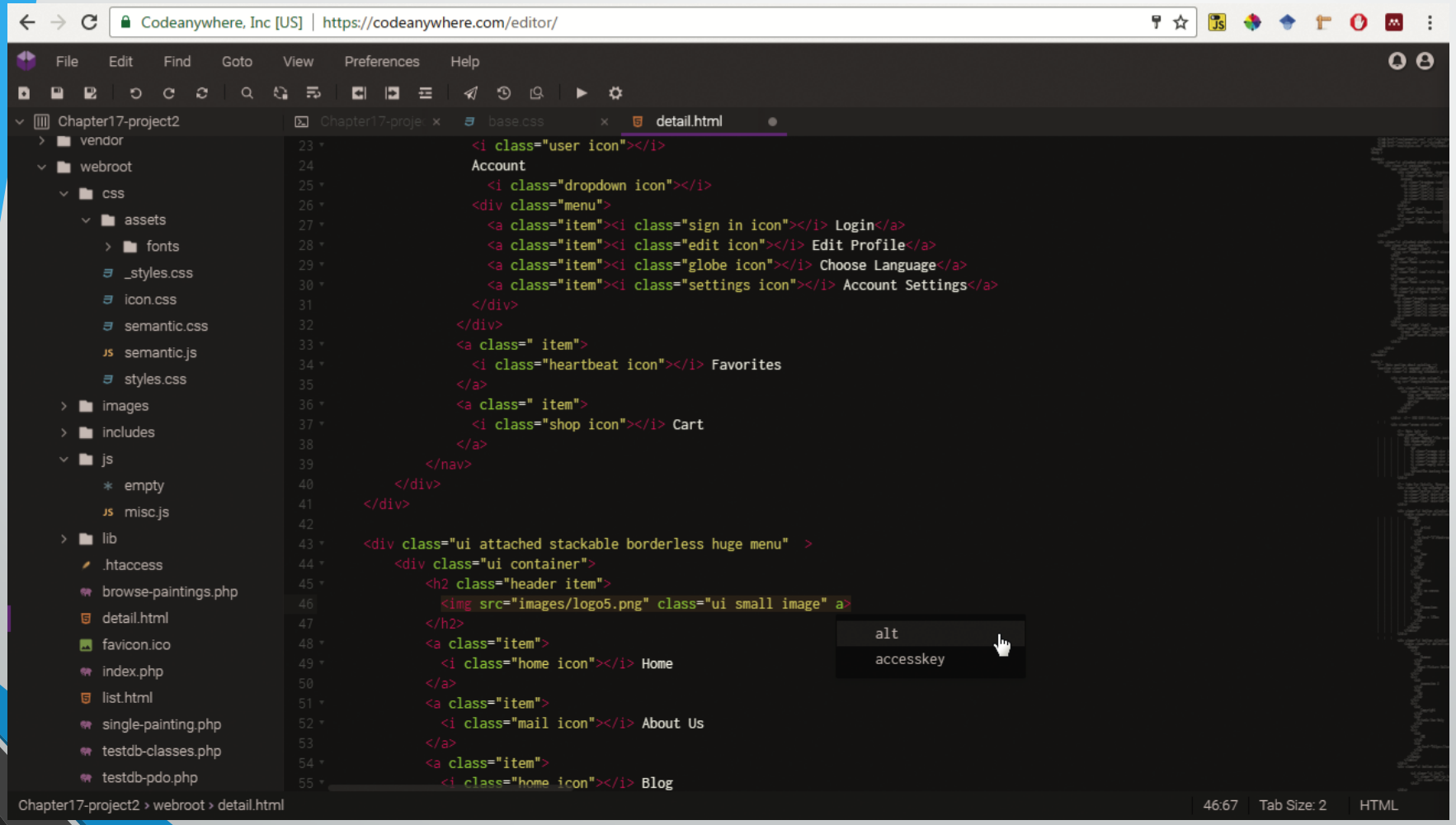
Tools Insights

IDEs



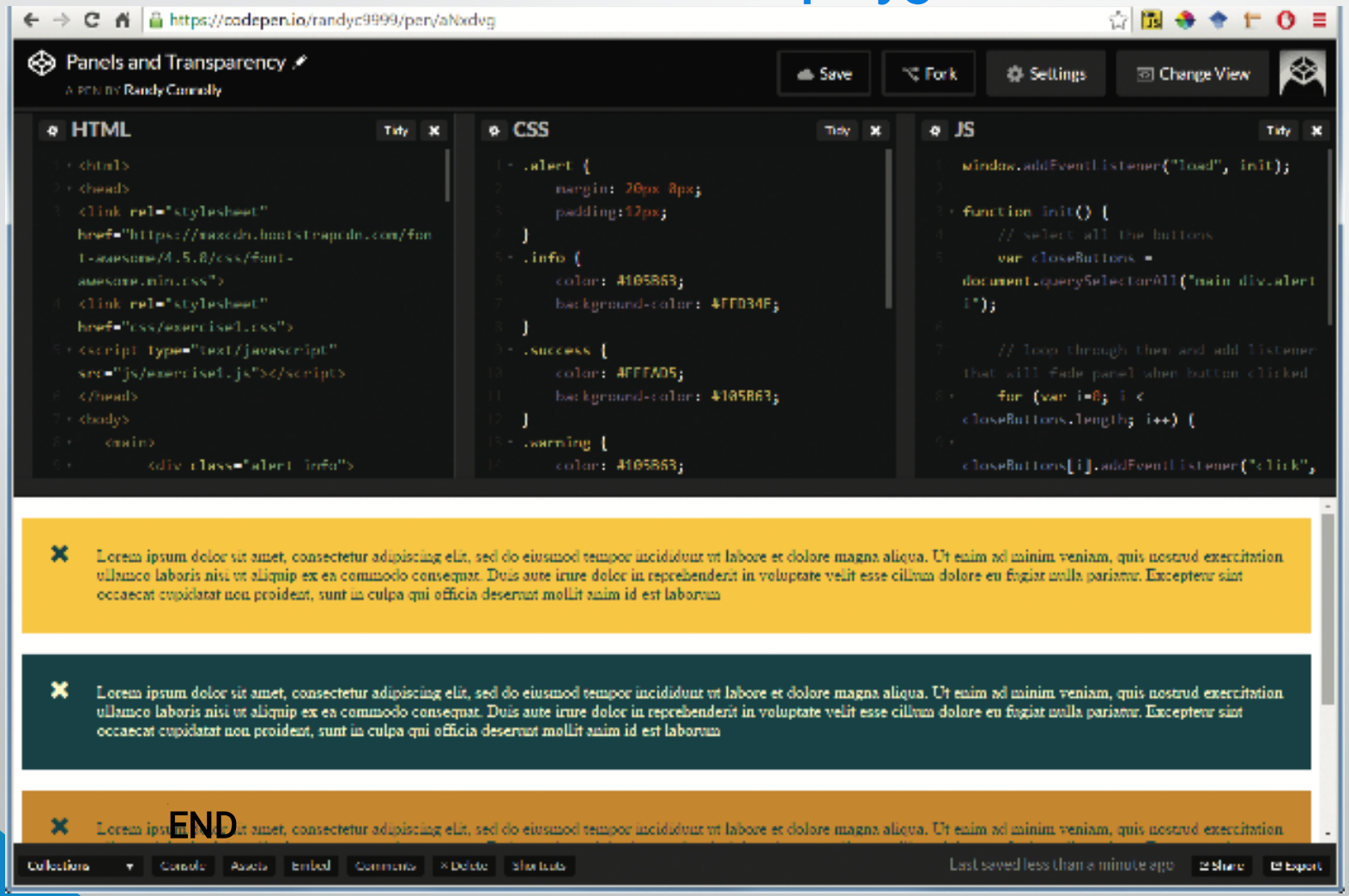
Tools Insights

Cloud-based Environments



Tools Insights

Code playgrounds



Web Systems and Technologies
TOPIC 19

Introduction to CSS

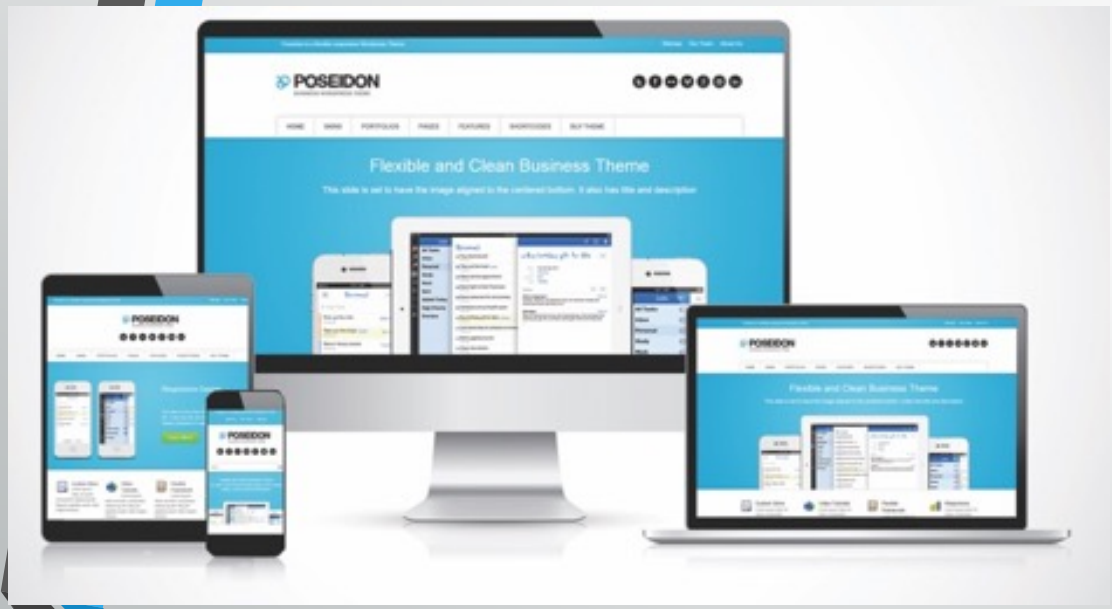
Introduction to CSS

Cascading Style Sheets (CSS)

- CSS is a W3C standard for describing the appearance of HTML elements
- With CSS, we can assign font properties, colors, sizes, borders, background images, and even position elements on the page
- CSS can be added directly to any HTML element (via the style attribute), within the <head> element, or, most commonly, in a separate text file that contains only CSS.

Introduction to CSS

Benefits of CSS



- Improved control over formatting
- Improved site maintainability
- Improved accessibility
- Improved page-download speed
- Improved output flexibility (responsive design)

Introduction to CSS

CSS Versions

- 1996 the CSS Level 1 Recommendation was published
- June 2011 - CSS2.1 becomes an official W3C Recommendation
- From CSS 3 onwards, recommendations are made based on modules. Following modules have been recommended:
  - Cascading and Inheritance
  - Color Module
  - Fonts Module
  - Media Queries
  - Selectors
  - Basic User Interface Module

Introduction to CSS

Browser Adoption

- Historically implementation of CSS was not consistent across major browsers
- CSS has a reputation for being a somewhat frustrating language due to browser implementation differences
- CSS was not really designed to be a layout language, so authors often find it tricky dealing with floating elements, relative positions, inconsistent height handling, overlapping margins, and non-intuitive naming

Web Systems and Technologies
TOPIC 20

CSS Syntax (Properties)

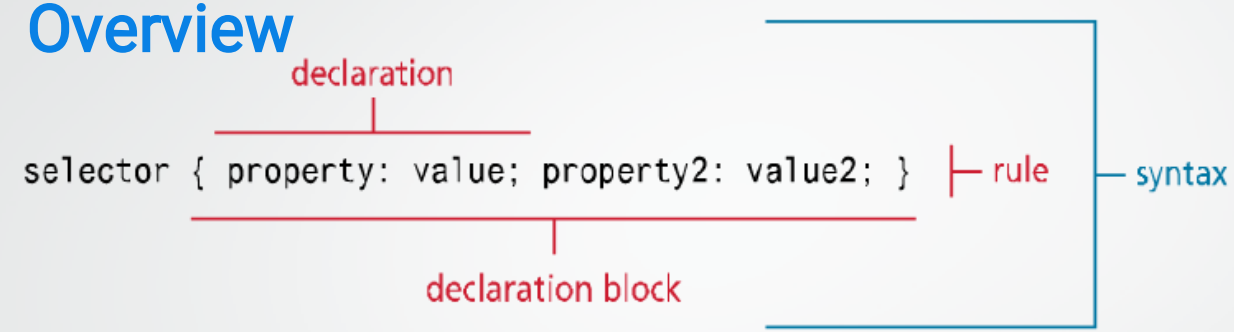
CSS Syntax

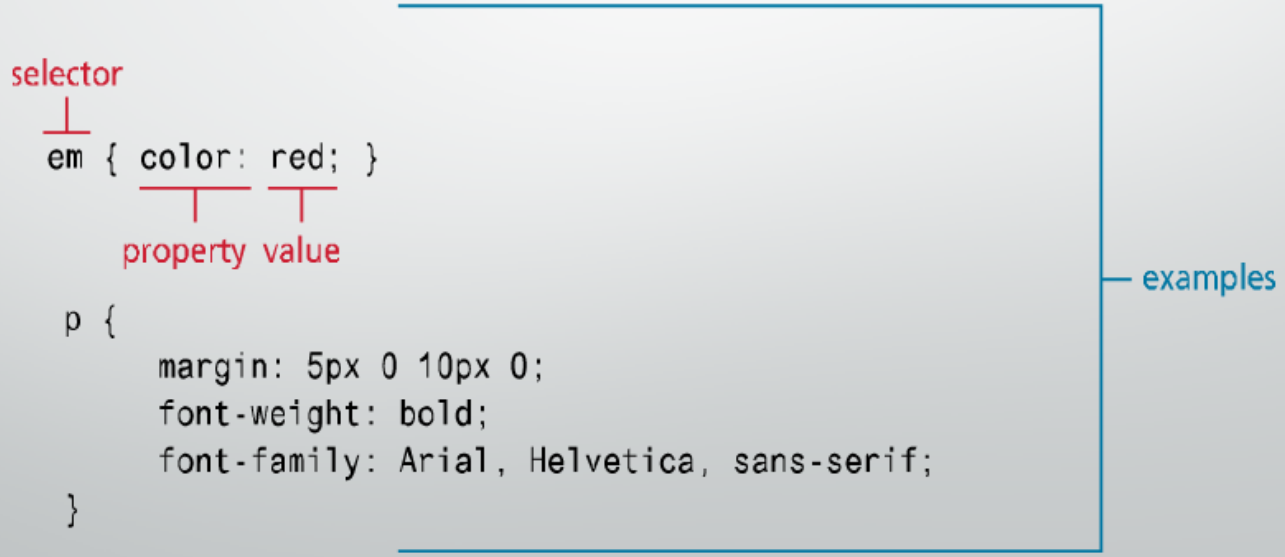
Overview

- A CSS document consists of one or more **style rules**
- A rule consists of a **selector** that identifies the HTML element or elements that will be affected, followed by a series of **property:value** pairs (each pair is also called a declaration)
- The series of declarations is also called the declaration block

CSS Syntax

Overview





CSS Syntax

Selectors

- Every CSS rule begins with a selector
- The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule
- Many ways to write selectors (stay tuned…)

CSS Syntax

Properties

- Each individual CSS declaration must contain a property.
- The CSS2.1 recommendation defines over a hundred different property names

CSS Syntax

Property Type	Property
Fonts	font font-family font-size font-style font-weight @font-face
Text	letter-spacing line-height text-align text-decoration text-indent

CSS Syntax

Properties

Property Type	Property
Color and Background	background background-color background-image background-position background-repeat box-shadow color opacity
Borders	border border-color border-width border-style border-top, border-left, ... border-image border-radius

CSS Syntax

Properties

Property Type	Property
Spacing	padding padding-bottom, padding-left, ... margin margin-bottom, margin-left, ...
Sizing	height max-height max-width min-height min-width width
Layout	bottom, left, right, top clear display float overflow position visibility z-index

CSS Syntax

Properties

Property Type	Property
Lists	list-style list-style-image list-style-type
Effects	animation filter perspective transform transition

Web Systems and Technologies
TOPIC 21

CSS Syntax (Values)

CSS Syntax

Values

- The unit of any given value is dependent upon the property.
- Some property values are from a predefined list of keywords. Others are values such as length measurements, percentages, numbers without units, color values, and URLs

CSS Syntax

Colors

- Name
- Hexadecimal
- RGB
- RGBa
- HSL

CSS Syntax

short question position property.

Relative and absolute Units

- Relative
  - px
  - em
  - %
  - vw,vh
- Absolute
  - in
  - cm
  - pt

Web Systems and Technologies
TOPIC 22

Location of Styles

Location of Styles

Inline Styles

- **Inline styles** are style rules placed within an HTML element via the style attribute

```
<h1>Share Your Travels</h1>
<h2 style="font-size: 24pt">Description</h2>
...
<h2 style="font-size: 24pt; font-weight:bold;">
Reviews</h2>
```

code line give thi or pucha huwa tha k yhn kn si location style ko use kiy gya ha

- Using inline styles is generally discouraged
- Handy for quickly testing out a style change

Location of Styles

Embedded Style Sheet

- Also called internal styles are style rules placed within the **<style>** element (inside the <head> element of an HTML document)

```
<head>
  <meta charset="utf-8">
  <title>Share Your Travels </title>
  <style>
  h1 { font-size: 24pt; }
  h2 {
    font-size: 18pt;
    font-weight: bold;
  }
  </style>
</head>
<body>
...
```

Location of Styles

External Style Sheet

- External style sheets are style rules placed within an external text file with the .css extension

```
<head>
  <meta charset="utf-8">
  <title>Share Your Travels</title>
  <link rel="stylesheet" href="styles.css" />
</head>
```

Location of Styles

External Style Sheet

Three different types of style sheets:

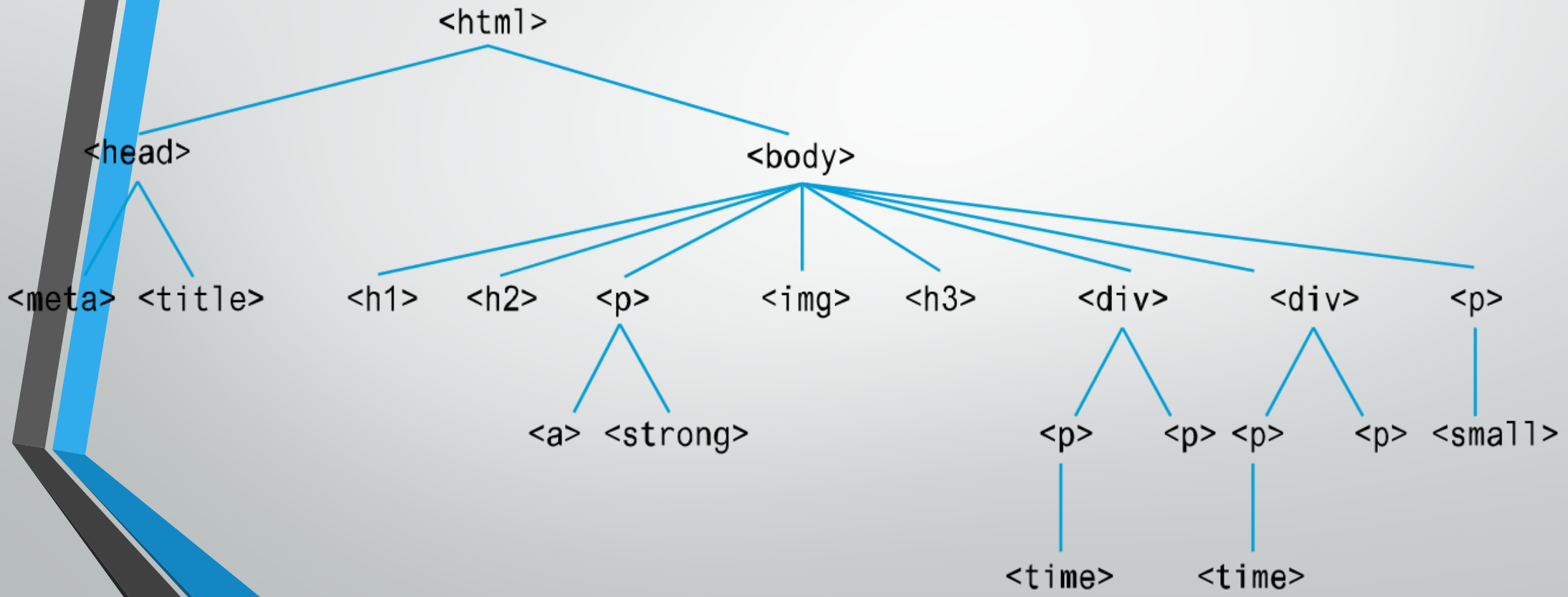
1. Author-created style sheets
2. User style sheets
3. Browser style sheets

Web Systems and Technologies
TOPIC 23

CSS Selectors
(Element, Class, ID)

Selectors

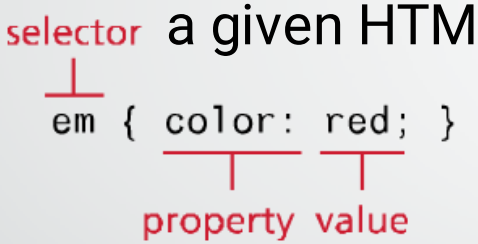
HTML DOM Tree



Selectors

Element Selectors

- **Element selectors** select all instances of a given HTML element.




```
p {
      margin: 5px 0 10px 0;
      font-weight: bold;
      font-family: Arial, Helvetica, sans-serif;
}
```

Selectors

Class Selectors

- A **class selector** allows you to simultaneously target different HTML elements regardless of their position in the document tree using the same class attribute value.

Selectors

which symbol is use for class selector.

Class Selectors

```
<head>
        <title>Share Your Travels </title>
        <style>
                .first {
                 font-style: italic;
                 color: red;
                }
        </style>
</head>


<body>
        <h1 class="first">Reviews</h1>
        <div>
          <p class="first">By Ricardo on <time>2016-05-23</time></p>
          <p>Easy on the HDR buddy.</p>
        </div>
        <hr/>
        <div>
          <p class="first">By Susan on <time>2016-11-18</time></p>
          <p>I love Central Park.</p>
        </div>
        <hr/>
</body>
```

Selectors

ID Selectors

- An **id selector** allows you to target a specific element by its id attribute regardless of its type or position

Selectors

ID Selectors

```
<head>
        <title>Share Your Travels </title>
        <style>
                #latestComment {
                 font-style: italic;
                 color: red;
                }
        </style>
</head>



<body>
        <h1 class="first">Reviews</h1>
        <div id="latestComment">
         <p class="first">By Ricardo on <time>2016-05-23</time></p>
         <p>Easy on the HDR buddy.</p>
        </div>
        <hr/>
        <div>
         <p class="first">By Susan on <time>2016-11-18</time></p>
         <p>I love Central Park.</p>
        </div>
        <hr/>
    ody>
```

Web System and Tecnologies CS 521

TOPIC 24

CSS Selectors (Attributes)

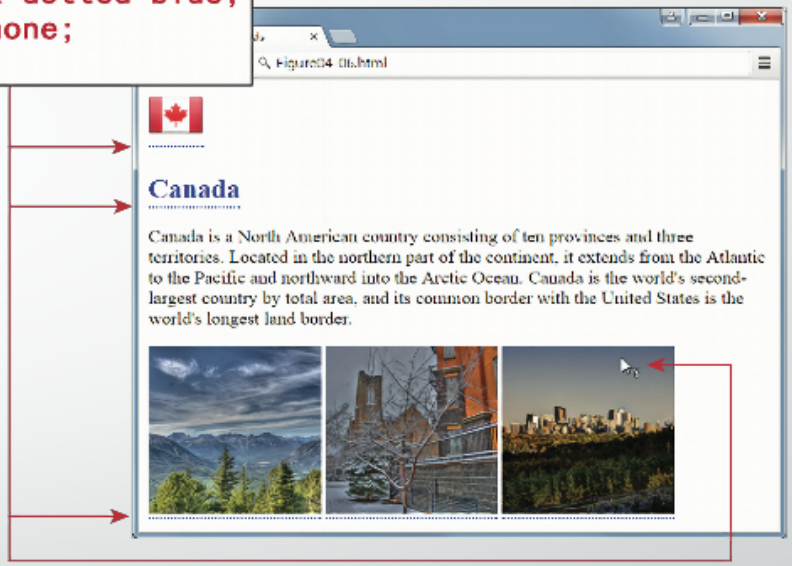
Attribute Selectors

- An **attribute selector** provides a way to select HTML elements either by the presence of an element attribute or by the value of an attribute
    - [title] { … }

CSS Selectors (Attributes)

```
<head>
    <meta charset="utf-8">
    <title>Share Your Travels</title>
    <style>
        [title] {
          cursor: help;
          padding-bottom: 3px;
          border-bottom: 2px dotted blue;
          text-decoration: none;
        }
    </style>
</head>
<body>
    <div>
        <img src="images/flags/CA.png" title="Canada Flag" />
        <h2><a href="countries.php?id=CA" title="see posts from Canada"> Canada</a></h2>
        <p>Canada is a North American country consisting of … </p>
        <div>
            <img src="images/square/6114907897.jpg" title="At top of Sulphur Mountain" />
            <img src="images/square/6592317633.jpg" title="Grace Presbyterian Church" />
            <img src="images/square/6592914823.jpg" title="Calgary Downtown" />
        </div>
    </div>
</body>
```

```
[title] {
   cursor: help;
   padding-bottom: 3px;
   border-bottom: 2px dotted blue;
   text-decoration: none;
}
```



CSS Selectors (Attributes)

Selector	Matches
[]	A specific attribute.
[=]	A specific attribute with a specific value.
[~=]	A specific attribute whose value matches at least one of the words in a space delimited list of words.
[^=]	A specific attribute whose value **begins** with a specified value.
[*=]	A specific attribute whose value **contains** a substring.
[$=]	A specific attribute whose value ends with a specified value.

The End

Web System and Tecnologies CS 521

TOPIC 25

CSS Selectors (Pseudo)

Pseudo-Element and Pseudo-Class Selectors:

- A **pseudo-element selector** is a way to select something that does not exist explicitly as an element in the HTML document tree but which is still a recognizable selectable object

- A **pseudo-class selector** does apply to an HTML element, but targets either a particular state or, in CSS3, a variety of family relationships

CSS Selectors (Pseudo)

Pseudo-Classes and -Elements

- Pseudo-Classes
  - a:link
  - a:visited
  - :hover
  - :active
  - :focus
  - :checked
  - :first-child
- Pseudo-Elements
  - ::first-letter
  - ::first-line
  - ::before
  - ::after
  - ::marker

CSS Selectors (Pseudo)

Pseudo-Element and Pseudo-Class Selectors:

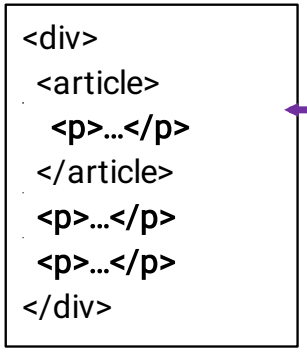
```
<style>
	a:link {
		text-decoration: underline;
		color: blue;
	}
	a:visited {
		text-decoration: underline;
		color: purple;
	}
	a:hover {
		text-decoration: none;
		font-weight: bold;
	}
	a:active {
		background-color: yellow;
	}
</style>
```

The End

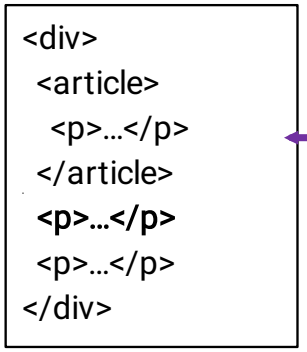
Web System and Tecnologies CS 521

TOPIC 26

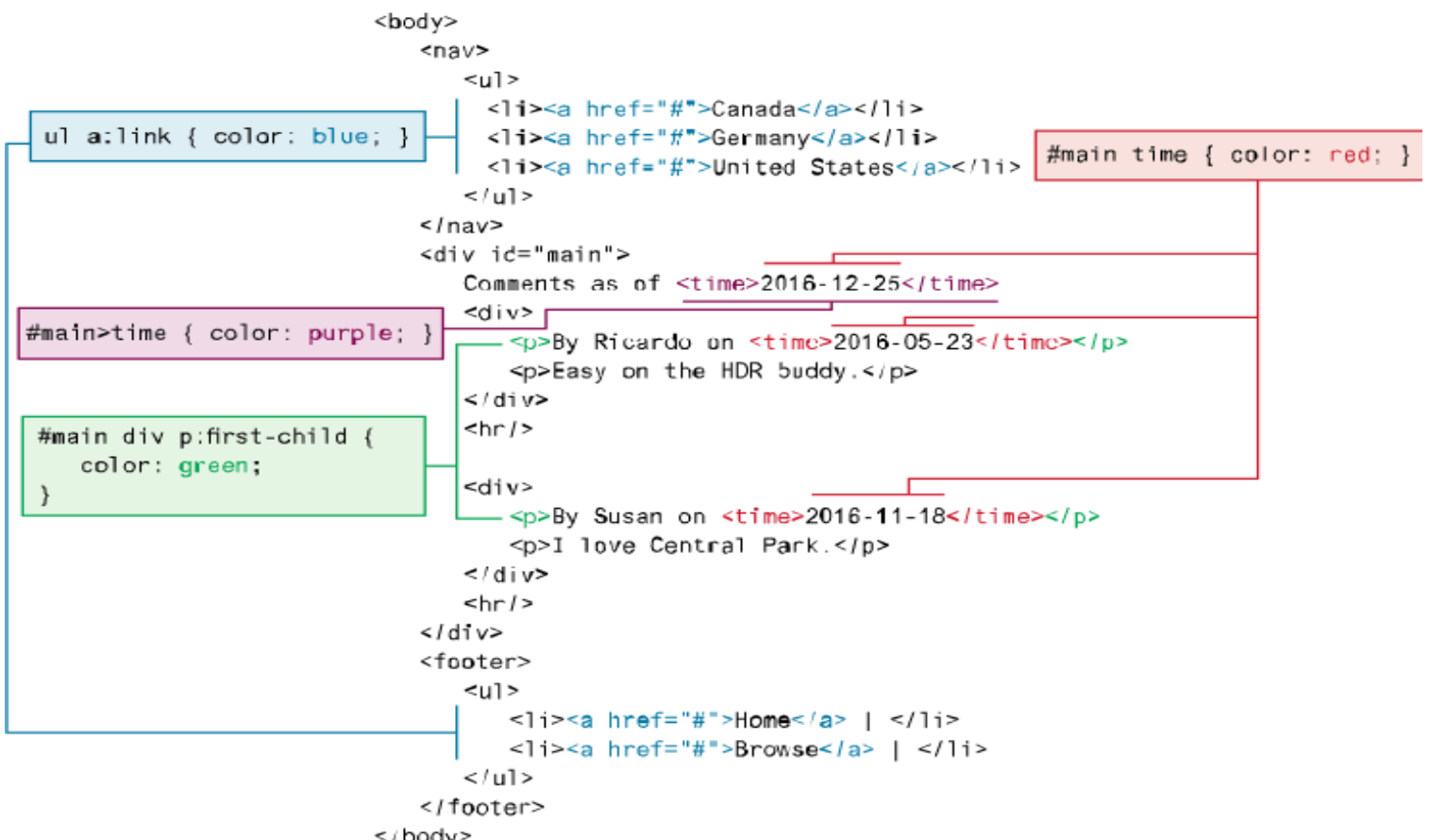
CSS Selectors (Contextual)



Selector	Matches	Example
Descendant	A specified element that is contained somewhere within another specified element.	**div p** Selects a <p> element that is contained somewhere within a element.
Child	A specified element that is a direct child of the specified element.	**div>p** Selects a <p> element that is a child of a element.
Adjacent Sibling	A specified element that is the next sibling (i.e., comes directly after) of the specified element	**h3+p** Selects the first <p> after any <h3>.
General Sibling	A specified element that shares the same parent as the specified element	**h3~p** Selects all the <p> elements that share the same parent as the <h3>.



CSS Selectors (Contextual)



The End

Web System and Tecnologies CS 521

TOPIC 27

The Cascade: How Styles Interact

CSS:

- The “Cascade" in CSS refers to how conflicting rules are handled.
- The downward movement of water down a cascade is meant to be analogous to how a given style rule will continue to take precedence with child elements
- CSS uses the following cascade principles to help it deal with conflicts:
  - Inheritance
  - Specificity
  - Location

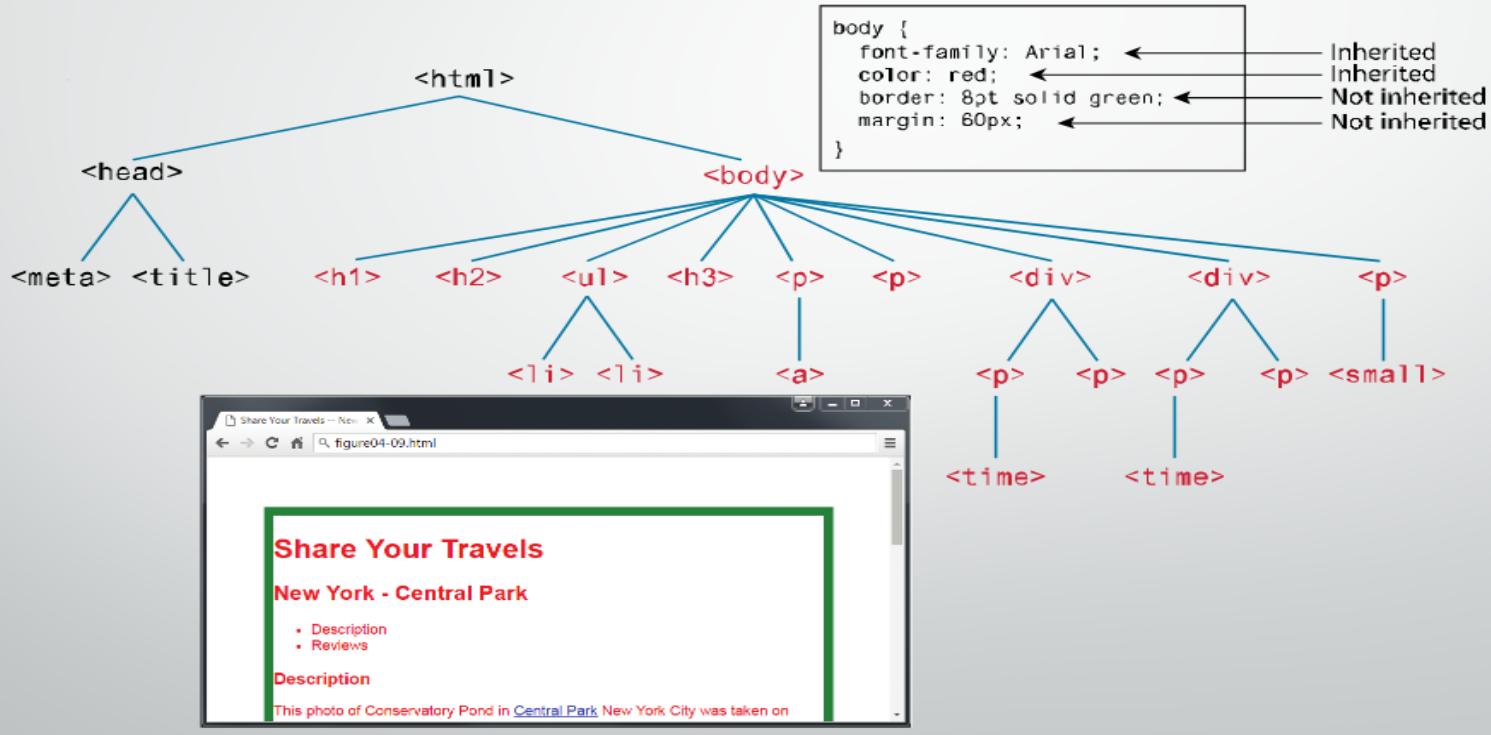
The Cascade: How Styles Interact

Inheritance:

- Many (but not all) CSS properties affect not only themselves but their descendants as well
- Inheritable:
  - font, color, list, and text properties
- Not inheritable:
  - layout, sizing, border, background, and spacing properties

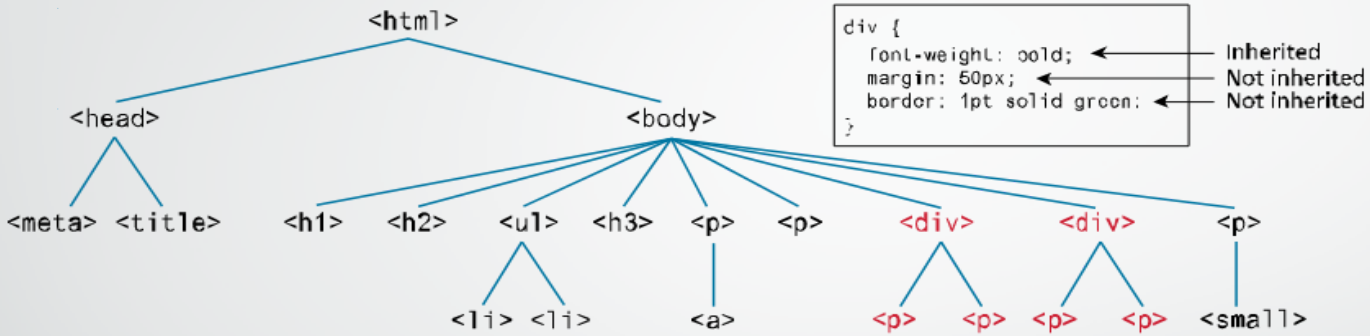
The Cascade: How Styles Interact

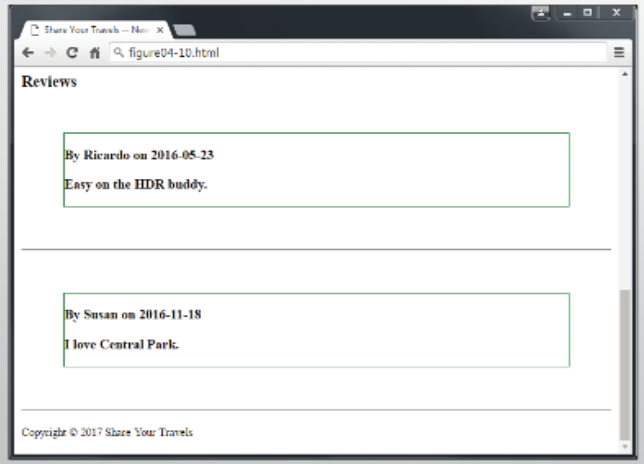
Inheritance:



The Cascade: How Styles Interact

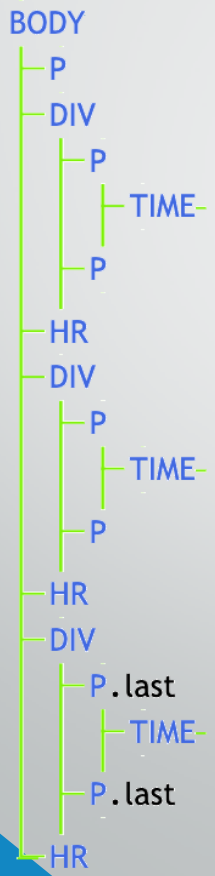
Inheritance:





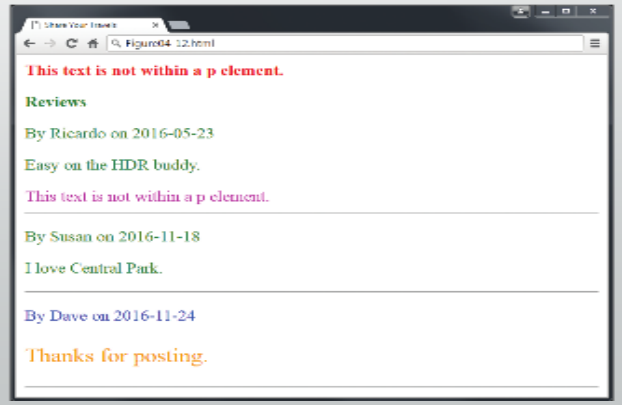
The Cascade: How Styles Interact

Specificity:



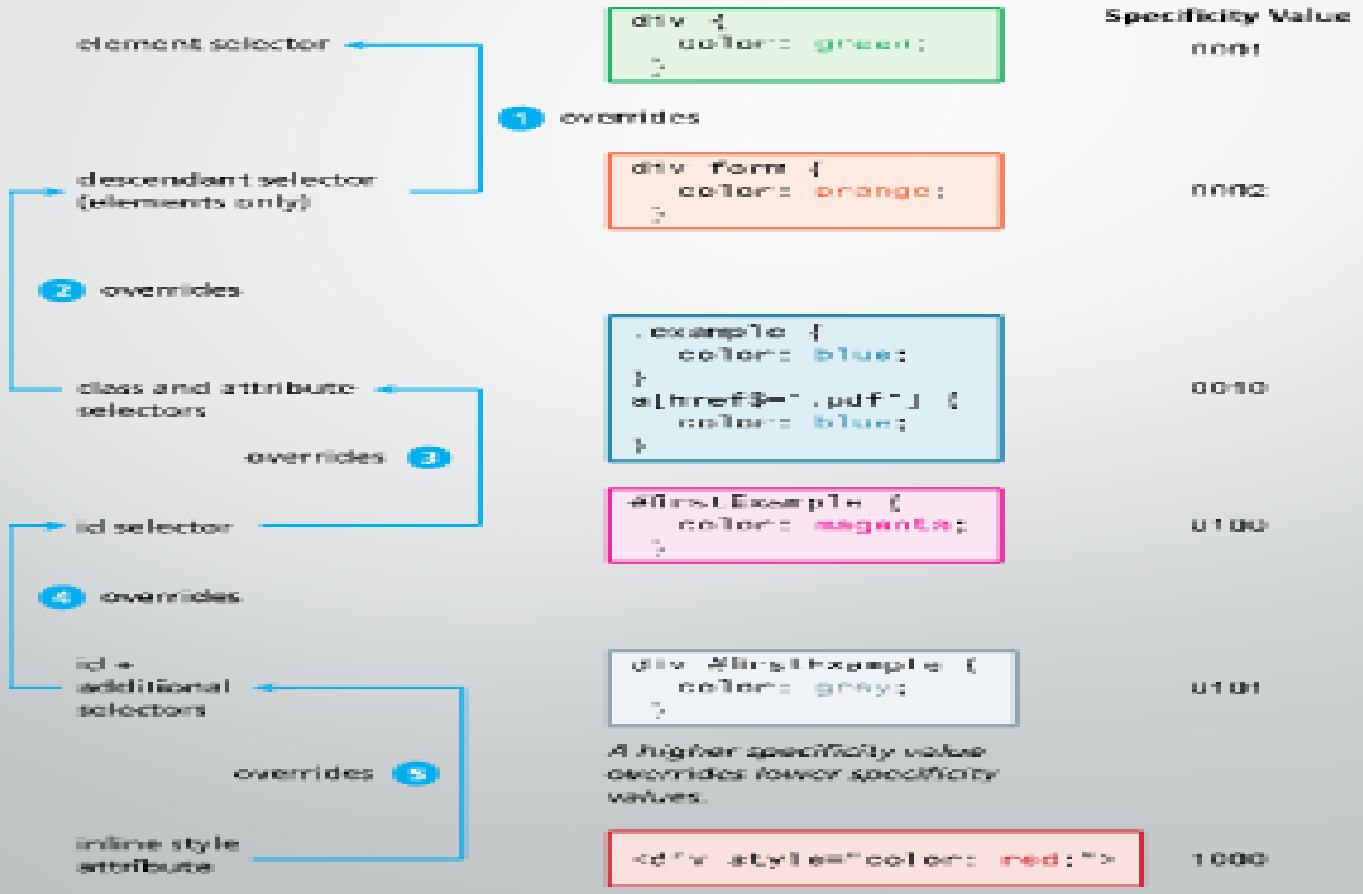

```
body {
  font-weight: bold;
  color: red;
}

div {
  font-weight: normal;
  color: magenta;
}

p {
  color: green;
}

.last {
  color: blue;
}

#verylast {
  color: orange;
  font-size: 10pt;
}
```

```
<body>
   This text is not within a p element.
   <p>Reviews</p>
   <div>
      <p>By Ricardo on <time>2016-05-23</time></p>
      <p>Easy on the HDR buddy.</p>
      This text is not within a p element.
   </div>
   <hr/>

   <div>
      <p>By Susan on <time>2016-11-18</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>

   <div>
      <p class="last">By Dave on <time>2016-11-24</time></p>
      <p class="last" id="verylast">Thanks for posting.</p>
   </div>
   <hr/>
</body>
```



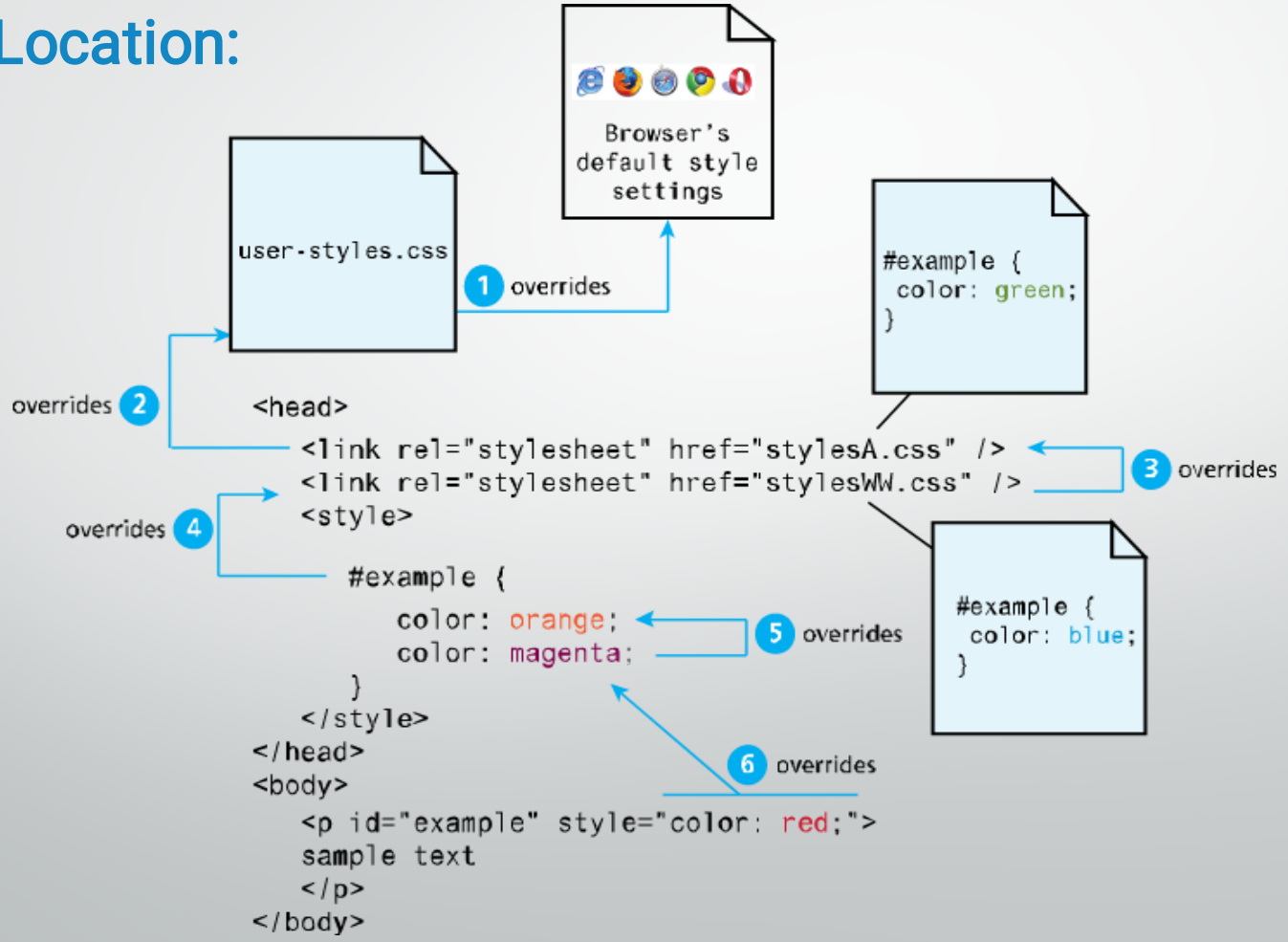
The Cascade: How Styles Interact

Specificity Algorithm:



The Cascade: How Styles Interact

Location:



Inline Style (style attribute): color: yellow; (Highest specificity because it's inline)
Embedded Styles - ID Selector: color: blue; (High specificity)
Embedded Styles - Class Selector: color: green; (Medium specificity)
Embedded Styles - Type Selector: color: red; (Low specificity)
external style sheet
user defined
browser defined

start from highest to lowest

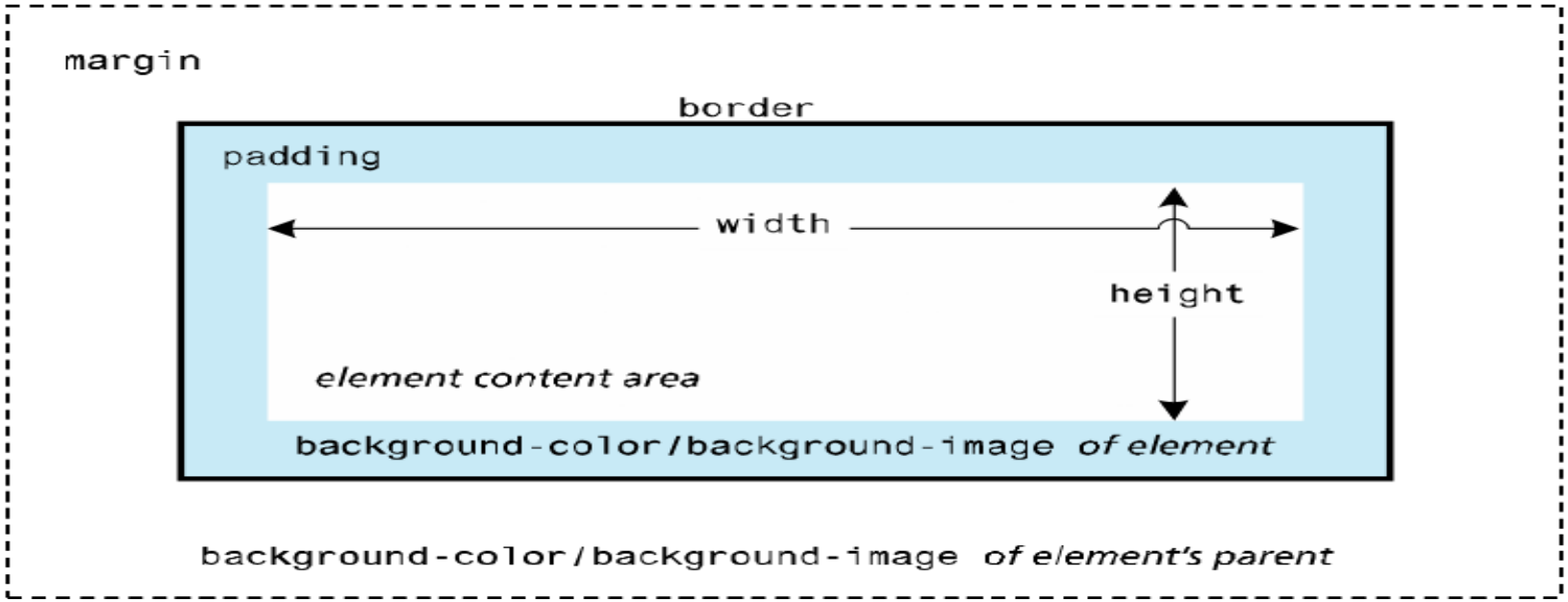
The End

TOPIC 28
WITH
VOICE NOTE

Web System and Tecnologies CS 521

The Box Model

Overview:



The Box Model

Background:

- Background
- Background-attachment
- Background-color
- Background-image
- Background-position
- Background-repeat
- Background-size

The Box Model

BackGround:

```
background-image: url(../images/backgrounds/body-background-tile.gif);
background-repeat: repeat;
```

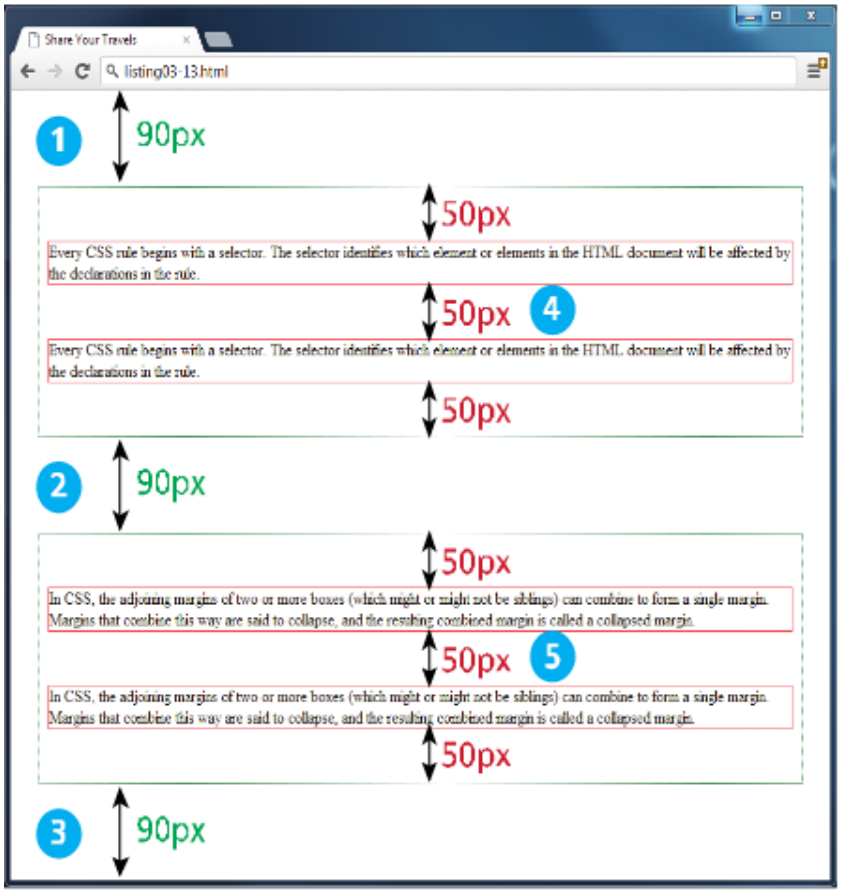
```
background-repeat: no-repeat;
```

```
background-repeat: repeat-y;
```

```
background-repeat: repeat-x;
```

The Box Model

Collapsing Margins:

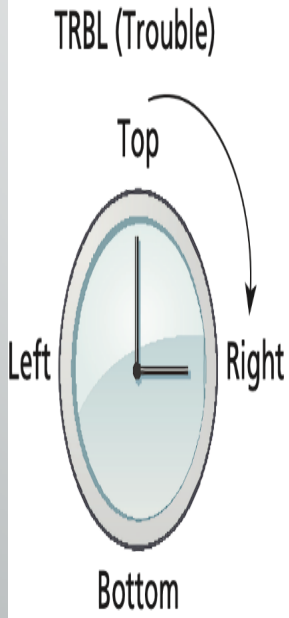



```
<div>
  <p>Every CSS rule ...</p>
  <p>Every CSS rule ...</p>
</div>
<div>
  <p>In CSS, the adjoining ... </p>
  <p>In CSS, the adjoining ... </p>
</div>
```

```
div {
    border: dotted 1pt green;
    padding: 0;
    margin: 90px 20px;
}
```

```
p {
    border: solid 1pt red;
    padding: 0;
    margin: 50px 20px;
}
```

The Box Model




```
border-color: top right bottom left;
```

```
border-color: red green orange blue;
```

CSS TRBL Shortcuts:

- border-top-color: red; /* sets just the top side */
- border-right-color: green; /* sets just the right side */
- border-bottom-color: yellow; /* sets just the bottom side */
- border-left-color: blue; /* sets just the left side */
- Alternately, we can set all four sides at once:
- border-color: red; /* sets all four sides to red */
- border-color: red green orange blue; /* sets 4 colors */

The End

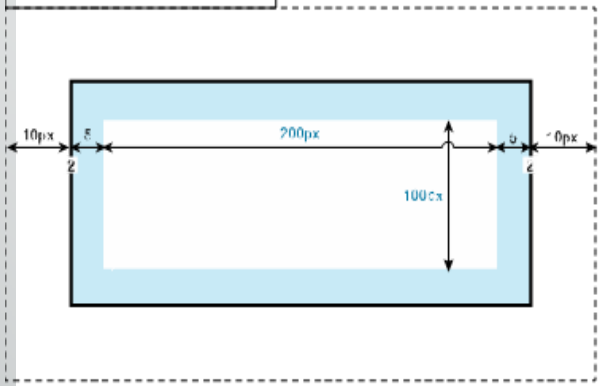
Web System and Tecnologies CS 521

TOPIC 29

The Box Model

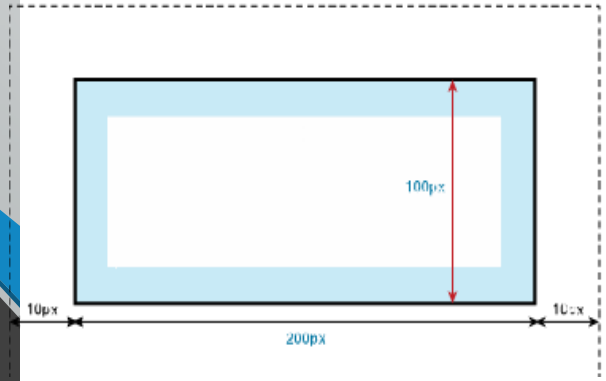
```
div {
  box-sizing: content-box;
  width: 200px;
  height: 100px;
  padding: 5px;
  margin: 10px;
  border: solid 2pt black;
}
```

True element width = 10 + 2 + 5 + 200 + 5 + 2 + 10 = 234 px
True element height = 10 + 2 + 5 + 100 + 5 + 2 + 10 = 134 px




```
div {
  box-sizing: border-box;
}
```

True element width = 10 + 200 + 10 = 220 px
True element height = 10 + 100 + 10 = 120 px

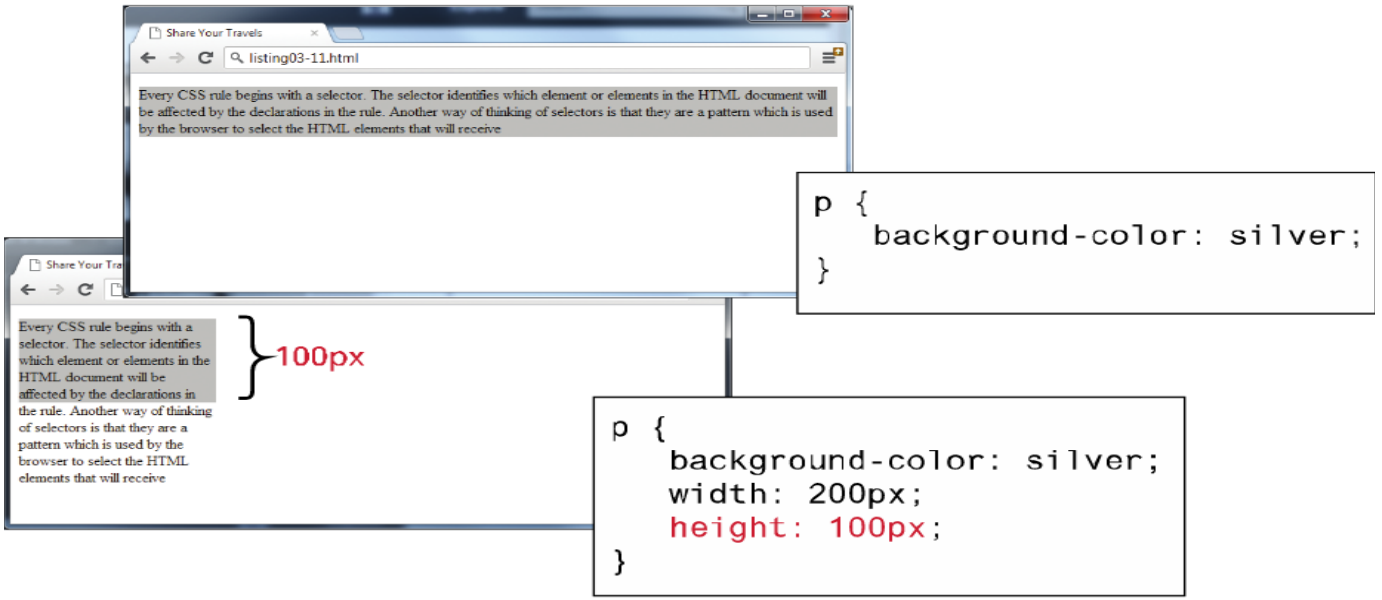


Box Dimensions:

- Content-box
  - width/height applied to contents only
  - borders and padding counted separately
  - margins counted separately
- Border-box
  - width/height include border and padding
  - margins counted separately

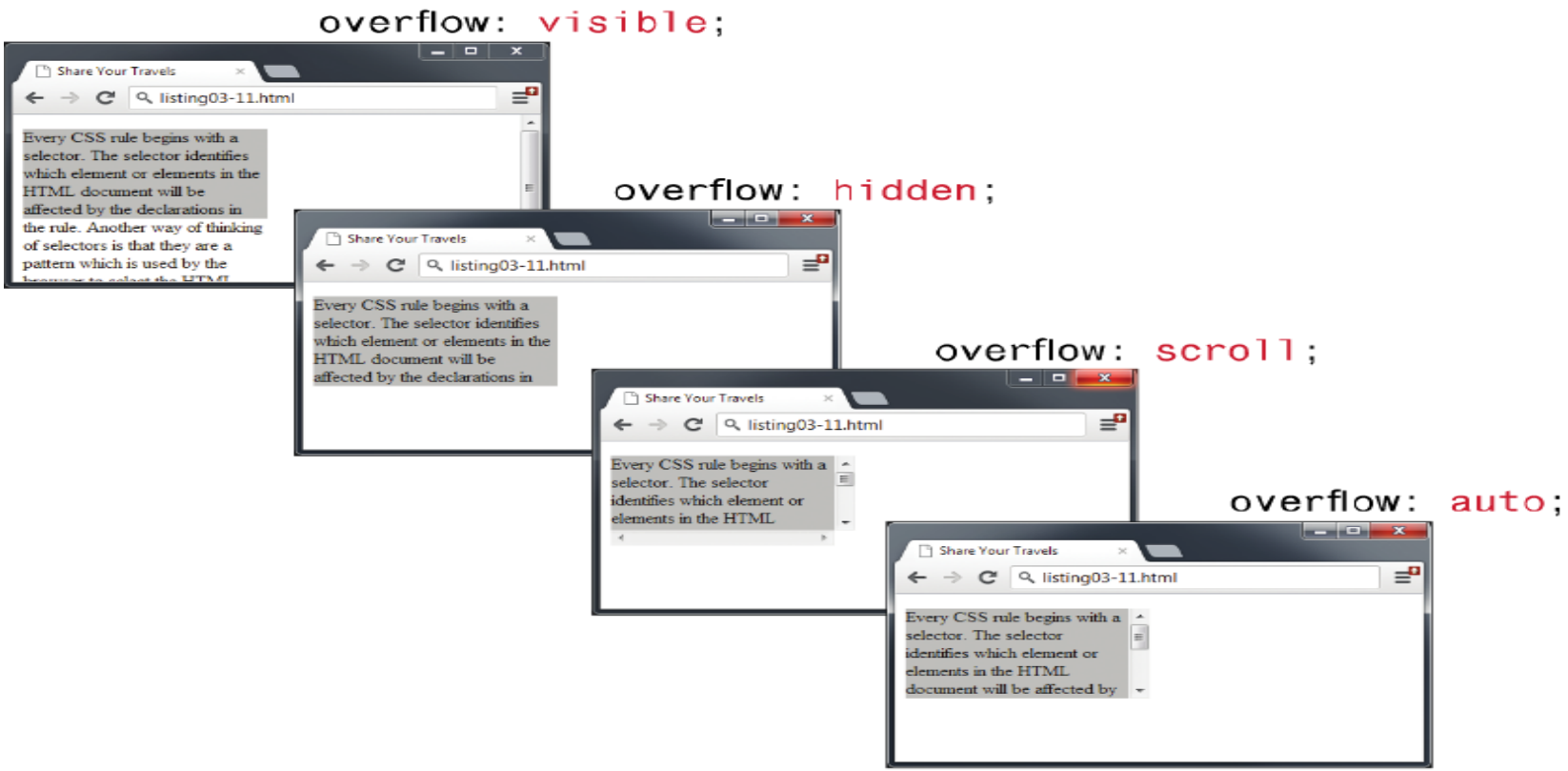
The Box Model

Limitation of Height Property:



The Box Model

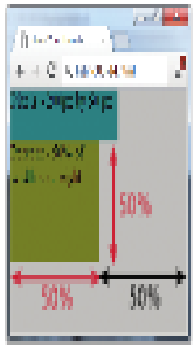
OverFlow Property:



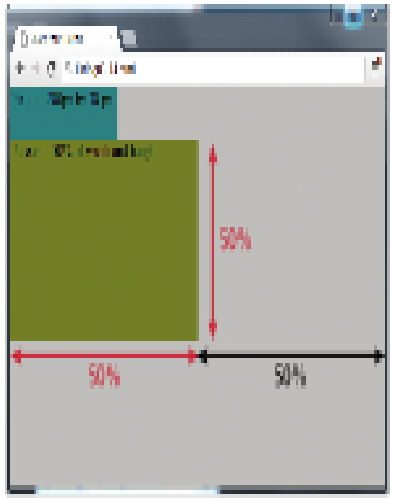
The Box Model

Box sizing Using Percentage:

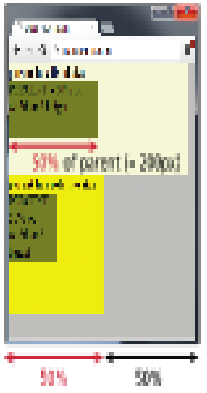
```
<style>
  html,body {
    margin:0;
    width:100%;
    height:100%;
    background: silver;
  }
  .pixels {
    width:200px;
    height:50px;
    background: teal;
  }
  .percent {
    width:50%;
    height:50%;
    background: olive;
  }
```




```
<body>
  <div class="pixels">
    Pixels - 200px by 50 px
  </div>
  <div class="percent">
    Percent - 50% of width and height
  </div>
</body>
```




```
  .parentFixed {
    width:400px;
    height:150px;
    background: beige;
  }
  .parentRelative {
    width:50%;
    height:50%;
    background: yellow;
  }
</style>
```

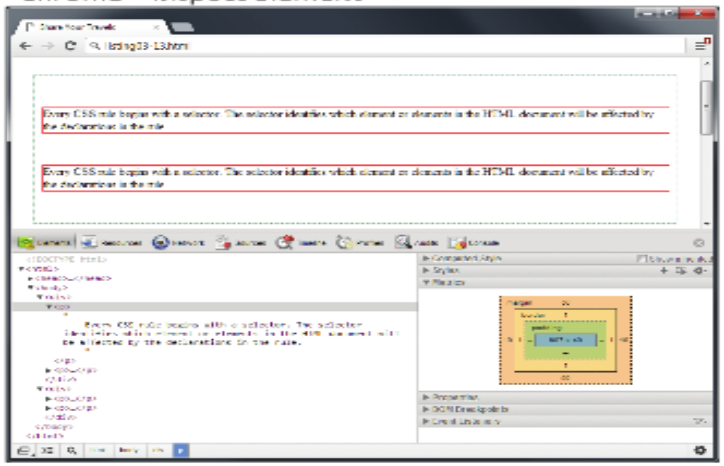



```
<body>
<div class="parentFixed">
  <strong>parent has fixed size</strong>
  <div class="percent">
    PERCENT - 50% of width and height
  </div>
</div>
<div class="parentRelative">
  <strong>parent has relative size</strong>
  <div class="percent">
    PERCENT - 50% of width and height
  </div>
</div>
</body>
```

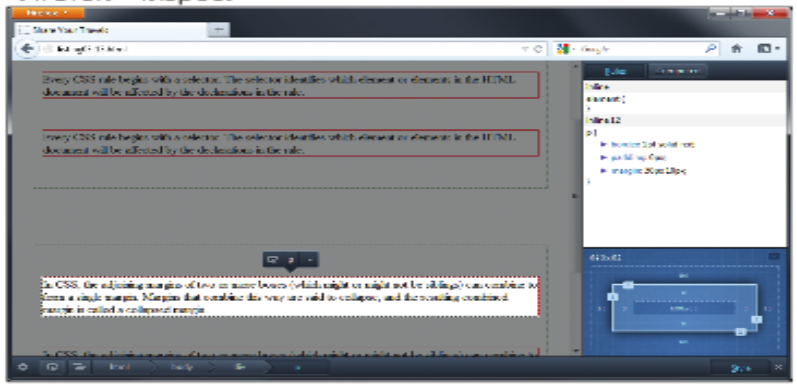
The Box Model

Inspecting CSS using Modern Browsers:

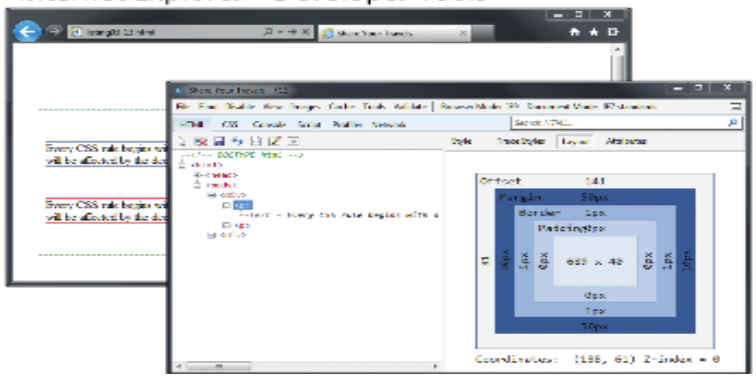
Chrome – Inspect Element



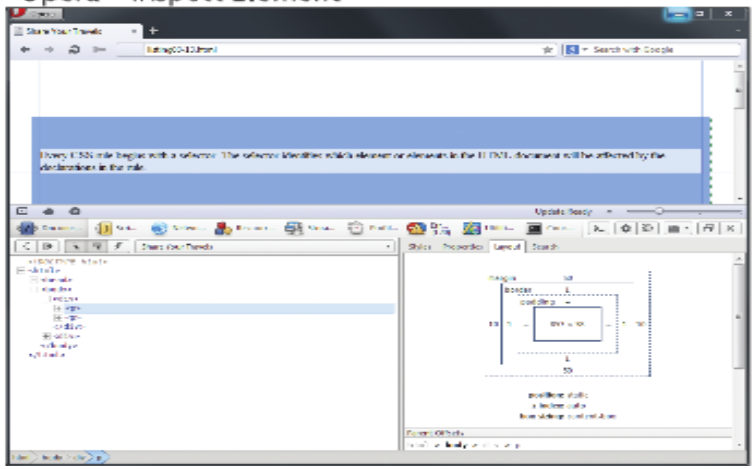
Firefox – Inspect



Internet Explorer – Developer Tools



Opera – Inspect Element



The End

TOPIC 30
WITH
VOICE NOTE

Web System and Tecnologies CS 521

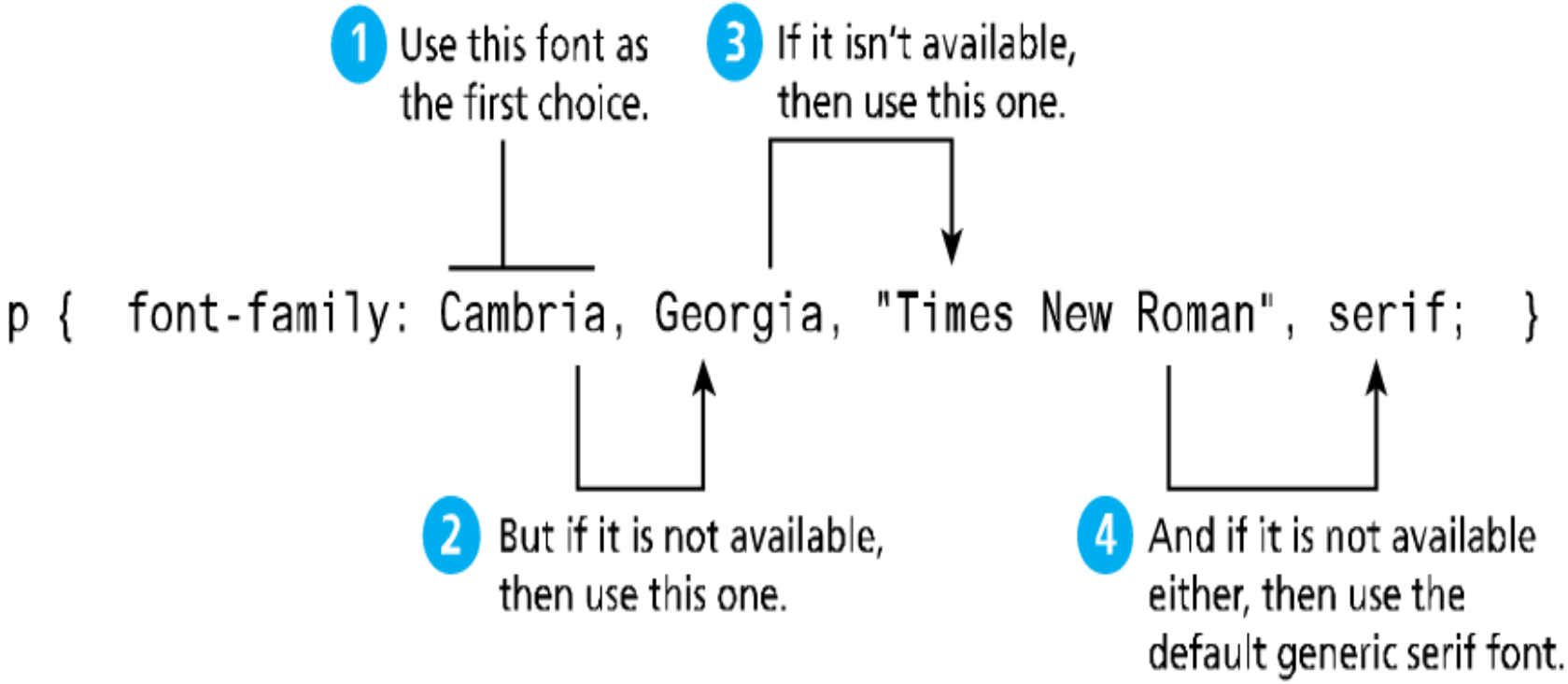
CSS Text Styling

Font Family:

Property	Description
font	A combined shorthand property that allows you to set the family, style, size, variant, and weight in one property. **style weight variant size font-family**
font-family	Specifies the typeface/font to use. More than one can be specified.
font-size	The size of the font in one of the measurement units
font-style	Specifies whether italic, oblique, or normal
font-variant	Specifies either small-caps text or none
font-weight	Specifies either normal, bold, bolder, lighter, or a value between 100 and 900 in multiples of 100, where larger number represents weightier (i.e., bolder) text.

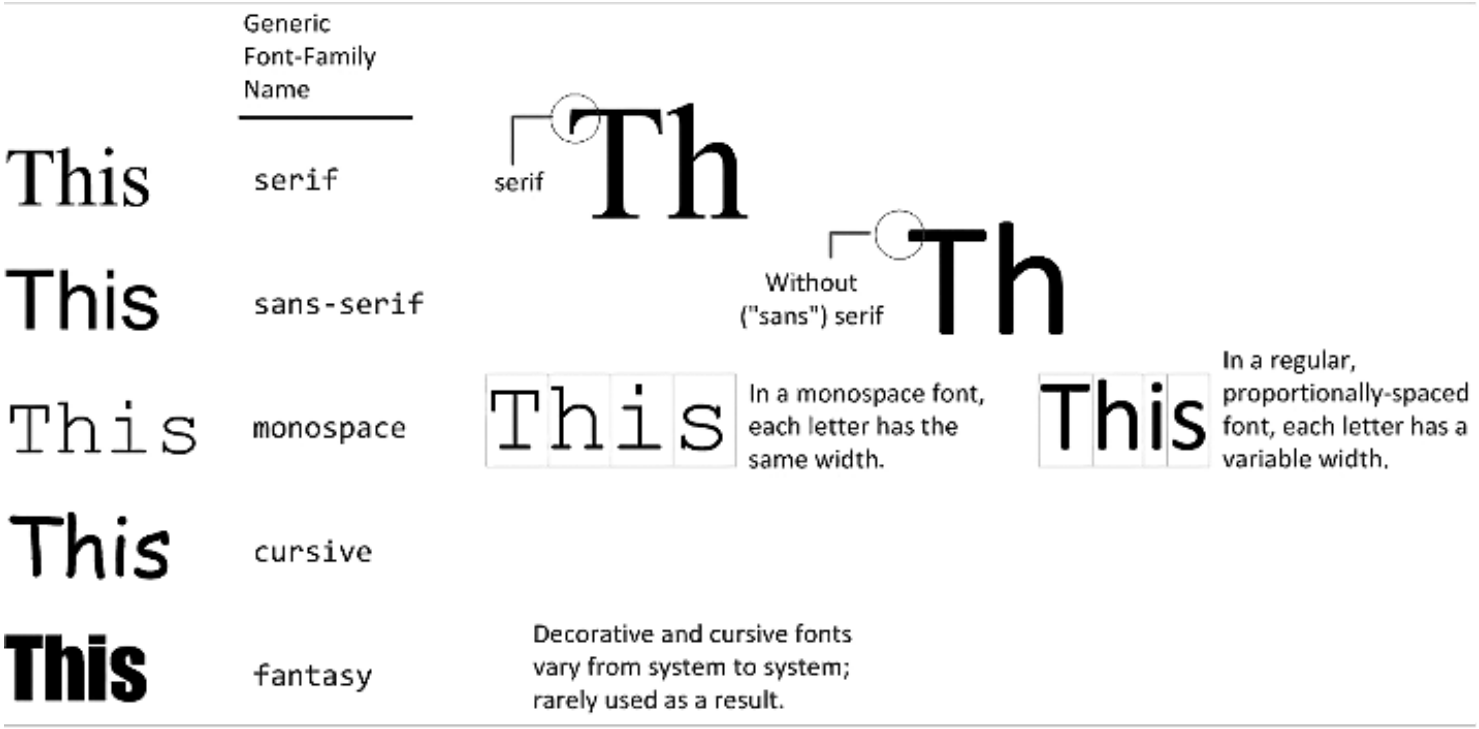
CSS Text Styling

Specifying the Font Family:



CSS Text Styling

Different Font Families:



CSS Text Styling

Font Sizes:

<body> Browser's default text size is usually 16 pixels

<p> 100% or 1em is 16 pixels

<h3> 125% or 1.125em is 18 pixels

<h2> 150% or 1.5em is 24 pixels

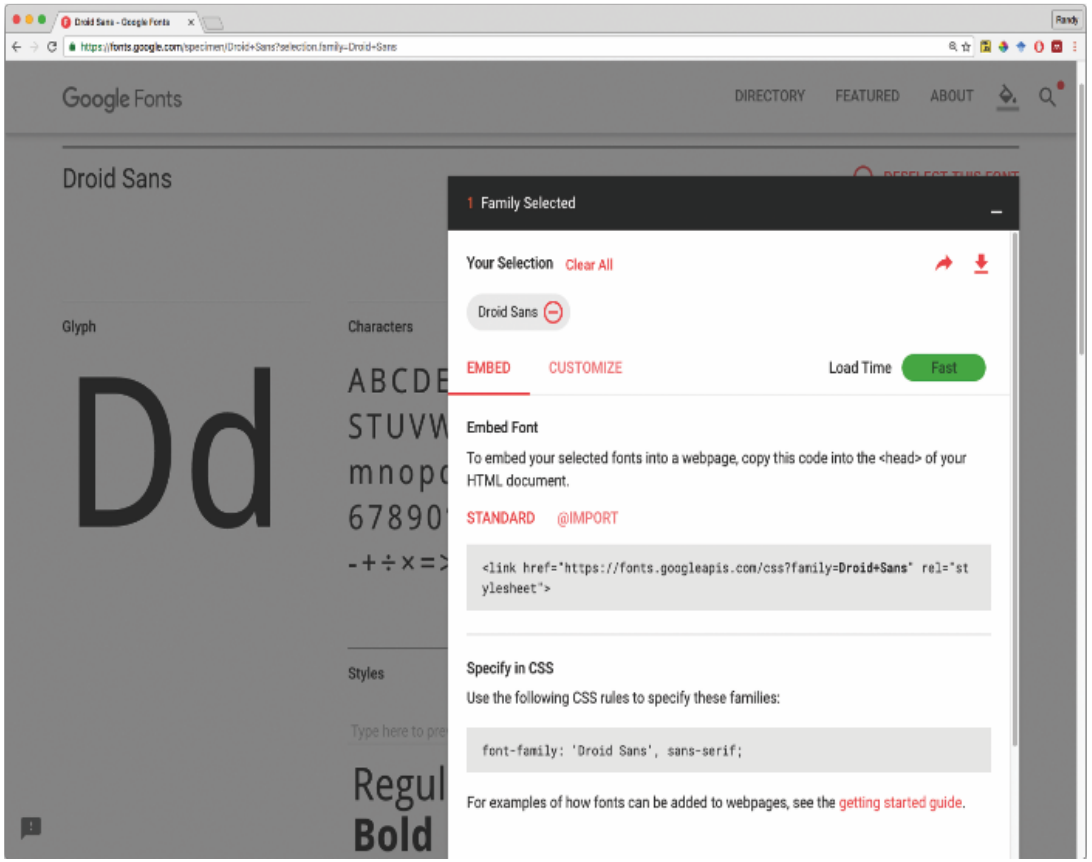
<h1> 200% or 2em is 32 pixels

```
/* using 16px scale */

body { font-size: 100%; }
p { font-size: 1em; }        /* 1.0  x 16 = 16 */
h3 { font-size: 1.125em; }   /* 1.25 x 16 = 18 */
h2 { font-size: 1.5em; }     /* 1.5 x 16  = 24 */
h1 { font-size: 2em; }       /* 2 x 16 = 32 */
```

```
<body>
  Browser's default text size is usually 16 pixels
  <p>100% or 1em is 16 pixels</p>
  <h3>125% or 1.125em is 18 pixels</h3>
  <h2>150% or 1.5em is 24 pixels</h2>
  <h1>200% or 2em is 32 pixels</h1>
</body>
```

CSS Text Styling

Embeded Fonts:



CSS Text Styling

Paragraph Properties:

- Just as there are properties that affect the font in CSS, there are also a range of CSS properties that affect text independently of the font.
  - letter-spacing
  - line-height
  - text-align
  - text-decoration
  - text-direction
  - text-shadow
  - …

CSS Text Styling

Just look at Text Shadow:

You will likely want the shadow color to be partly transparent

y offset
x offset
blur size
shadow color

1 `text-shadow: 20px 20px 10px rgba(0,0.0,0.5);`

2 `text-shadow: 4px 4px 0 #5C6BC0, 8px 8px 0 #7986CB, 12px 12px 0 #9FA8DA;`

multiple shadows can be defined (separated by commas)

3 `text-shadow: 0 1px 1px #1A237E;`

4 `box-shadow: 0px 0px 30px #1A237E:`

box shadows work in the same way as text shadows

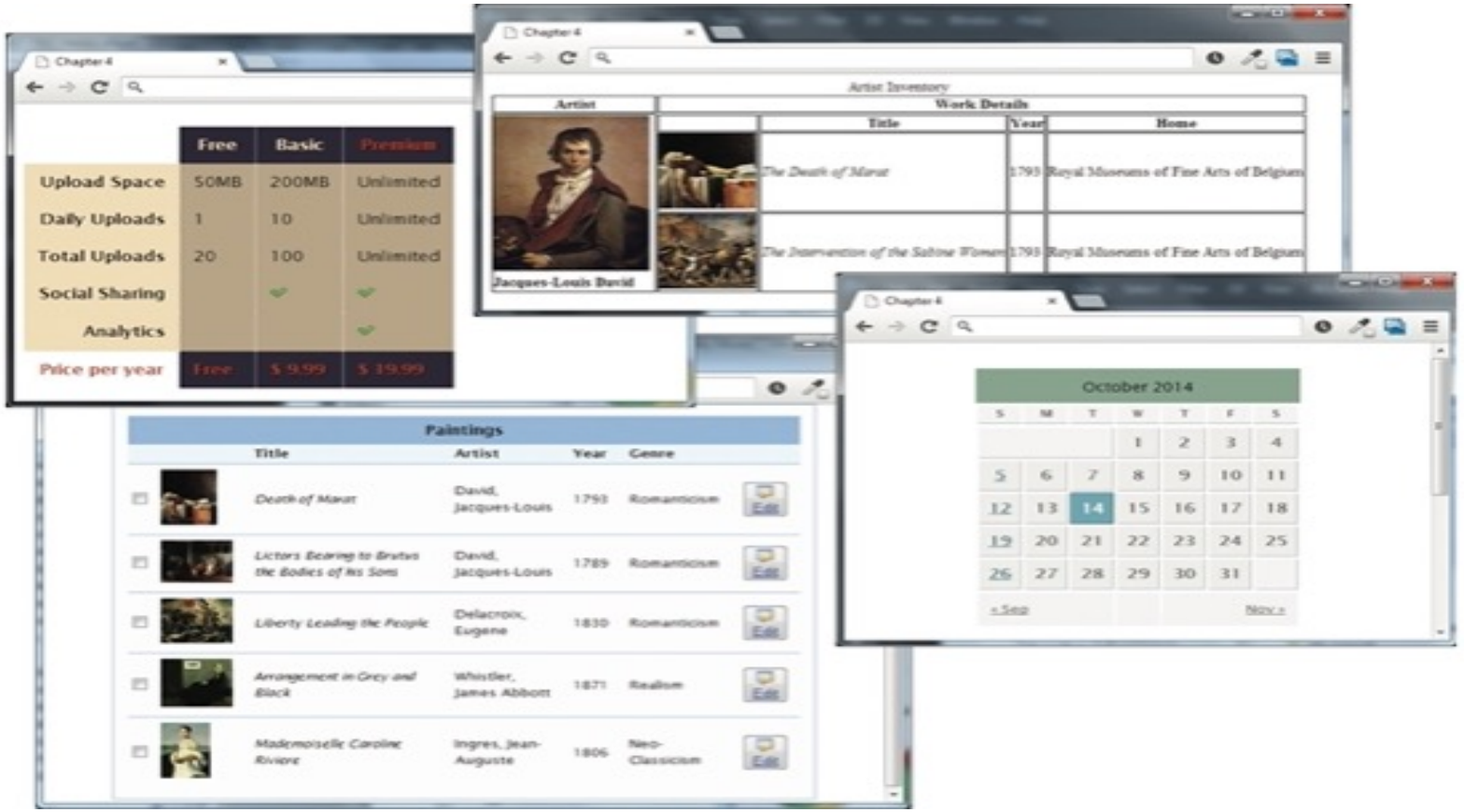
The End

Web System and Tecnologies CS 521

TOPIC 31

Introducing Tables

Basic Table Structure:



Introducing Tables

Basic Table Structure:

<table>

Title <th>	Artist <th>	Year <th>	Width <th>	Height <th>
The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

<tr>
<tr>
<tr>

```
<table>
    <tr>
        <th>Title</th>
        <th>Artist</th>
        <th>Year</th>
        <th>Width</th>
        <th>Height</th>
    </tr>
    <tr>
        <td>The Death of Marat</td>
        <td>Jacques-Louis David</td>
        <td>1793</td>
        <td>162cm</td>
        <td>128cm</td>
    </tr>
    <tr>
        <td>Burial at Ornans</td>
        <td>Gustave Courbet</td>
        <td>1849</td>
        <td>314cm</td>
        <td>663cm</td>
    </tr>
</table>
```

Introducing Tables

Adding Headings:

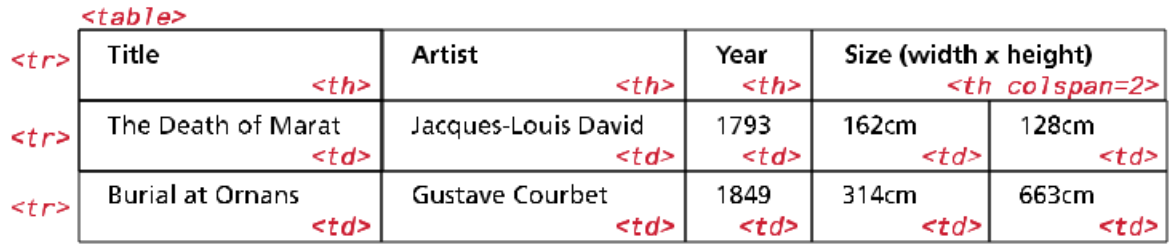
<table>

<tr>	Title <th>	Artist <th>	Year <th>	Width <th>	Height <th>
<tr>	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

```
<table>
    <tr>
        <th>Title</th>
        <th>Artist</th>
        <th>Year</th>
        <th>Width</th>
        <th>Height</th>
    </tr>
    <tr>
        <td>The Death of Marat</td>
        <td>Jacques-Louis David</td>
        <td>1793</td>
        <td>162cm</td>
        <td>128cm</td>
    </tr>
    <tr>
        <td>Burial at Ornans</td>
        <td>Gustave Courbet</td>
        <td>1849</td>
        <td>314cm</td>
        <td>663cm</td>
    </tr>
</table>
```

Introducing Tables

Spanning Columns:



Notice that this row now only has four cell elements.

```
<table>
   <tr>
      <th>Title</th>
      <th>Artist</th>
      <th>Year</th>
      <th colspan="2">Size (width x height)</th>
   </tr>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   ...
</table>
```

Introducing Tables

Spanning Rows:

<table>

Artist <th>	Title <th>	Year <th>	<tr>
Jacques-Louis David <td rowspan=3>	The Death of Marat <td>	1793 <td>	<tr>
	The Intervention of the Sabine Women <td>	1799 <td>	<tr>
	Napoleon Crossing the Alps <td>	1800 <td>	<tr>

```
<table>
    <tr>
        <th>Title</th>
        <th>Artist</th>
        <th>Year</th>
    </tr>
    <tr>
        <td rowspan="3">Jacques-Louis David</td>
        <td>The Death of Marat</td>
        <td>1793</td>
    </lr>
    <tr>
        <td>The Intervention of the Sabine Women</td>
        <td>1799</td>
    </tr>
    <tr>
        <td>Napoleon Crossing the Alps</td>
        <td>1800</td>
    </tr>
</table>
```

Notice that these two rows now only have two cell elements.

The End

Web Systems and Technologies

TOPIC 32

Introducing Tables (2)

Introducing Tables

- **Using Tables for Layout**


```
<table>
  <tr>
    <td>
      <img src="images/959.jpg" alt="Castle"/>
    </td>
    <td>
      <h2>Castle</h2>
      <p>Lewes, UK</p>
      <p>Photo by: Michele Brooks</p>
      <p>Built in 1069, the castle has a tremendous
         view of the town of Lewes and the
         surrounding countryside.
      </p>

      <h3>Other Images by Michele Brooks</h3>

      <table>
        <tr>
          <td><img src="images/484.jpg" /></td>
          <td><img src="images 537.jpg"  ></td>
        </tr>
        <tr>
          <td><img src="images/700.jpg" /></td>
          <td><img src='images/829.jpg' /></ td>
        </tr>
      </table>
    </td>
  < tr>
</table>
```

Introducing Tables

- ## Additional Table Elements

A title for the table is good for accessibility.

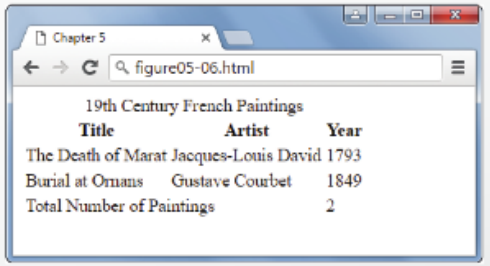
These describe our columns, and can be used to aid in styling.

Table header could potentially also include other <tr> elements.

Yes, the table footer comes *before* the body.

Potentially, with styling the browser can scroll this information, while keeping the header and footer fixed in place.

```
<table>
    <caption>19th Century French Paintings</caption>
    <col class="artistName" />
    <colgroup id="paintingColumns">
        <col />
        <col />
    </colgroup>

    <thead>
        <tr>
            <th>Title</th>
            <th>Artist</th>
            <th>Year</th>
        </tr>
    </thead>

    <tfoot>
        <tr>
            <td colspan="2">Total Number of Paintings</td>
            <td>2</td>
        </tr>
    </tfoot>

    <tbody>
        <tr>
            <td>The Death of Marat</td>
            <td>Jacques-Louis David</td>
            <td>1793</td>
        </tr>
        <tr>
            <td>Burial at Ornans</td>
            <td>Gustave Courbet</td>
            <td>1849</td>
        </tr>
    </tbody>

</table>
```



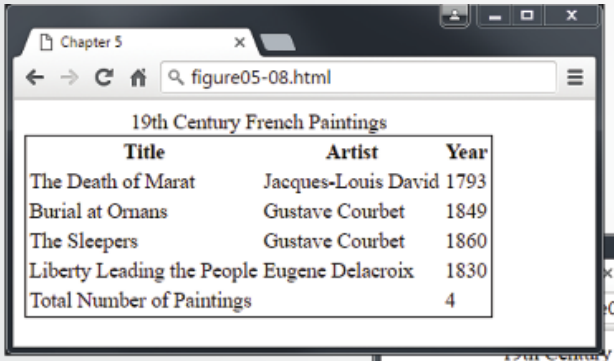
Web Systems and Technologies

TOPIC 33

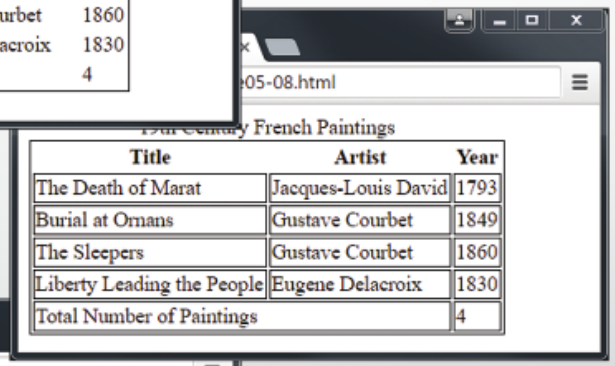
Styling Tables

Styling Tables

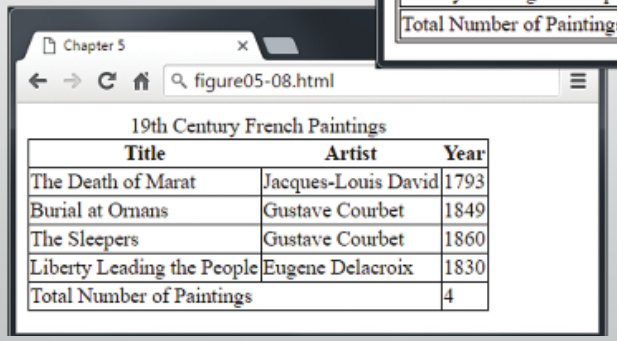
- Table Borders




```
table {
   border: solid 1pt black;
}
```



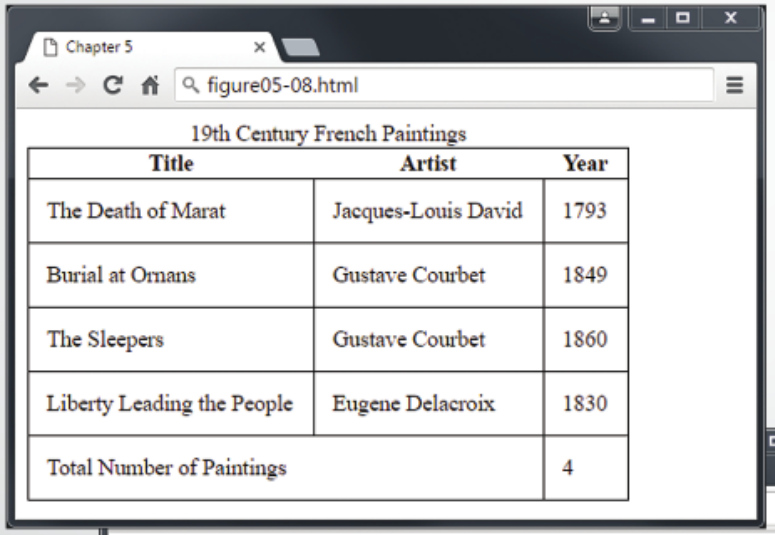

```
table {
   border: solid 1pt black;
}
td {
   border: solid 1pt black;
}
```



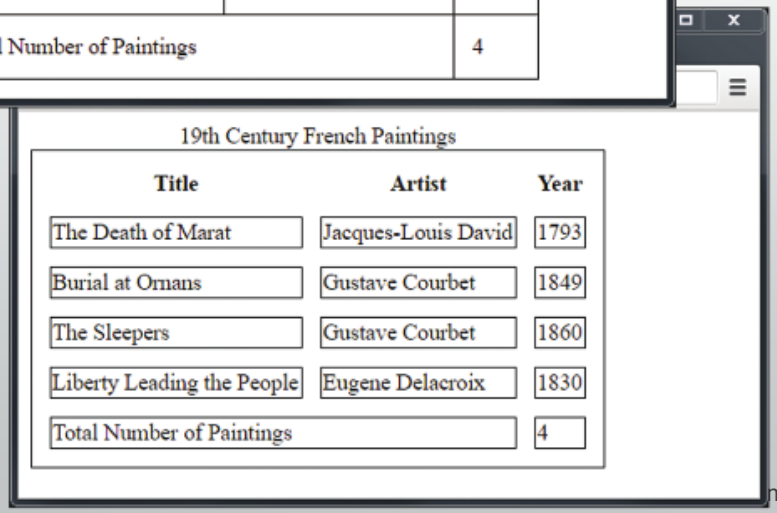

```
table {
   border: solid 1pt black;
   border-collapse: collapse;
}
td {
   border: solid 1pt black;
}
```

Styling Tables

- Table Borders



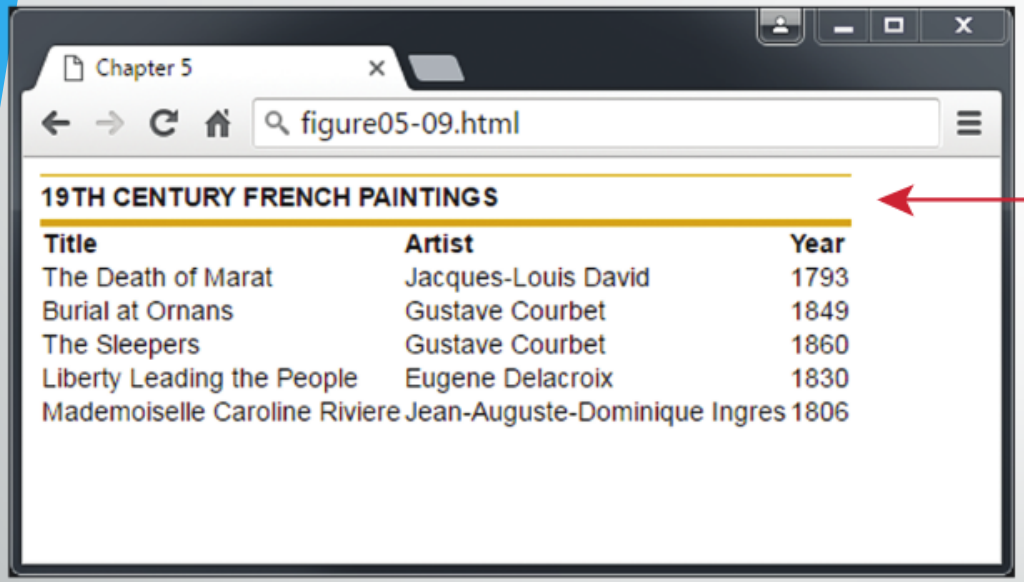

```
table {
   border: solid 1pt black;
   border-collapse: collapse;
}
td {
   border: solid 1pt black;
   padding: 10pt;
}
```




```
table {
   border: solid 1pt black;
   border-spacing: 10pt;
}
td {
   border: solid 1pt black;
}
```

Styling Tables

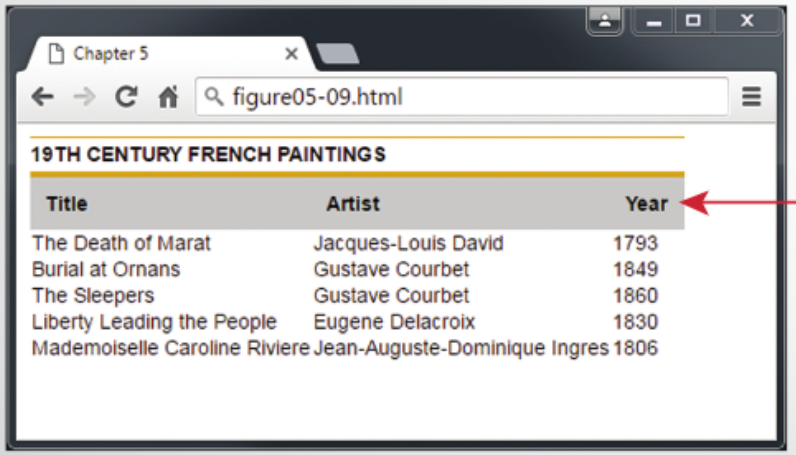
- Boxes




```
caption {
    font-weight: bold;
    padding: 0.25em 0 0.25em 0;
    text-align: left;
    text-transform: uppercase;
    border-top: 1px solid #DCA806;
}
table {
    font-size: 0.8em;
    font-family: Arial, sans-serif;
    border-collapse: collapse;
    border-top: 4px solid #DCA806;
    border-bottom: 1px solid white;
    text-align: left;
}
```

Styling Tables

- Boxes




```
thead tr {
    background-color: #CACACA;
}
th {
    padding: 0.75em;
}
```

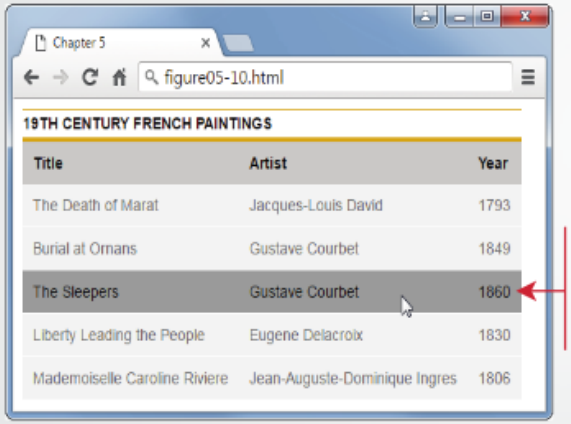
Styling Tables

- **Boxes**

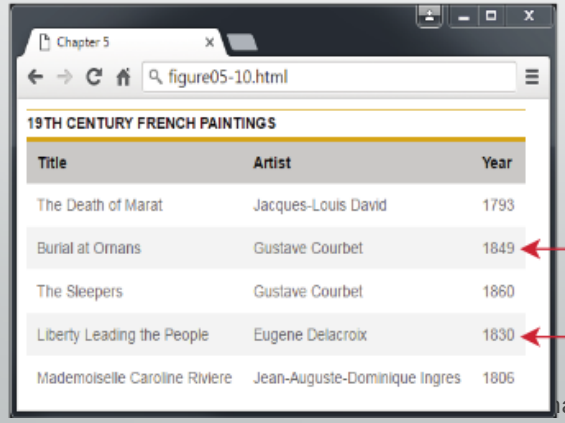

```
tbody tr {
   background-color: #F1F1F1;
   border-bottom: 1px solid white;
   color: #6E6E6E;
}
tbody td {
   padding: 0.75em;
}
```

Styling Tables

- Hover & Zebras




```
tbody tr:hover {
   background-color: #9e9e9e;
   color: black;
}
```




```
tbody tr:nth-child(even) {
   background-color: lightgray;
}
```

Styling Tables

- **Accessible Tables**
- Use tables for data, not layout
- Use the <caption> element
- Connect cells with a textual description in the header

```
<caption>Famous Paintings</caption>
<tr>
  <th scope="col">Title</th>
  <th scope="col">Artist</th>
  <th scope="col">Year</th>
  <th scope="col">Width</th>
  <th scope="col">Height</th>
</tr>
```

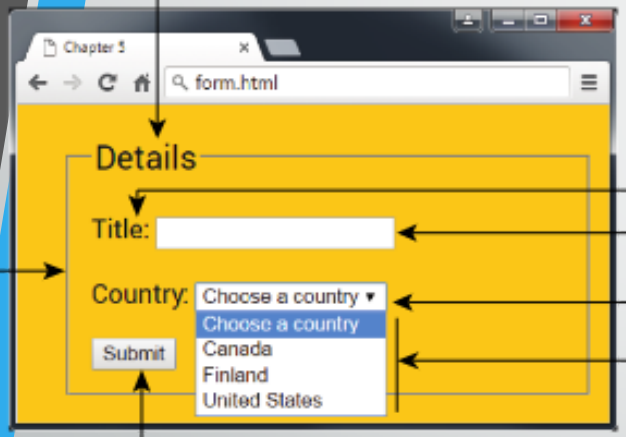
Web Systems and Technologies

TOPIC 34

Introducing Forms

Introducing Forms

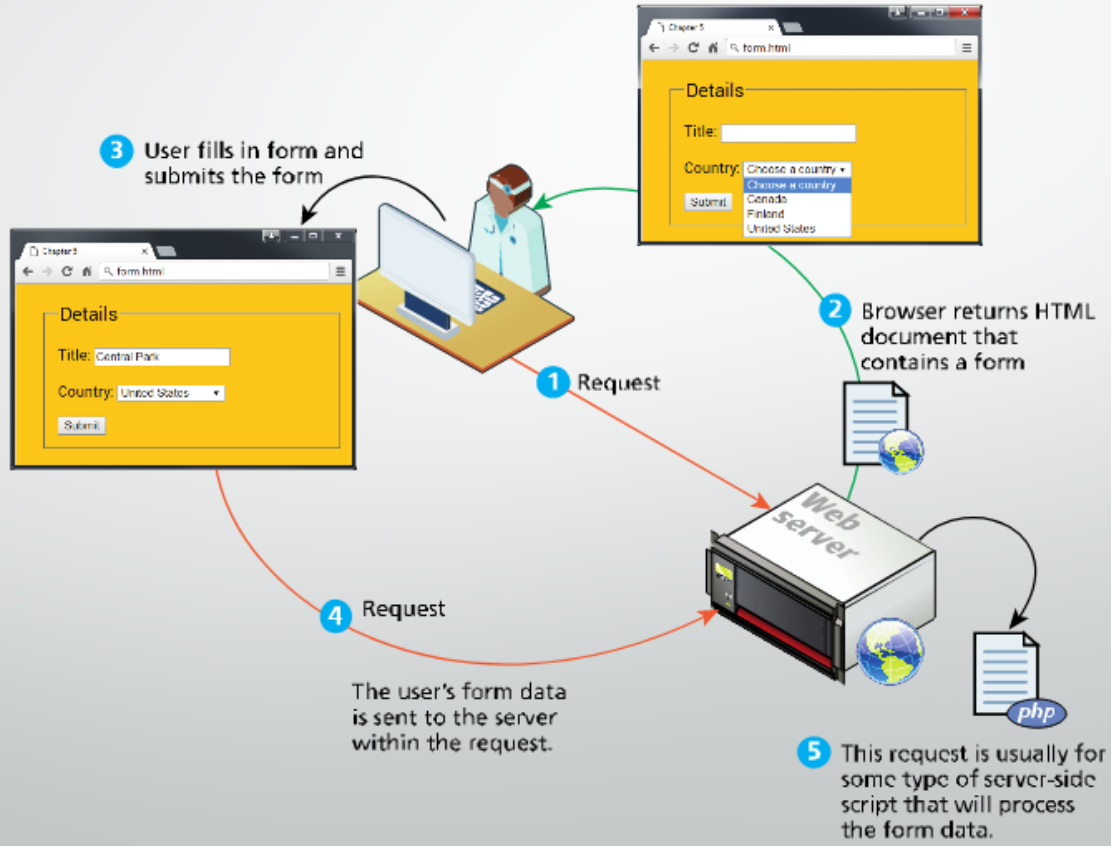
- Form Structure




```
<form method="post" action="process.php">
 <fieldset>
  <legend>Details</legend>
  <p>
   <label>Title: </label>
   <input type="text" name="title" />
  </p>
  <p>
   <label>Country: </label>
   <select name="where">
    <option>Choose a country</option>
    <option>Canada</option>
    <option>Finland</option>
    <option>United States</option>
   </select>
  </p>
  <input type="submit" />
 </fieldset>
</form>
```

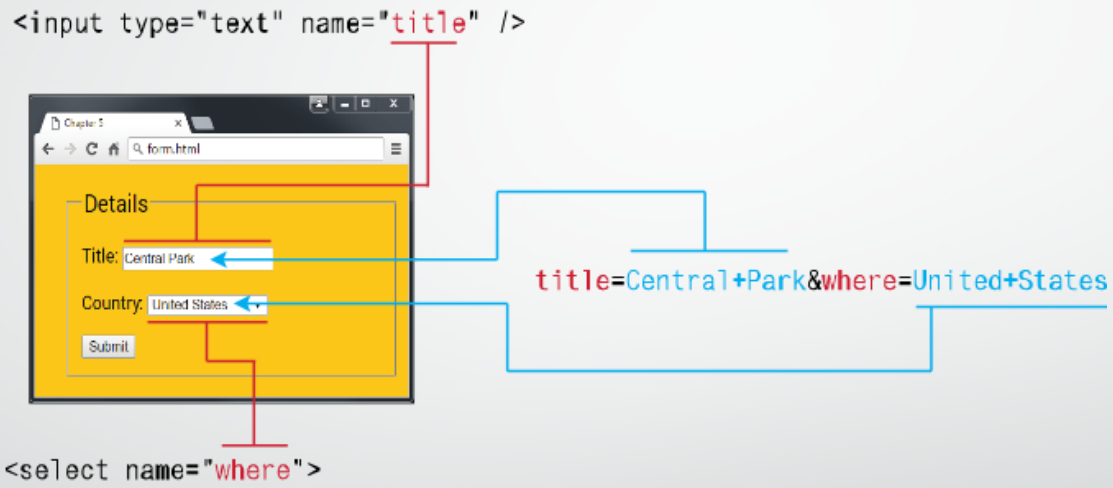
Introducing Forms

- How Forms Work



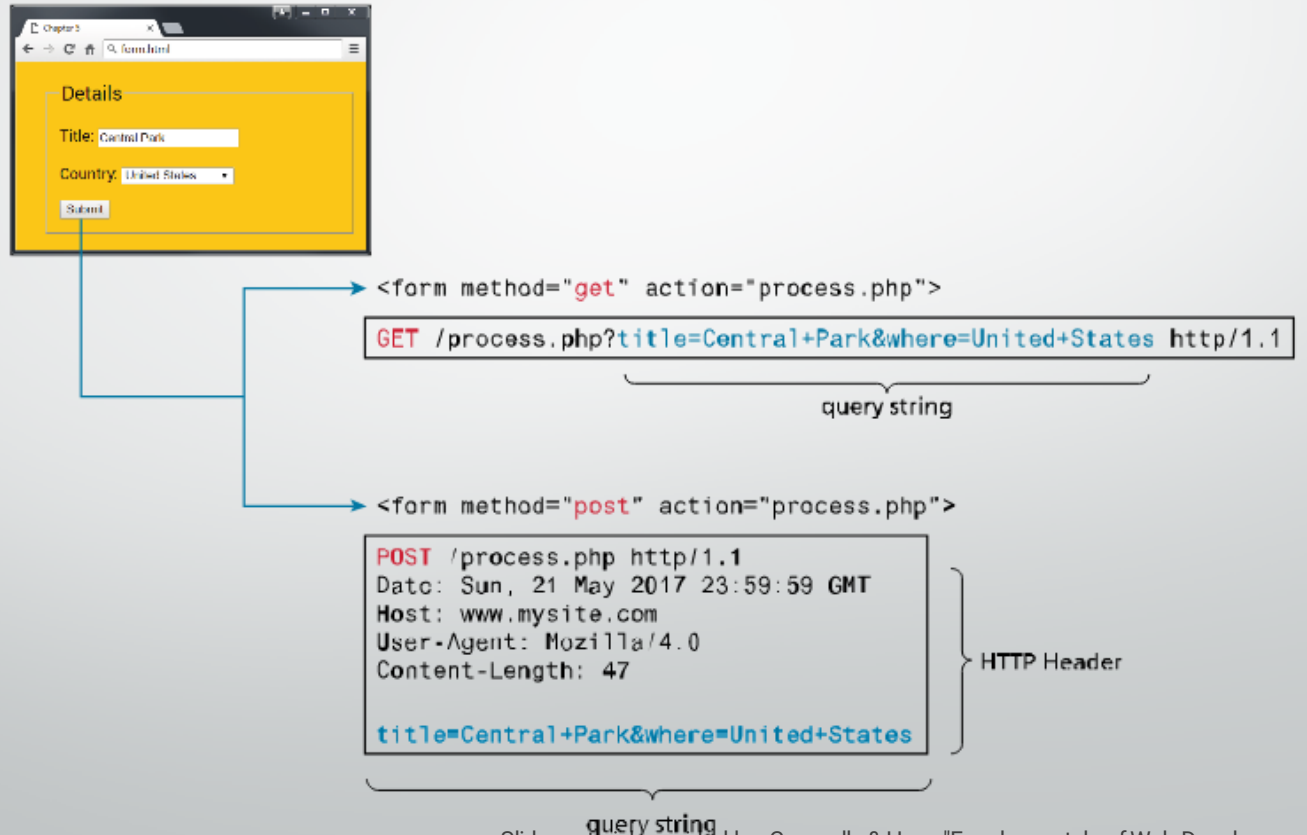
Introducing Forms

- **Query Strings**



Introducing Forms

- GET and POST



Introducing Forms

- **HTML Form elements**

  <form>
  <button>
  <input>
  <label>
  <datalist>
  <fieldset>
  <legend>
  <select>
  <option>
  <optgroup>
  <textarea>
  <output>

Web Systems and Technologies
TOPIC 35

Form Control Elements (1)

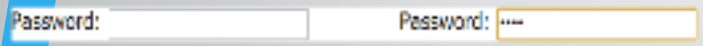
Form Control Elements

- Text Input Controls

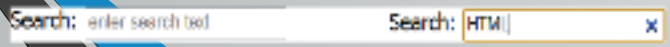
text

textarea

password

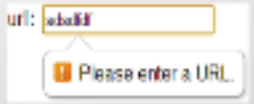


search

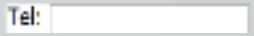


email

url



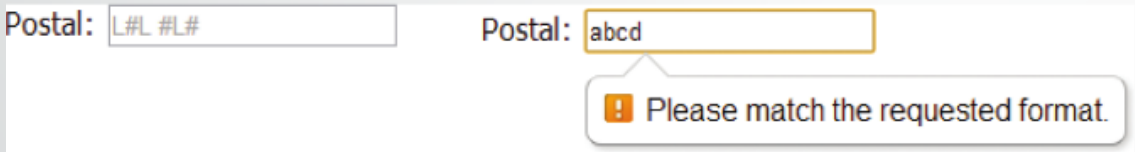
tel

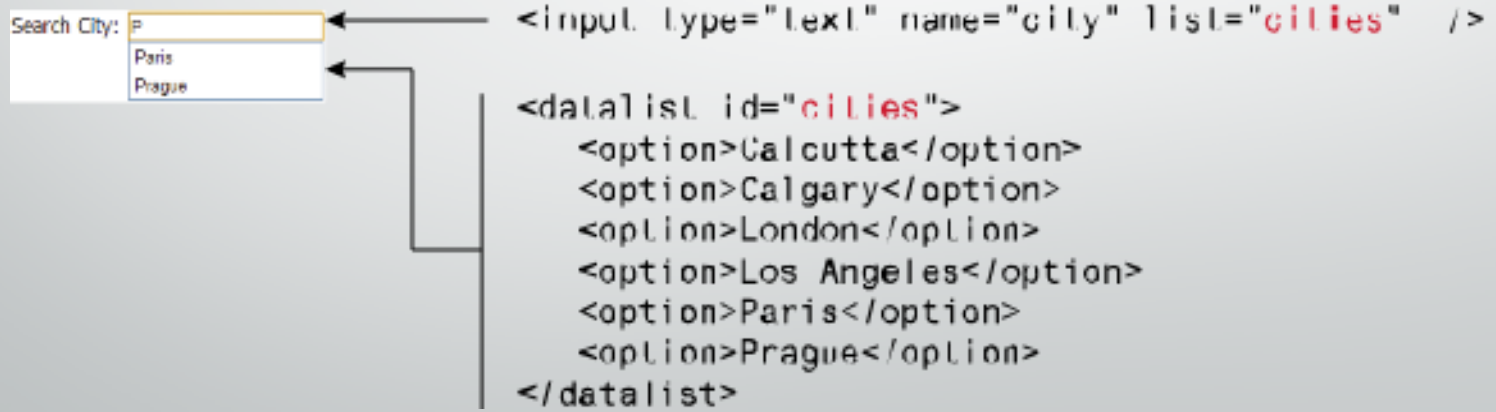


Form Control Elements

- New in HTML5 – pattern and datalist

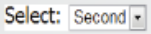
```
<input type="text" ... placeholder="L#L #L#" pattern="[a-z][0-9][a-z] [0-9][a-z][0-9]"  />
```

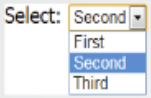




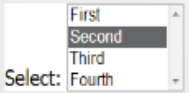
Form Control Elements

- **Choice Controls (<select>)**

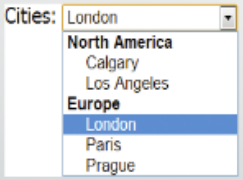





```
<select name="choices">
  <option>First</option>
  <option selected>Second</option>
  <option>Third</option>
</select>
```



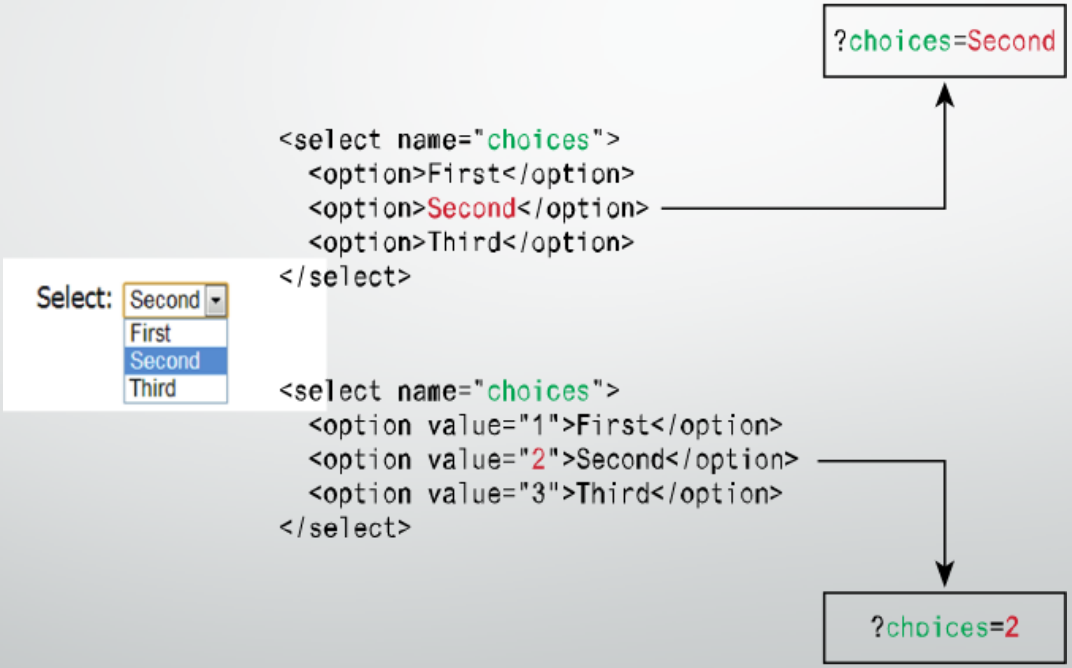

```
<select size="3" ... >
```




```
<select ... >
 <optgroup label="North America">
   <option>Calgary</option>
   <option>Los Angeles</option>
 </optgroup>
 <optgroup label="Europe">
   <option>London</option>
   <option>Paris</option>
   <option>Prague</option>
 </optgroup>
</select>
```

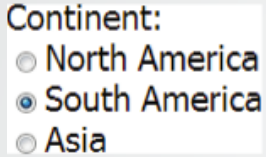
Form Control Elements

- **Choice Controls <select> using value attribute**



Form Control Elements

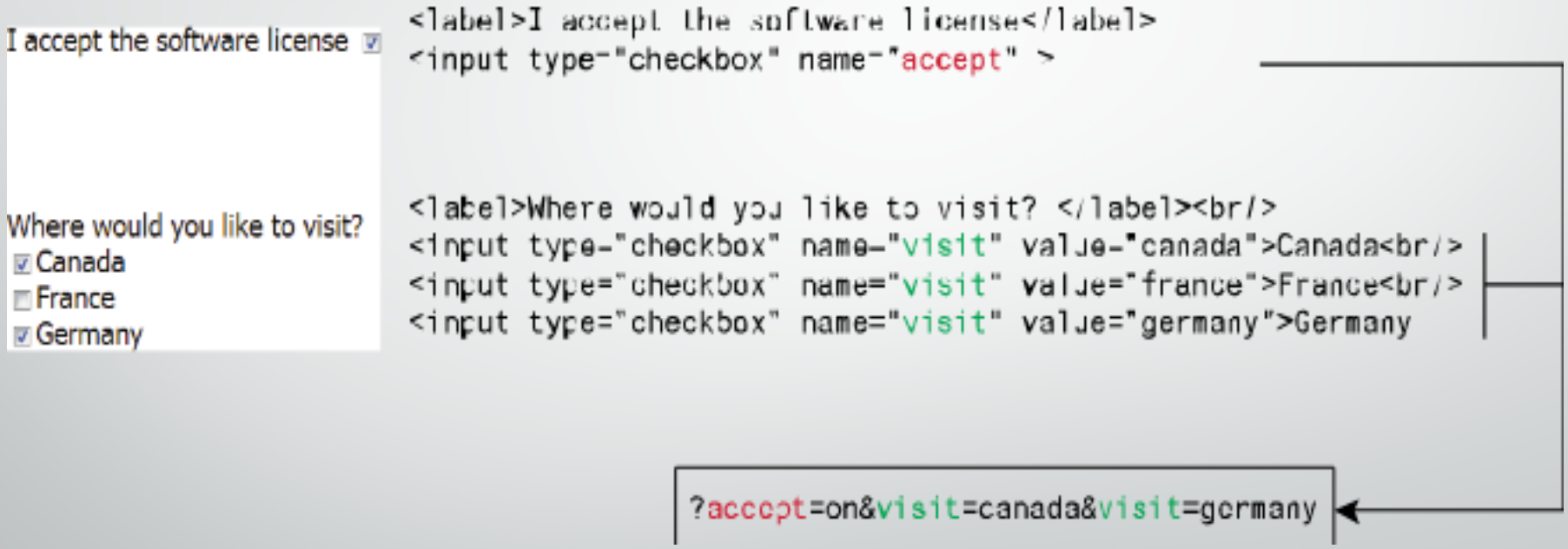
- **Button Controls (radio)**




```
<input type="radio" name="where" value="1">North America<br/>
<input type="radio" name="where" value="2" checked>South America<br/>
<input type="radio" name="where" value="3">Asia
```

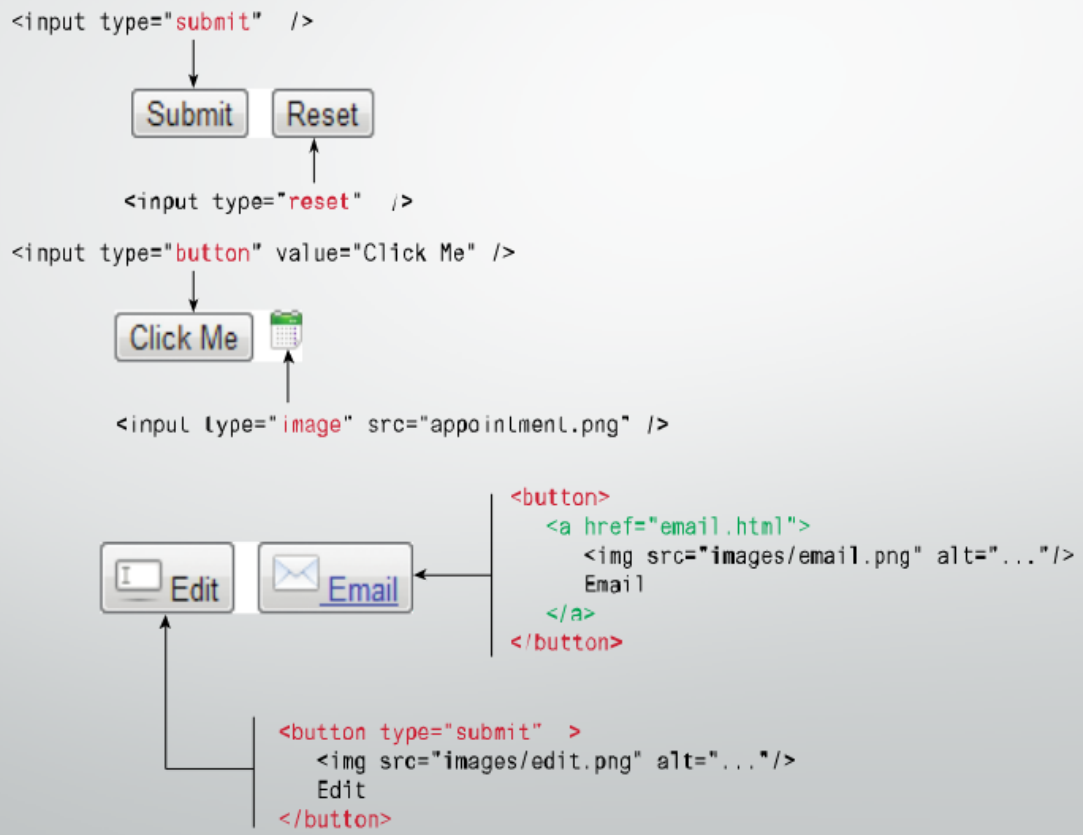
Form Control Elements

- **Button Controls (checkbox)**



Form Control Elements

- **Button Controls (checkbox)**



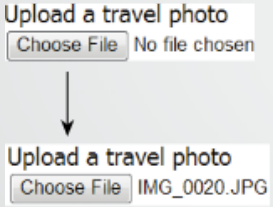
Web Systems and Technologies

TOPIC 36

Form Control Elements (2)

Form Control Elements

- **Specialized Controls (file upload)**

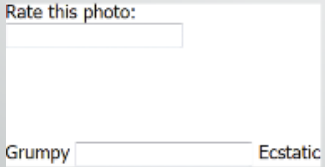



```
<form method="post" enctype="multipart/form-data" ... >
   ...
   <label>Upload a travel photo</label>
   <input type="file" name="photo" />
   ...
</form>
```

Form Control Elements

- Specialized Controls (number and range controls)


```
<label>Rate this photo: <br/>
<input type="number" min="1" max="5" name="rate" />

Grumpy
<input type="range" min="0" max="10" step="1" name="happiness" />
Ecstatic
```



Controls as they appear in browser that doesn't support these input types

Form Control Elements

- **Specialized Controls (colour)**

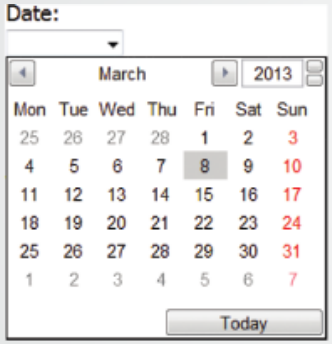

```
<label>Background Color: <br/>
<input type="color" name="back" />
```

Background Color:

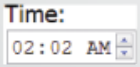
Control as it appears in browser that doesn't support this input type

Form Control Elements

- **Date and Time Controls**

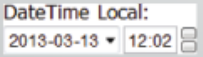



```
<label>Date: <br/>
<input type="date" ... />
```




```
<input type="time" ... />
```

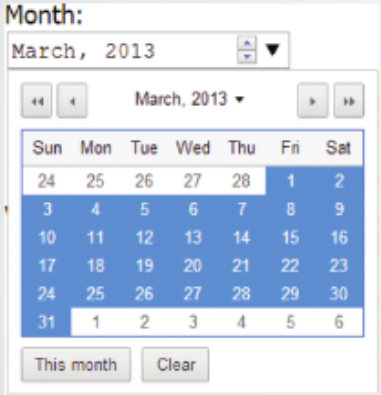

```
<input type="datetime" ... />
```



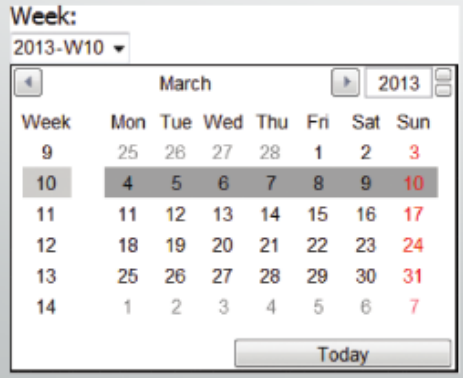

```
<input type="datetime-local" ... />
```

Form Control Elements

- **Date and Time Controls**




```
<input type="month" ... />
```




```
<input type="week" ... />
```

Form Control Elements

- **Accessible Form**

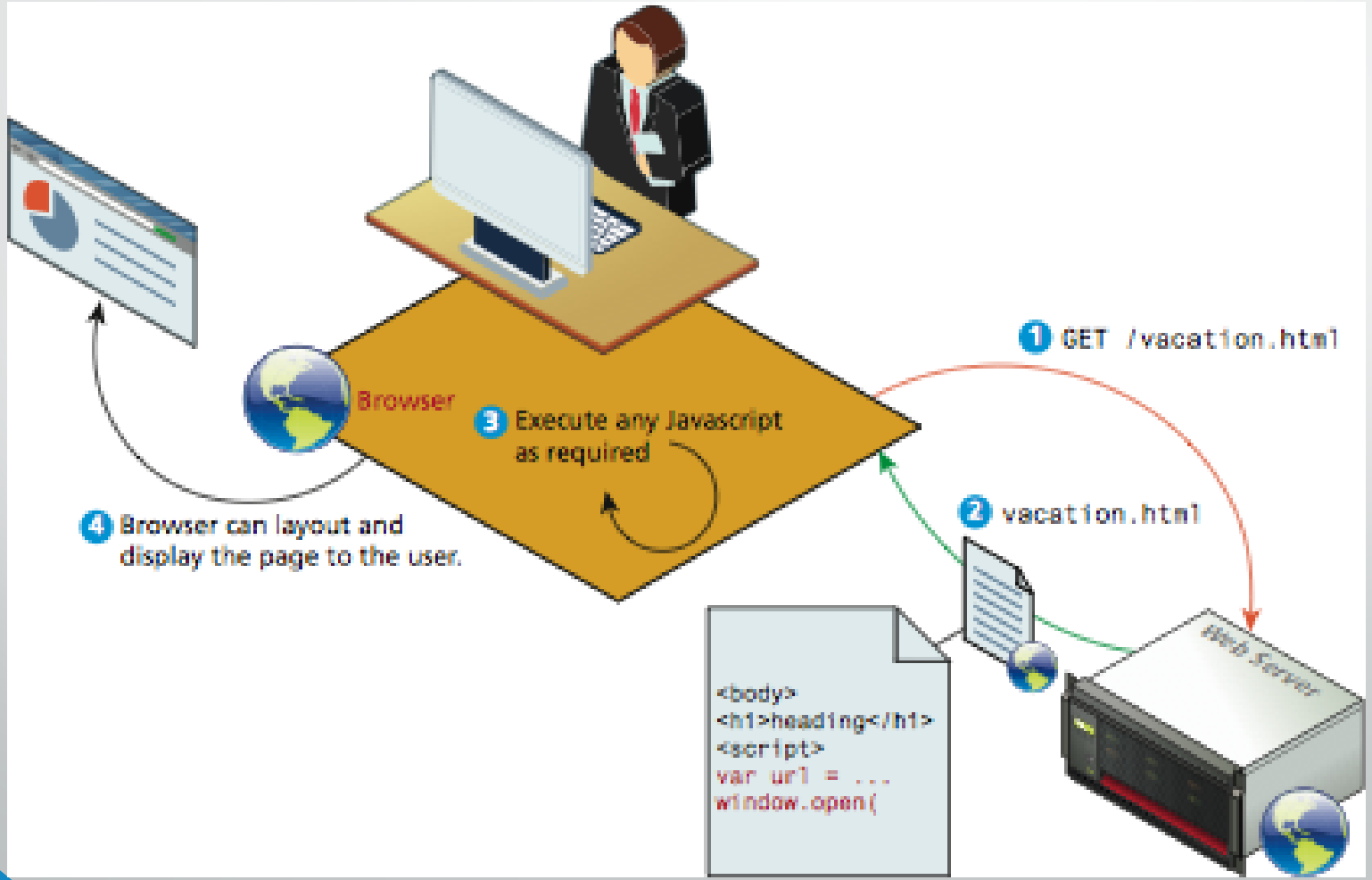
```
<label for="f-title">Title: </label>

<input type="text" name="title" id="f-title"/>


<label for="f-country">Country: </label>

<select name="where" id="f-country">
    <option>Choose a country</option>
    <option>Canada</option>
    <option>Finland</option>
    <option>United States</option>
  </select>
```

Web System and Tecnologies CS 521

WEEK 05 TOPIC 37 TO 45

Introduction to JavaScript

Client Side Scripting



Introduction to JavaScript

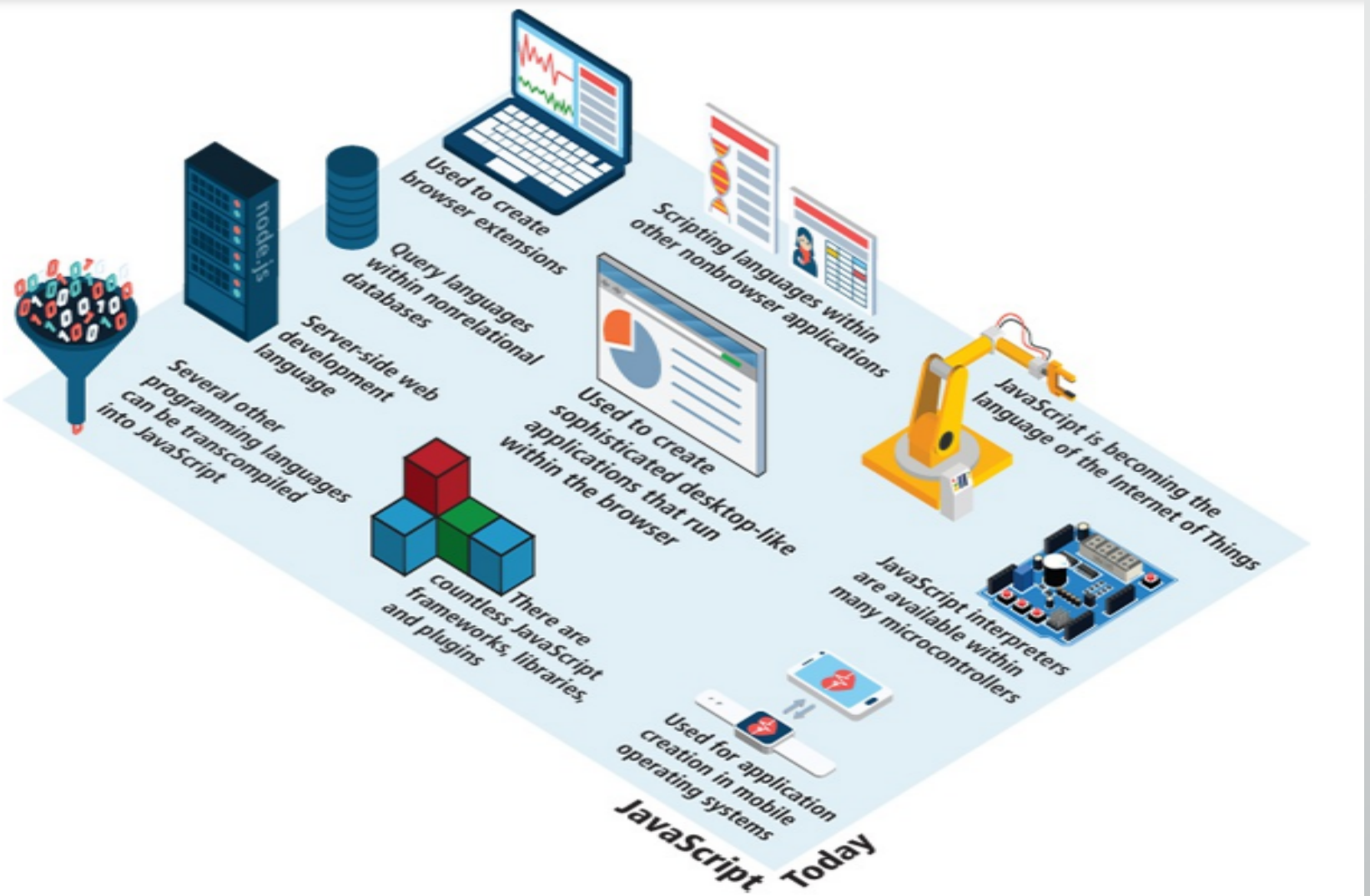
JavaScript History

- JavaScript was introduced by Netscape in their Navigator browser back in 1996
- JavaScript that is supported by your browser contains language features
  - not included in the current ECMAScript specification and
  - missing certain language features from that specification
- Commonly used version of ECMAScript is the Sixth Edition (ES6)
  - 12th Edition (ECMAScript 2021) released in 2021

JavaScript and Web 2.0

- Early JavaScript had only a few common uses
- 2000s onward saw more sophisticated uses for JavaScript
- AJAX as both an acronym and a general term

JavaScript in Contemporary Software Development



Web System and Tecnologies CS 521

Writing and Storing the Script

Inline JavaScript

- Inline JavaScript refers to the practice of includingJavaScript code directly within certain HTML attributes

```
<a href="javaScript:OpenWindow();">more info</a>
```

```
<input type="button" onClick="alert('Are you sure?');" />
```

Writing and Storing the Script

Embedded JavaScript

- Embedded JavaScript refers to the practice of placing JavaScript code within a <script> element

```
<script type="text/javascript">
        /* A JavaScript Comment */
        alert("Hello World!");
</script>
```

Writing and Storing the Script

External JavaScript

- External JavaScript files typically contain function definitions, data definitions, and entire frameworks.

```
<head>
    <script type="text/javascript" src="greeting.js"></script>
</head>
```

Writing and Storing the Script

Users without JavaScript

- Web crawler
- Browser plug-in
- Text-based client
- Visually disabled client
- The <NoScript> Tag

Introduction to JavaScript

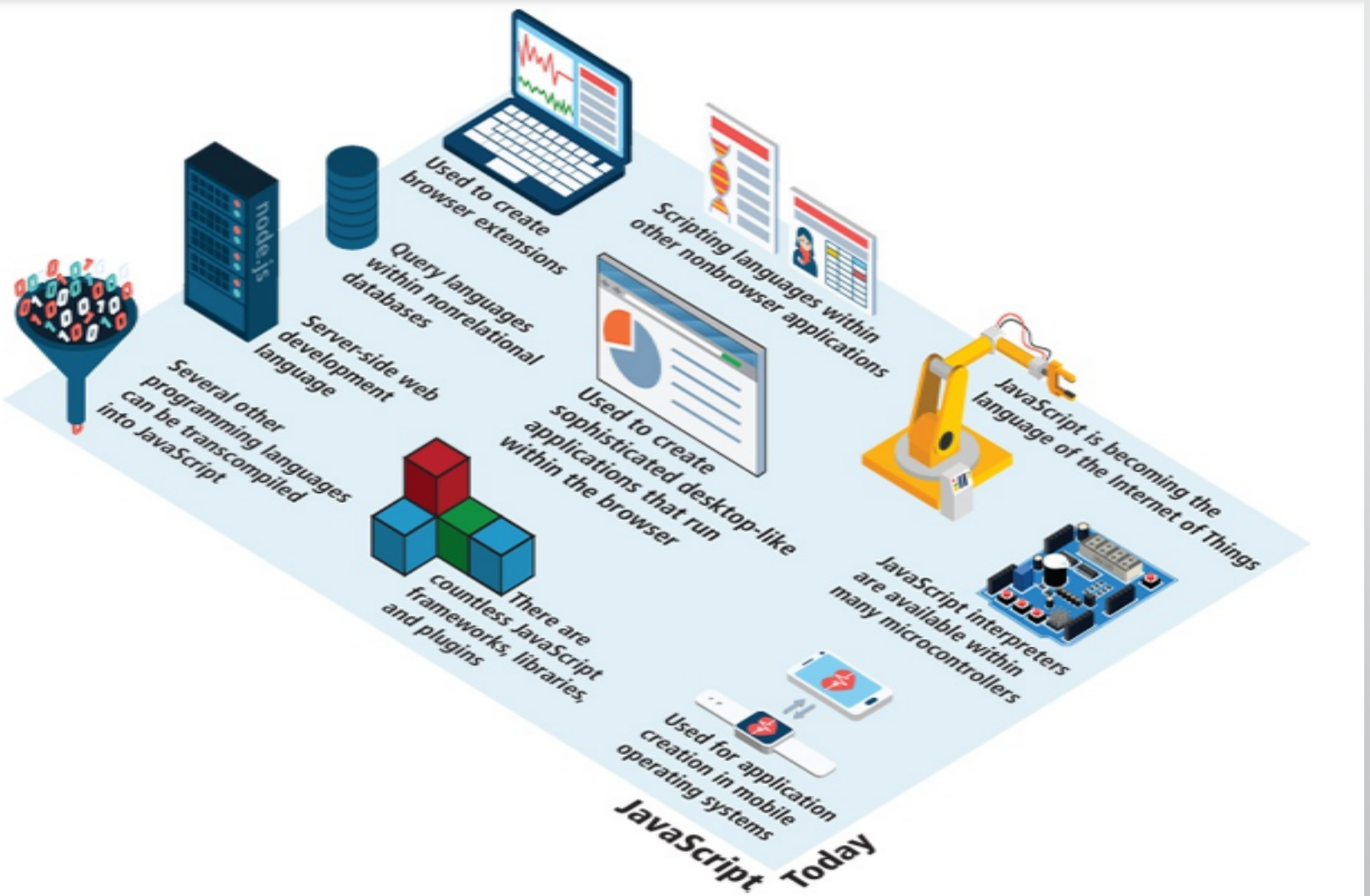
JavaScript History

- JavaScript was introduced by Netscape in their Navigator browser back in 1996
- JavaScript that is supported by your browser contains language features
  - not included in the current ECMAScript specification and
  - missing certain language features from that specification
- Commonly used version of ECMAScript is the Sixth Edition (ES6)
  - 12th Edition (ECMAScript 2021) released in 2021

JavaScript and Web 2.0

- Early JavaScript had only a few common uses
- 2000s onward saw more sophisticated uses for JavaScript
- AJAX as both an acronym and a general term

JavaScript in Contemporary Software Development

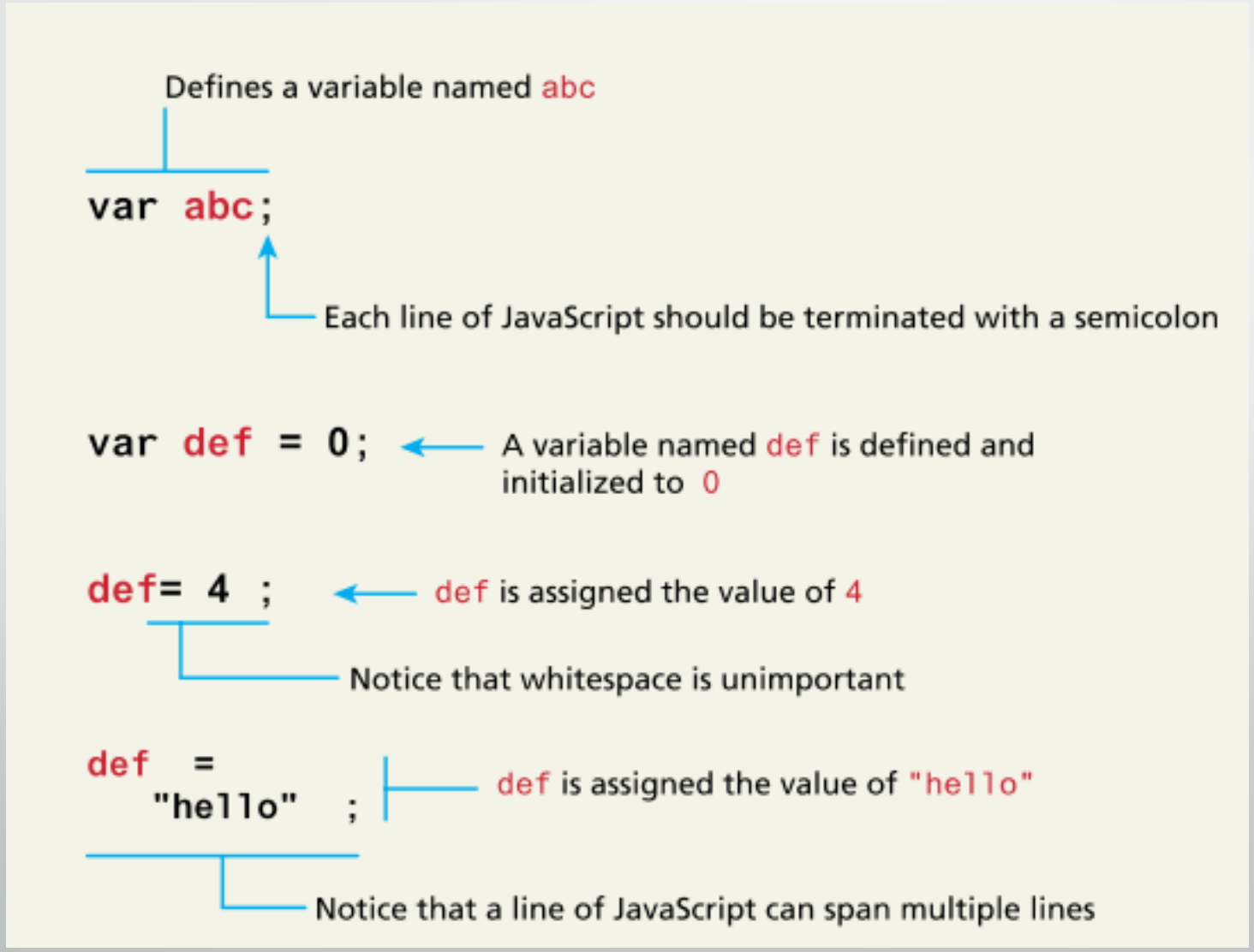


Variables and Data Types

Variables

- Variables in JavaScript are dynamically typed
- This simplifies variable declarations, since we do not require the familiar data-type identifiers
- Instead we simply use the var keyword

Example variable declarations and Assignments



Data Types

for which purpose we use data types?
answer. storage format.

- Two basic data types:
  - Reference types (usually referred to as objects) and
  - primitive types (Boolean, Number, String, …)

Reference Types

```
var abc = 27;
var def = "hello";
```

variables with primitive types

```
var foo = [45, 35, 25];
```

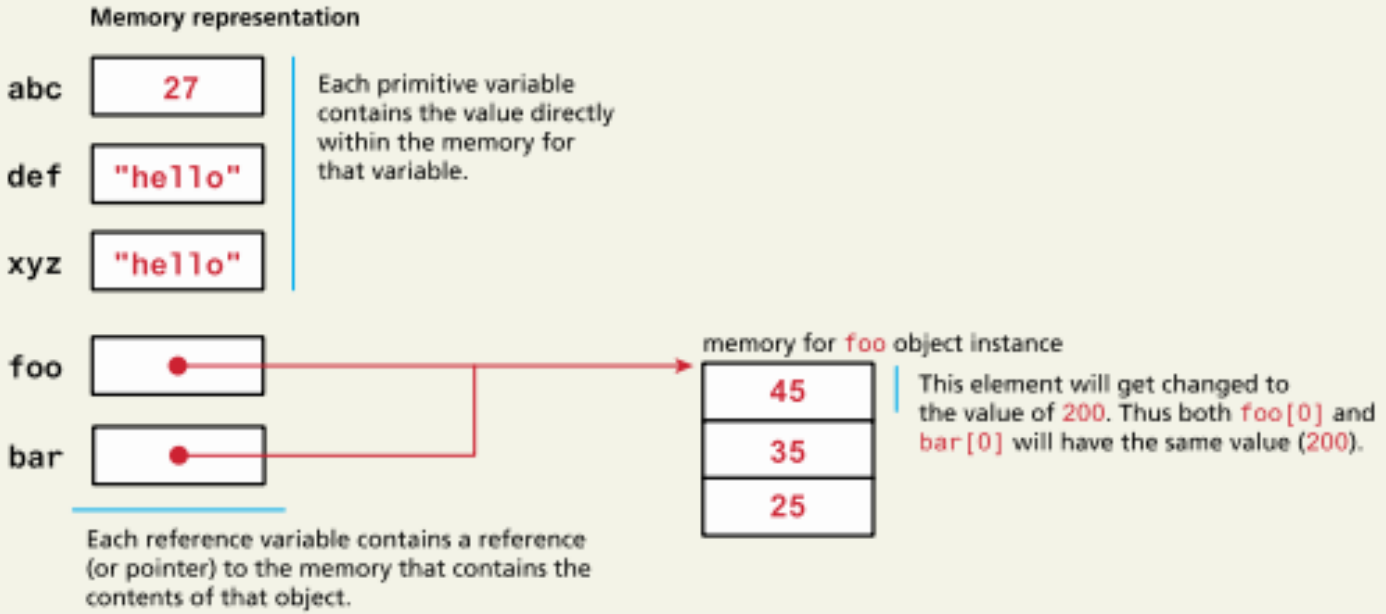
variable with reference type (i.e., array object)

```
var xyz = def;
var bar = foo;
```

these new variables differ in important ways (see below)

```
bar[0] = 200;
```

changes value of the first element of array



JavaScript Output

Output

- alert()
  - Displays content within a pop-up box.
- document.write()
  - Outputs the content (as markup)
- console.log()
  - Displays content in the Browser’s JavaScript console.

Message (alert box)

- Syntax
  - alert(“Hello World”);
- Avoid frequent use

Document

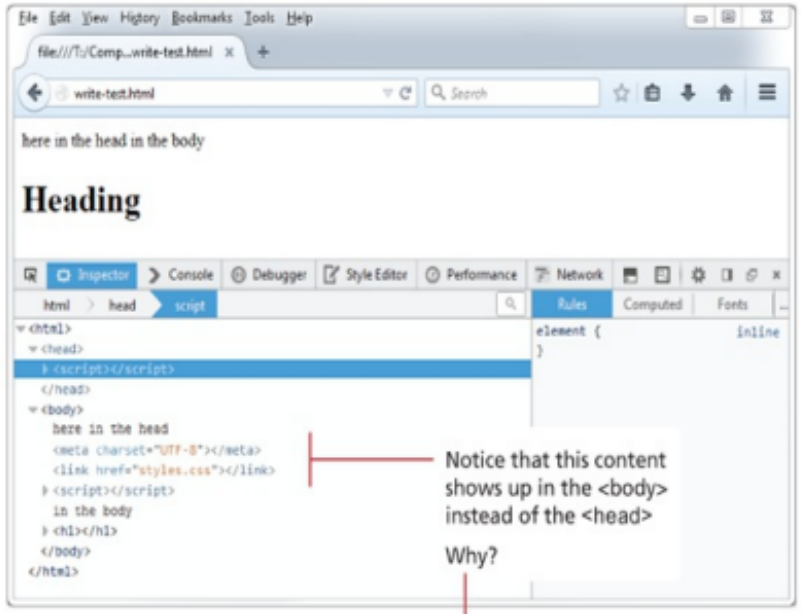
- Use the JavaScript's document object

```
var name = "Randy";
document.write("<h1>Title</h1>");
// this uses the concatenate operator (+)
document.write("Hello " + name + " and welcome");
```

Fun with document.write()

```
<html>
<head>
<script>
    document.write('here in the head');

    document.write('<meta charset="UTF-8">');
    document.write('<link href=styles.css>');
</script>
</head>
<body>
<script>
    document.write("in the body");
    document.write("<h1>Heading</h1>");
</script>
</body>
</html>
```

We want this to appear here in the <head>



Generated content

Notice that this content shows up in the <body> instead of the <head>

Why?

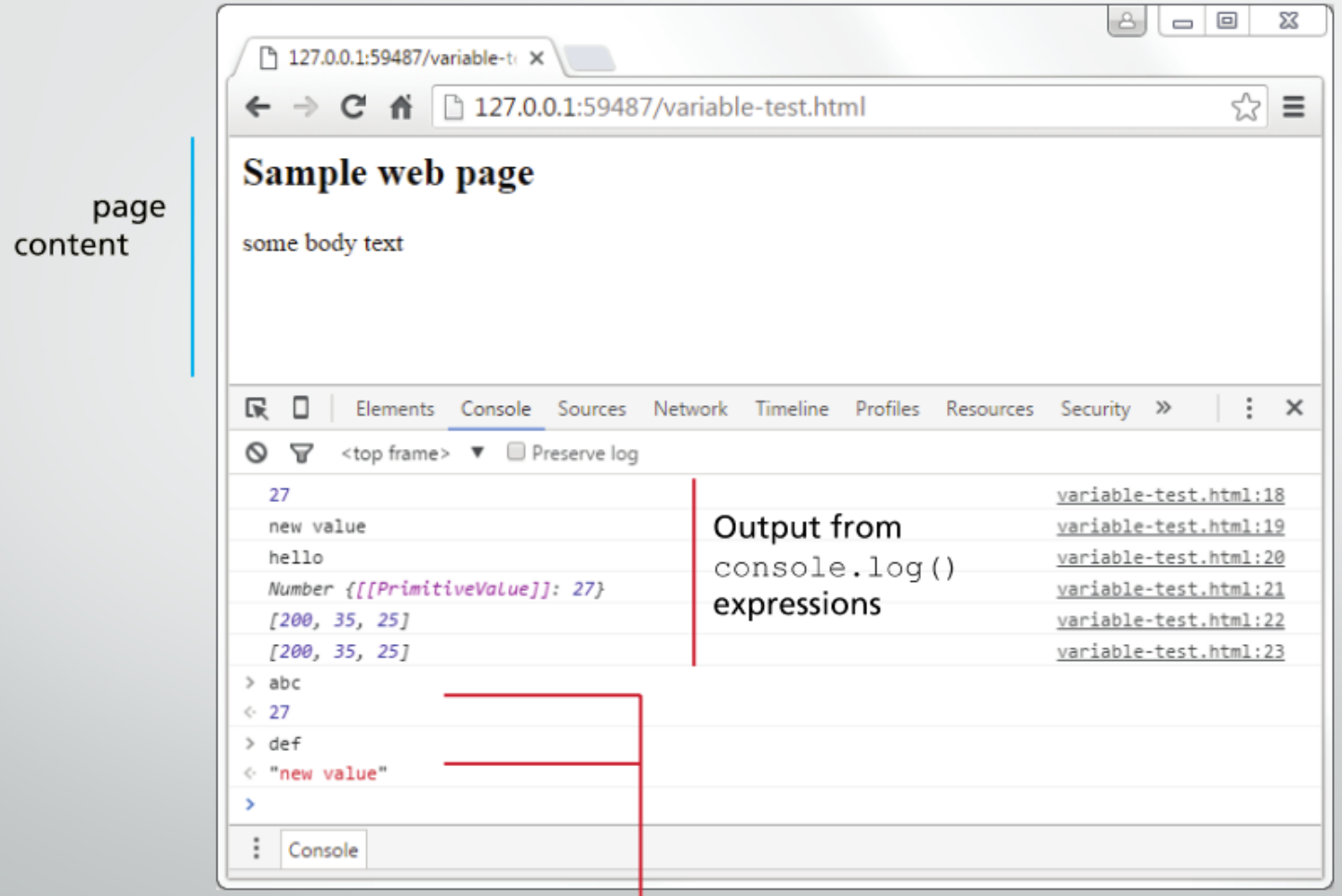
Browser Inspector displays HTML content that is being displayed (static and dynamic)

The appearance of this line will shift the following `write()` calls to the <body>

Console's Log

- Allmost all modern browsers provide a JavaScript for developers and debugging
- Use console.log() to display output in the console

Chrome JavaScript Console



Conditions

Conditional Statements

if then code given output btni thi .

- If ... else if ... else
- Switch
- Conditional Assignment

Conditional Statements

```
if (hourOfDay > 4 && hourOfDay < 12) {
        greeting = "Good Morning";
}
else if (hourOfDay >= 12 && hourOfDay < 18) {
        greeting = "Good Afternoon";
}
else {
        greeting = "Good Evening";
}
```

Switch Statement

```
switch (artType) {
        case "PT":
                output = "Painting";
                break;
        case "SC":
                output = "Sculpture";
                break;
        default:
        output = "Other";
}
```

Conditional Assignment

```
/* x conditional assignment */
x = (y==4) ? "y is 4" : "y is not 4";
```

Condition — Value if true — Value if false

```
/* equivalent to */
if (y==4) {
   x = "y is 4";
}
else {
   x = "y is not 4";
}
```

Truthy and Falsy

In JavaScript, a value is said to be **truthy** if it translates to true, while a value is said to be **falsy** if it translates to false.

- Almost all values in JavaScript are truthy
- false, null, "", '', 0, NaN, and undefined are falsy

Arrays

array start with which index.

Arrays are one of the most commonly used data structures in programming.

JavaScript provides two main ways to define an array.

- object literal notation
- use the Array() constructor

Arrays (Object Literal Notation)

The literal notation approach is generally preferred since it involves less typing, is more readable, and executes a little bit quicker

```
var years = [1855, 1648, 1420];

var countries = ["Canada", "France",
                 "Germany", "Nigeria",
                 "Thailand", "United States"];

var mess = [53, "Canada", true, 1420];
```

Array Constructor

- We can use the Array constructor to create an array

```
var countries = new Array("Canada",          "France"
,"Germany",          "Nigeria","Thailand", "United
            States");

var A = new Array(5);
```

Arrays

- Some common features

arrays in JavaScript are zero indexed

[] notation for access

.length gives the length of the array

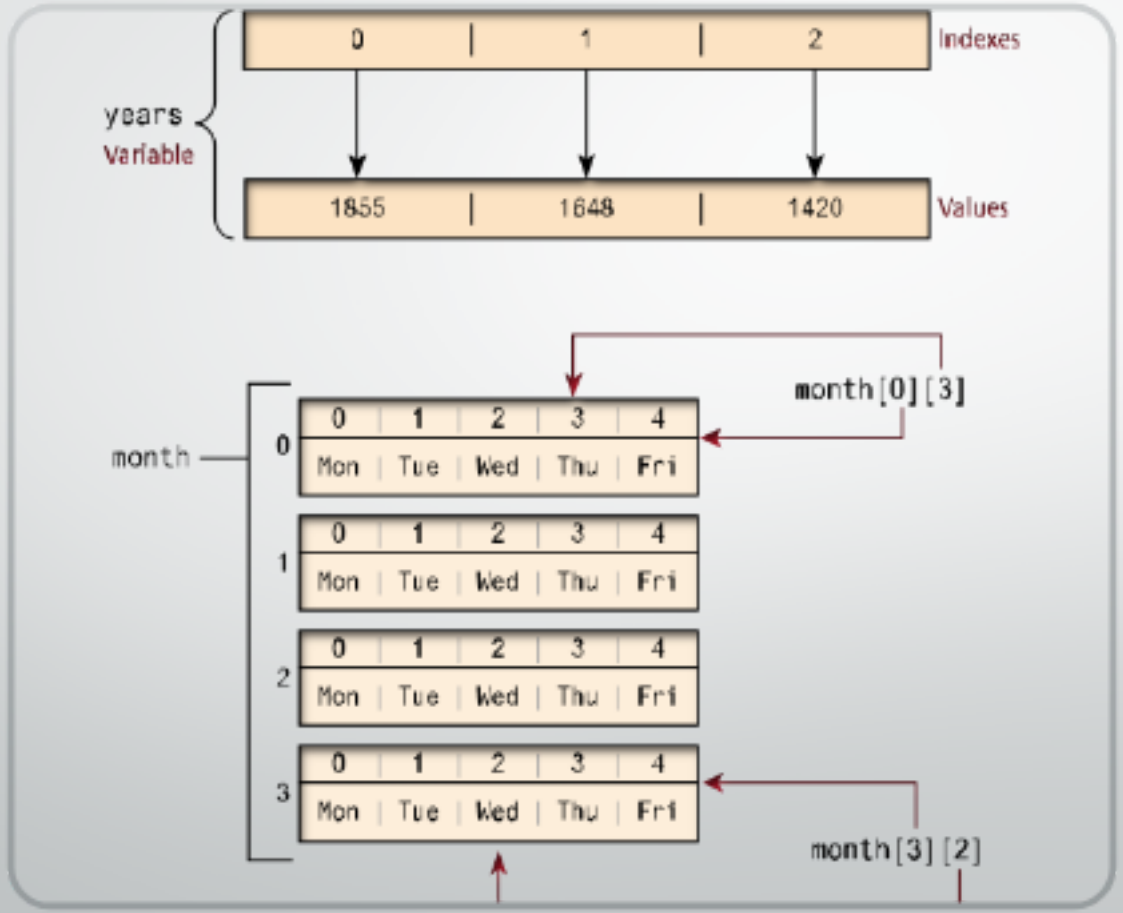
.push()

.pop()

concat(), slice(), join(), reverse(), shift(), and sort()

Arrays

- Arrays Illustrated



Loops

- Types of Loops

Used for repetitively executing statements

while

do while

for

forEach

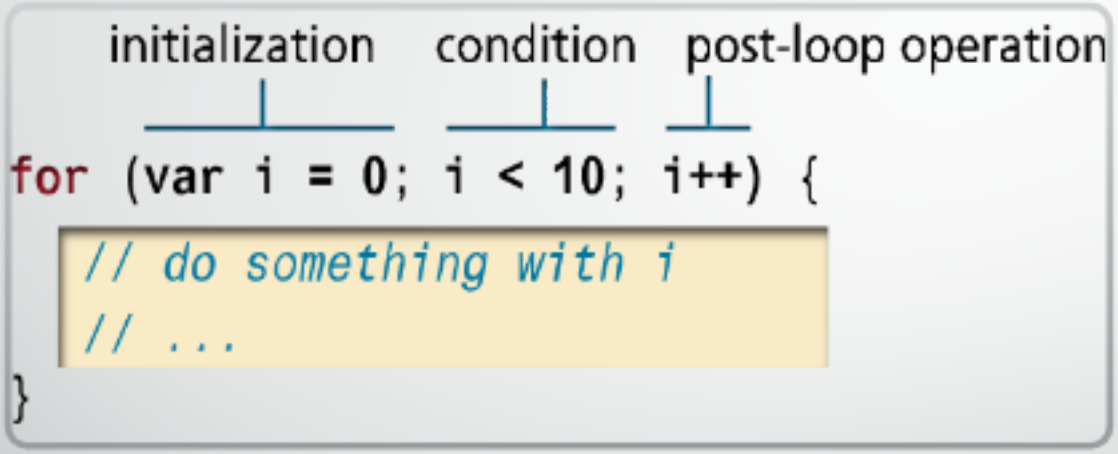
Loops

- While and do . . . while Loops

```
var count = 0;
while (count < 10) {
    // do something
    // ...
    count++;
    }
count = 0;
do {
// do something
    // ...
    count++;
} while (count < 10);
```

Loops

- For Loop



Loops

- ForEach Loop

- forEach is used with an array
- Processes each element of an array

```
var countries = ["Canada", "France",
                 "Germany", "Nigeria",
                 "Thailand", "United States"];
countries.forEach( function(country) {
                 console.log(country);
                 }
)
```

Objects

Object Creation - Object Literal Notation

```
var objName = {
    name1: value1,
    name2: value2,
        // ...
    nameN: valueN
    };
```

Objects

Object Creation - Object Literal Notation

- Access using either of:

objName.name1

objName["name1"]

Objects

Object Creation – Constructed Form

```
// first create an empty object
var objName = new Object();
// then define properties for this object
objName.name1 = value1;
objName.name2 = value2;
```

Web Systems and Technologies

Functions (1)

Functions

Function Declarations vs. Function Expressions

- **Functions** are the building block for modular code in JavaScript

```
function subtotal(price,quantity) {
        return price * quantity;
}
```

- The above is formally called a **function declaration**, called or invoked by using the () operator

```
var result = subtotal(10,2);
```

Functions

Function Declarations vs. Function Expressions

```
// defines a function using a function expression
var sub = function subtotal(price,quantity) {
return price * quantity;
};
// invokes the function
var result = sub(10,2);
```

It is conventional to leave out the function name in function expressions

Functions

- Anonymous Function Expressions

```
// defines a function using an anonymous function expression

var calculateSubtotal = function (price,quantity) {

return price * quantity;

};

// invokes the function

var result = calculateSubtotal(10,2);
```

Functions

Arrow Functions

- Have a shorter syntax
- Can convert normal functions to arrow
- Useful for anonymous and call functions

```
//normal function
function square(n) {
  return n*n;
}

//arrow function
sq = (n) => n*n

//Another example
sum = (n,m) => n+m
```

WEEK 6

TOPIC 46 TO 54

Web Systems and Technologies

- Functions-Nested
- Functions-Callback
- Functions-Objects
- Functions-Scope
- Exception handling using Try and Catch
- JavaScript built in Objects (Data, Math, Navigator, String, and etc.)
- Object prototypes: Prototypes for Extending Objects
- Introduction to the Document Object Model (DOM)
- Accessing DOM Clements

Functions (2)

Nested Functions

nested function mcq

```
function calculateTotal(price,quantity) {
        var subtotal = price * quantity;
        return subtotal + calculateTax(subtotal);
        // this function is nested
        function calculateTax(subtotal) {
                var taxRate = 0.05;
                var tax = subtotal * taxRate;
                return tax;
        }
}
```

Hoisting in JavaScript

Function declaration is hoisted to the beginning of its scope

```
function calculateTotal(price,quantity) {
    var subtotal = price * quantity;
    return subtotal + calculateTax(subtotal);

    function calculateTax(subtotal) {
        var taxRate = 0.05;
        var tax = subtotal * taxRate;
        return tax;
    }
}
```

Variable declaration is hoisted to the beginning of its scope

BUT

Variable assignment is not hoisted

```
function calculateTotal(price,quantity) {
    var subtotal = price * quantity;
    return subtotal + calculateTax(subtotal);

    var calculateTax = function (subtotal) {
        var taxRate = 0.05;
        var tax = subtotal * taxRate;
        return tax;
    };
}
```

THUS

The value of the `calculateTax` variable here is undefined

Callback Functions

where we call the call back function.

Callback Functions

```
var calculateTotal = function (price, quantity, tax) {
    var subtotal = price * quantity;
    return subtotal + tax(subtotal);
};

var calcTax = function (subtotal) {
    var taxRate = 0.05;
    var tax = subtotal * taxRate;
    return tax;
};

var temp = calculateTotal(50,2,calcTax);
```

2 The local parameter variable tax is a reference to the calcTax() function

1 Passing the calcTax() function object as a parameter

We can say that calcTax variable here is a callback function

Callback Functions

Passing an anonymous function definition as a callback function parameter

```
var temp = calculateTotal( 50, 2,
                           function (subtotal) {
                              var taxRate = 0.05;
                              var tax = subtotal * taxRate;
                              return tax;
                           }
                         );
```

Functions - Objects

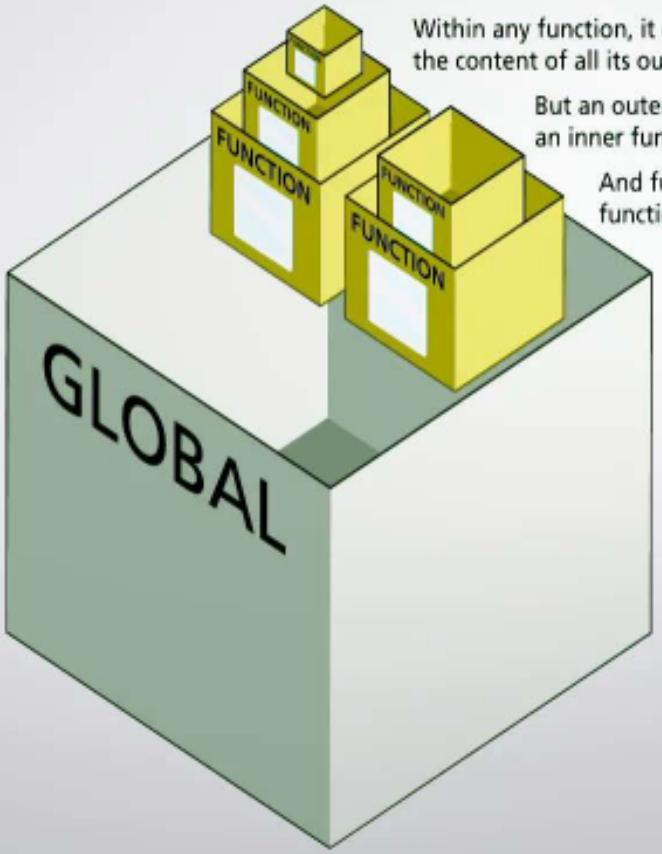
Objects and Functions Together

```
var order = {
    salesDate : "May 5, 2017",
    product : {
        type: "laptop",
        price: 500.00,
        output: function () {
            return this.type + ' $' + this.price;
        }
    },
    customer : {
        name: "Sue Smith",
        address: "123 Somewhere St",
        output: function () {
            return this.name + ', ' + this.address;
        }
    },
    output: function () {
            return 'Date' + this.salesDate;
    }
};
```

Functions (5)

Scope in JavaScript

Each function is like a box
with a one-way window



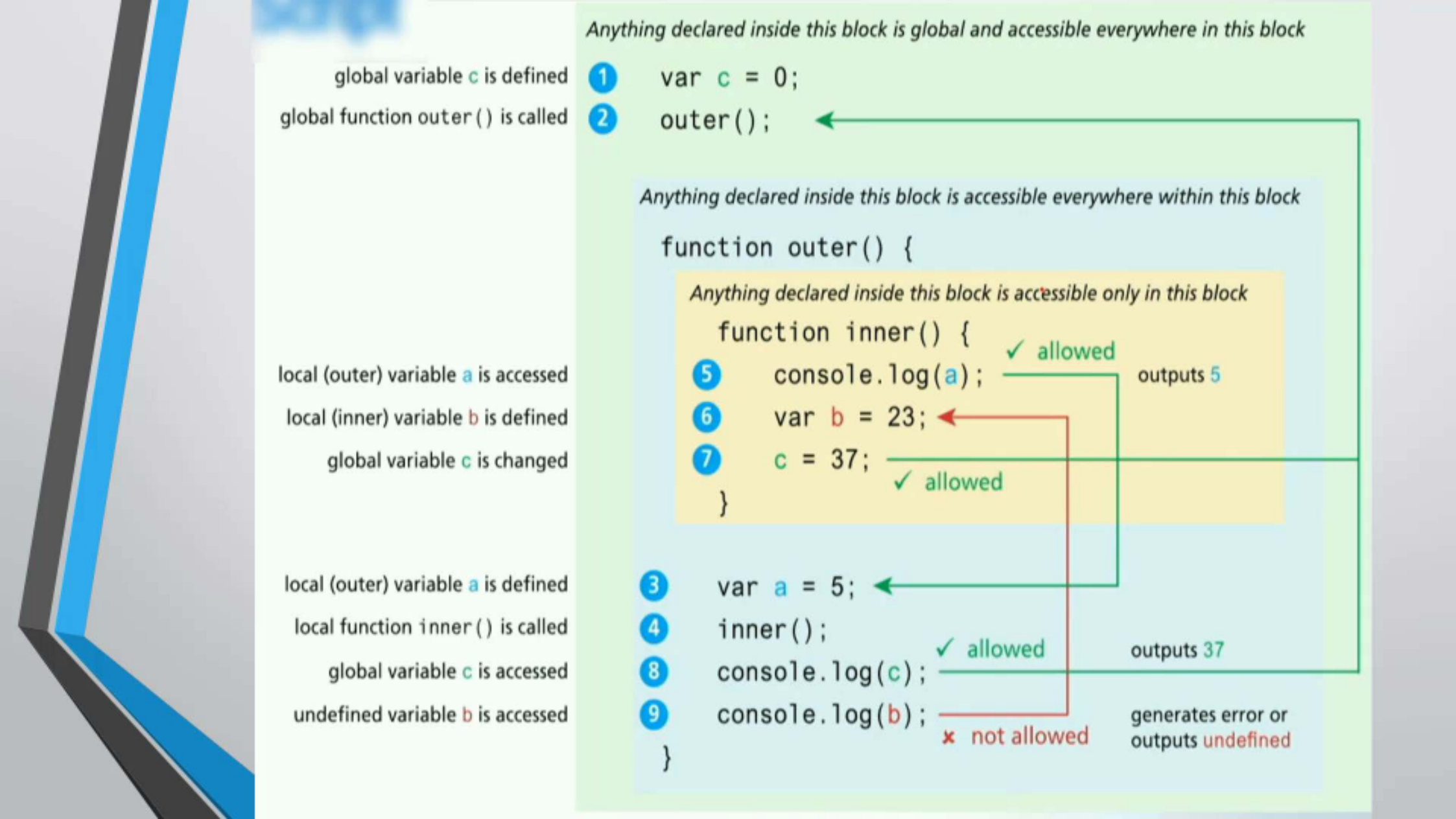
Within any function, it can see out at the content of all its outer boxes

But an outer function can't look into an inner function

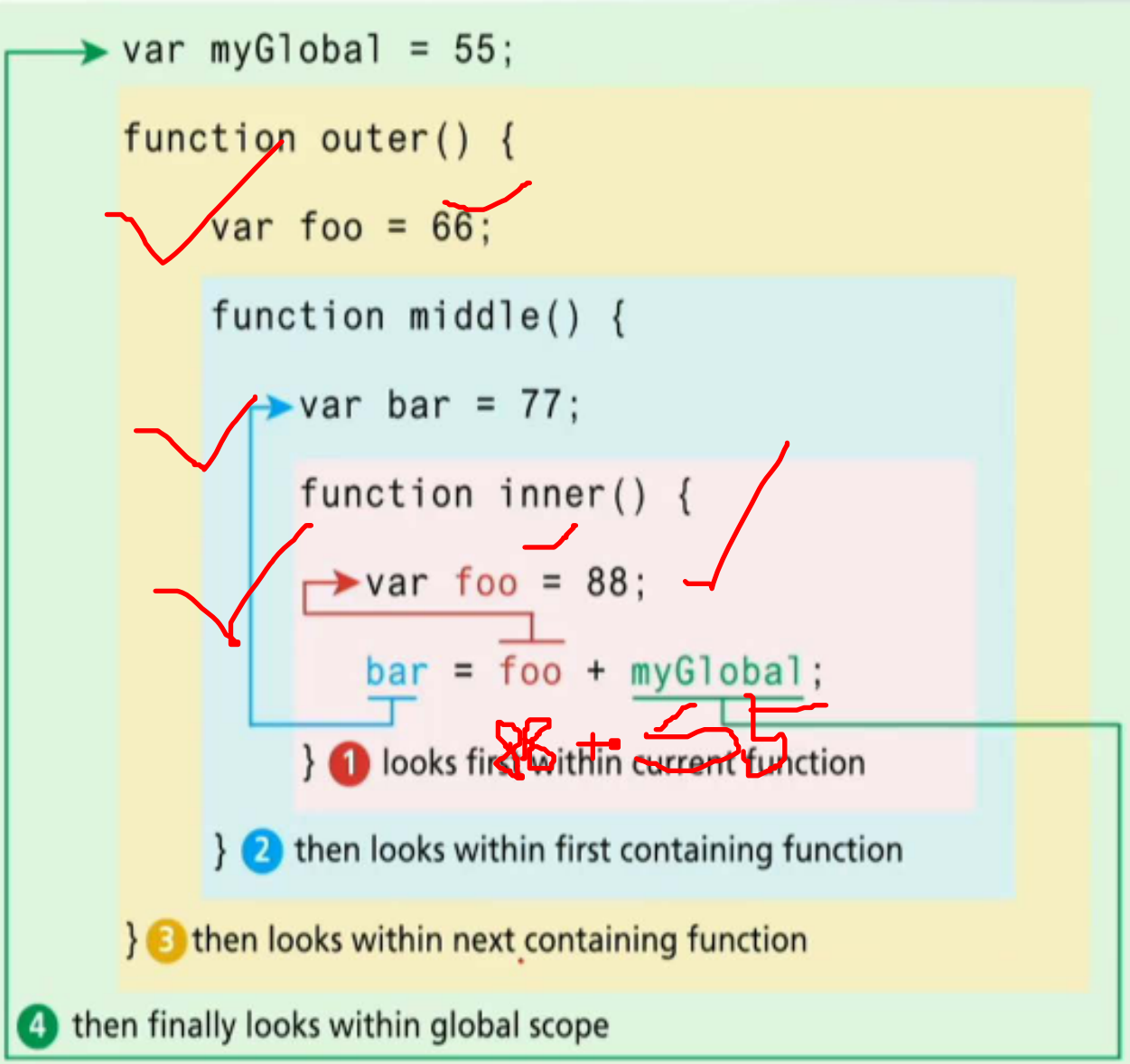
And functions can't see into other functions at the same level

All functions can see anything within global scope

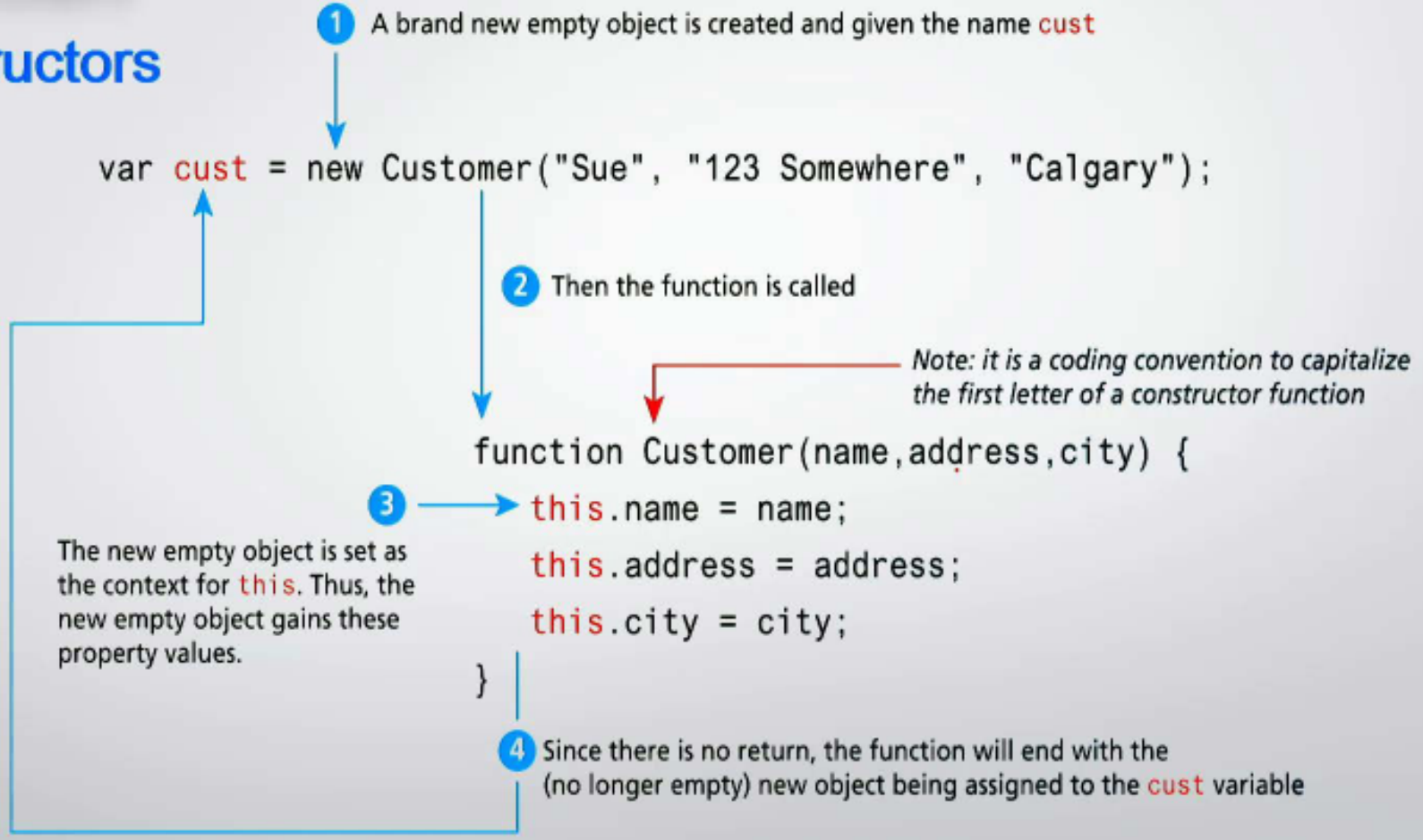
Scope ends at global ...
functions can't see outside of the global box


Anything declared inside this block is global and accessible everywhere in this block
global variable c is defined
1 var c = 0;
global function outer() is called
2 outer();
Anything declared inside this block is accessible everywhere within this block
function outer() {
Anything declared inside this block is accessible only in this block
function inner() {
local (outer) variable a is accessed
5 console.log(a);
✓ allowed
outputs 5
local (inner) variable b is defined
6 var b = 23;
global variable c is changed
7 c = 37;
✓ allowed
}
local (outer) variable a is defined
3 var a = 5;
local function inner() is called
4 inner();
global variable c is accessed
8 console.log(c);
✓ allowed
outputs 37
undefined variable b is accessed
9 console.log(b);
✘ not allowed
generates error or
outputs undefined
}

Remember that scope is determined at design-time



Function Constructors



Exception Handling

Introduction

- Errors occur at runtime are called exceptions
- Might result in unexpected behavior if unhandled
- Handling exceptions avoid unexpected termination

Examples

- TypeError

```
var x
console.log(x.a)
```

- RangeError

```
var A = new Array(-5)
```

- ReferenceError

```
function f() {
  var y = 2;
}
console.log(y)
```

Handling Exceptions

try catch block start with which keyword.

- Exceptions are handled using try/catch
- Statements expected to result in an exception are put in **try** block
- Exception handling is done in **catch** block

```
try {
    var A = new Array(-5);
} catch (e) {
    console.log(e);
}
```

Throwing Exceptions

- Exceptions can be generated (thrown) by the programmer using **throw**

```
try {
    …
    if (age < 0 || age > 150) {
      throw("InvalidAge: Valid age
is between 0 and 150");
    }
} catch (e) {
    console.log(e);
}
```

Catching Specific Exceptions

- Use instanceof to catch specific exceptions

```
try {
    var A = new Array(-5);

} catch (e) {
    if(e instanceof RangeError) {
        ...
    }
}
```

The *finally* Block

- The finally block is executed irrespective of exception being thrown or not
- Useful for releasing resources

```
try {
    //connect to a database
} catch (e) {
    //an error occurred
    //rollback
} finally {
    //disconnect from the database
}
```

Web Systems and Technologies

Built-in Objects

Built-in Objects

Working with Maths

- Math object provides many functions
- Called through Math object, e.g., Math.abs(-5) returns 5
  - Trigonometry functions: sin(), cos(), …
  - Rounding: floor(), ceil(), round(), …
  - Statistical: min(), max(), …
  - Math.random() returns a value between [0,1).

```
function rand_int_between(min,max) {
  return Math.floor(Math.random()*
         (max-min+1)+min);
}
```

Built-in Objects

Working with Dates

short question from builtin date

- Date object provides functions to work with dates and times
  - Date.now(): milliseconds passed since Jan 1, 1970
  - Date.parse(): converts a string to a date, e.g., Date.parse('Sep 1, 2021')
  - toDateString(): converts a date object to a string in human readable form

```
var today = new Date();
console.log(today.toDateString())
```

Built-in Objects

Working with Patterns

- RegExp object is used for matching text with patterns
  - test(): tests if a string follows a pattern

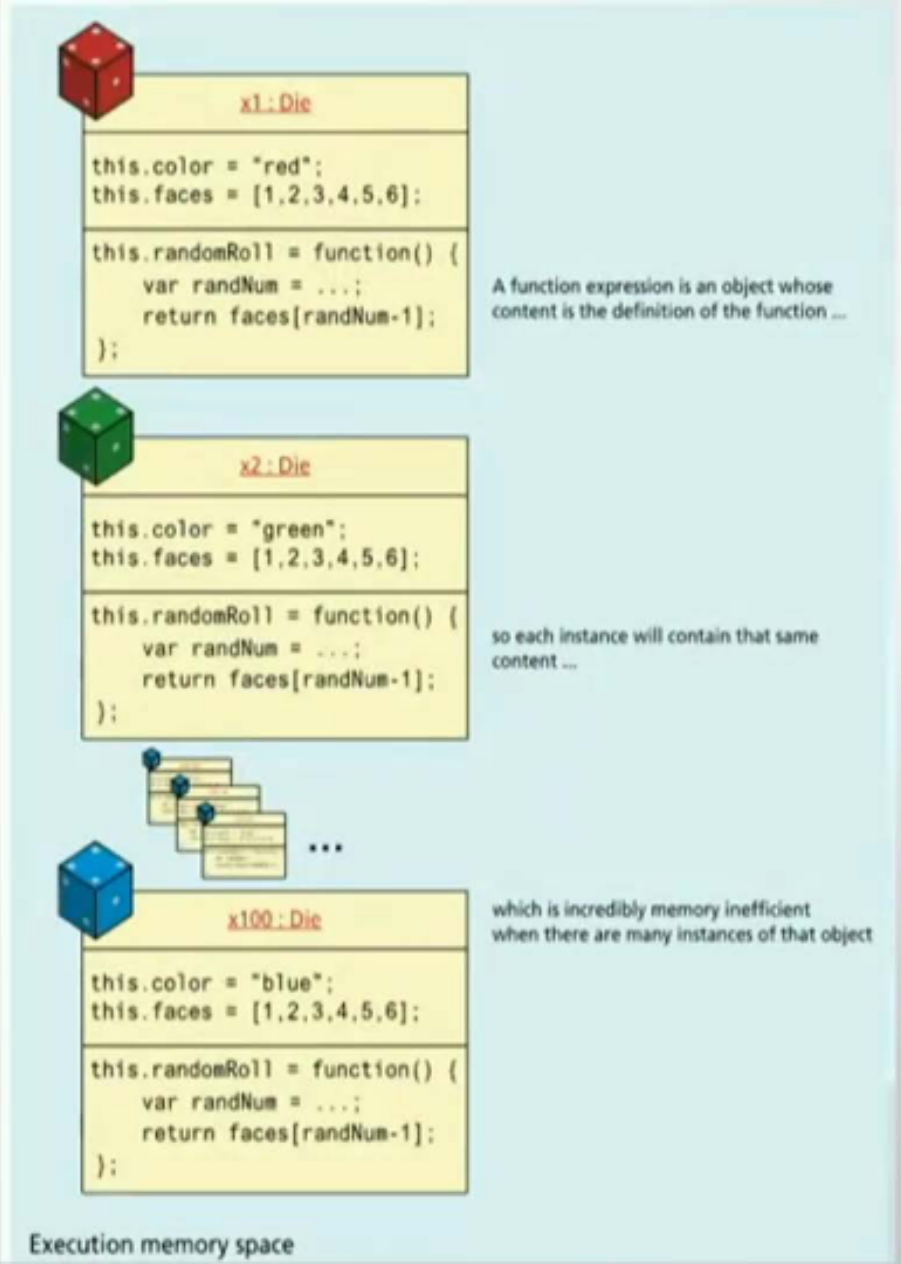
```
//Example
var pattern = /\d+/
var pattern = new RegExp('\\d+')
var num = 123;
var str = 'abc';
console.log(pattern.test(num))
console.log(pattern.test(str))
```

Object Prototypes

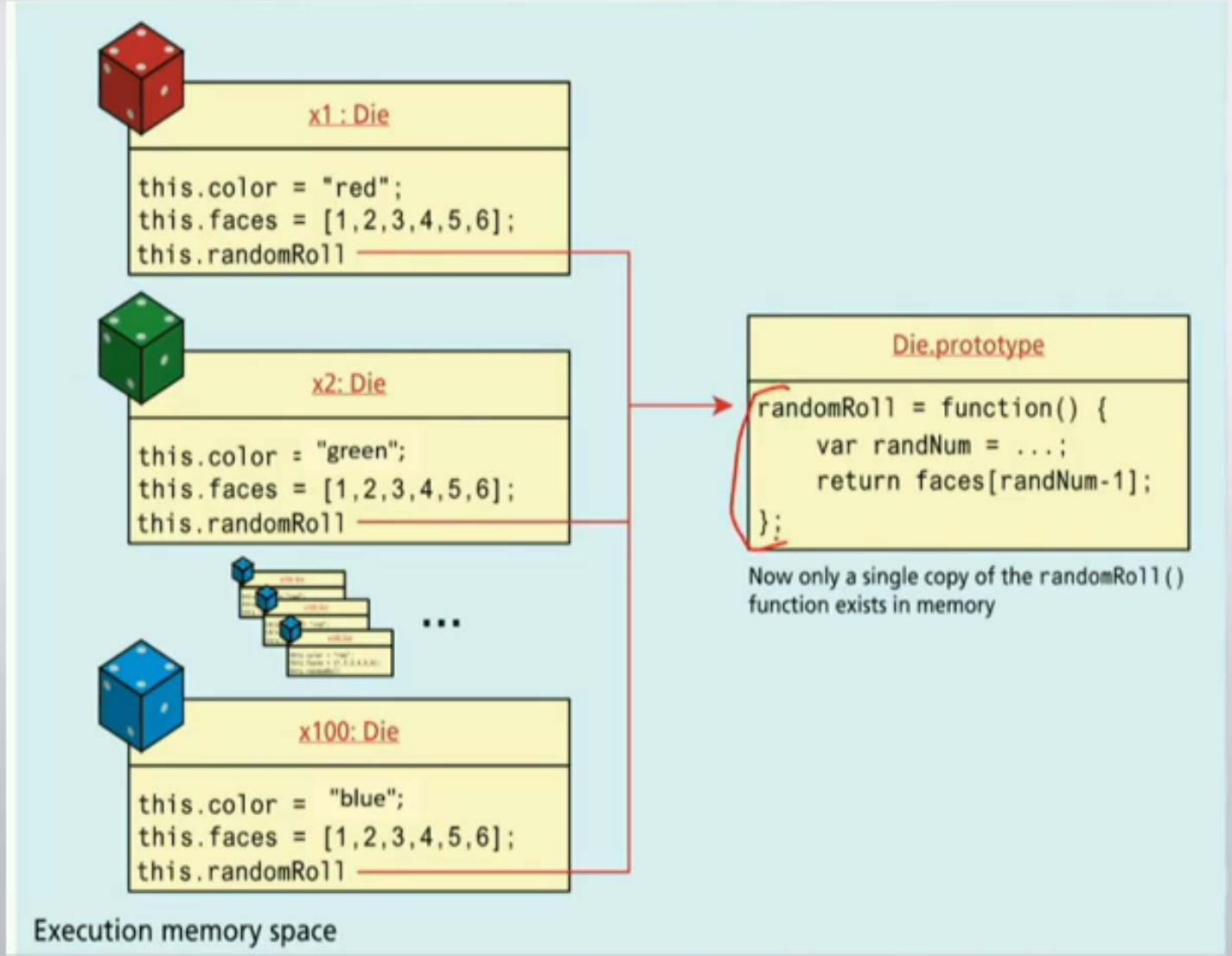
Introduction

- **Prototypes** are an essential syntax mechanism in JavaScript, and are used to make JavaScript behave more like an object-oriented language

Methds Get Duplicated



Using Prototype reduces Duplication at Runtime



Using Prototypes to Extend Other Objects

```
String.prototype.countChars = function (c) {
        var count=0;
        for (var i=0;i<this.length;i++) {
                if (this.charAt(i) == c)
                        count++;
                }
        return count;
}
var msg = "Hello World";
console.log(msg + "has" + msg.countChars("l") + "
letter l's");
```

The Document Object Model (DOM)

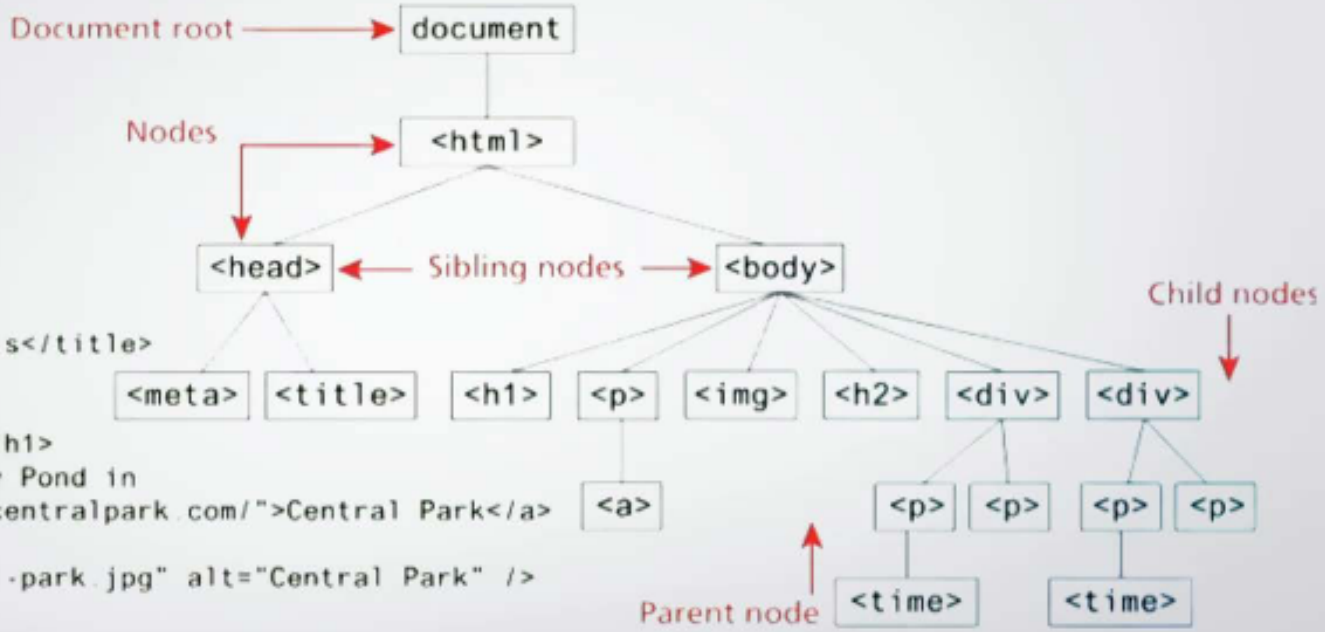
why we use dom.

Introduction

- Any HTML document can be represented in form of a tree called Document Object Model
- Available as a data structure in JavaScript
- Helps in accessing and manipulating HTML elements through JavaScript

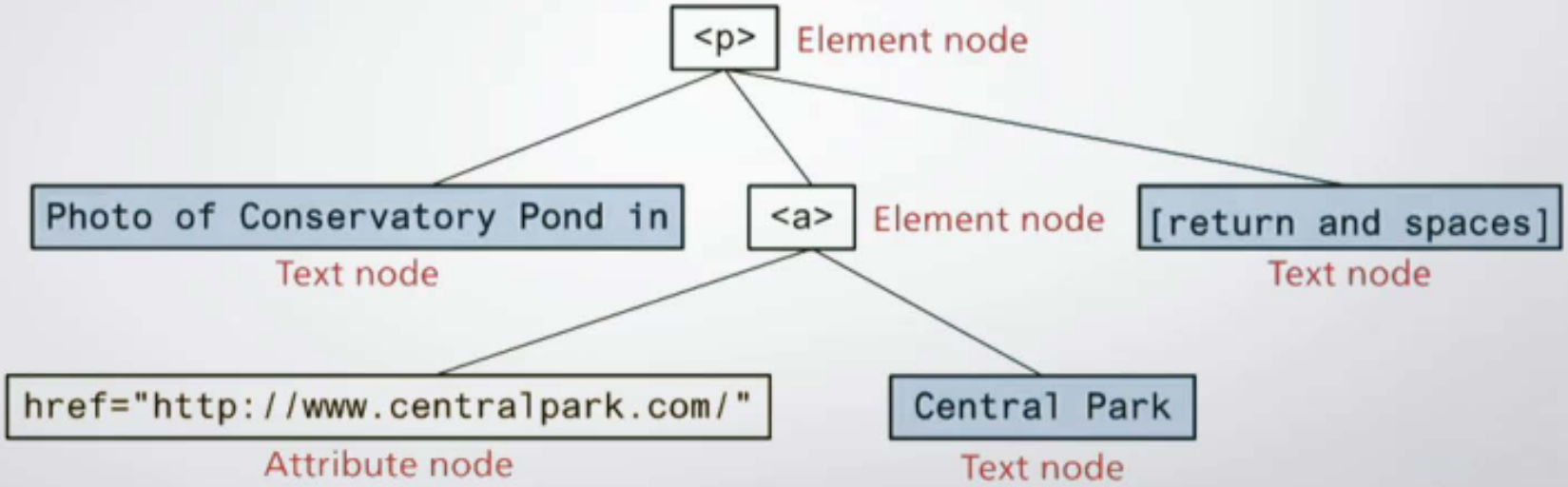
Overview

```
<html>
<head>
   <meta charset="utf-8">
   <title>Share Your Travels</title>
</head>
<body>
   <h1>Share Your Travels</h1>
   <p>Photo of Conservatory Pond in
      <a href="http://www.centralpark.com/">Central Park</a>
   </p>
   <img src="images/central-park.jpg" alt="Central Park" />
   <h2>Reviews</h2>
   <div id="latestComment">
      <p>By Ricardo on <time>2016-05-23</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>
   <div>
      <p>By Susan on <time>2016-11-18</time></p>
      <p>I love Central Park.</p>
   </div>
</body>
</html>
```



Nodes

```
<p>Photo of Conservatory Pond in
   <a href="http://www.centralpark.com/">Central Park</a>
</p>
```



DOM Related Data Types

- Document
  - The root document
- Node
  - Every object in the HTML document is a node (elements, attributes, text, etc.)
- NodeList
  - Array of elements
- Element
  - Refers to an HTML element or a node
- Attribute
  - Refers to an attribute

Accessing DOM Elements

Document Object

- The DOM **document** object is the root JavaScript object representing the entire HTML document

```
// retrieve the URL of the current page

var a = document.URL;

// retrieve the page encoding, for
example ISO-8859-1

var b = document.inputEncoding;
```

Selection Methods

- Classic
  - getElementById()
  - getElementsByTagName()
  - getElementsByClassName()
- Newer
  - querySelector() and
  - querySelectorAll()

Selection Methods

```
var node = document.getElementById("latest");
```

```
<body>
    <h1>Reviews</h1>
    <div id="latest">
        <p>By Ricardo on <time>2016-05-23</time></p>
        <p class="comment">Easy on the HDR buddy.</p>
    </div>
    <hr/>
    <div>
        <p>By Susan on <time>2016-11-18</time></p>
        <p class="comment">I love Central Park.</p>
    </div>
    <hr/>
</body>
```

```
var list2 = document.getElementByClassName("comment");
```

```
var list1 = document.getElementsByTagName("div");
```

Query Selector

querySelectorAll("nav ul a:link")

querySelectorAll("#main div time")

querySelector("#main>time")

querySelector("footer")

```
<body>
  <nav>
    <ul>
      <li><a href="#">Canada</a></li>
      <li><a href="#">Germany</a></li>
      <li><a href="#">United States</a></li>
    </ul>
  </nav>
  <div id="main">
    Comments as of
    <time>November 15, 2012</time>
    <div>
      <p>By Ricardo on <time>September 15, 2012</time></p>
      <p>Easy on the HDR buddy.</p>
    </div>

    <div>
      <p>By Susan on <time>October 1, 2012</time></p>
      <p>I love Central Park.</p>
    </div>
  </div>
  <footer>
    <ul>
      <li><a href="#">Home</a> | </li>
      <li><a href="#">Browse</a> | </li>
    </ul>
  </footer>
</body>
```

Element Node Object

- Element Node object represents an HTML element in the hierarchy, contained between the opening <> and closing </> tags for this element. Every node has
  - classList
  - className
  - id
  - innerHTML
  - style
  - tagName

Element Node Object (2)

- Some properties are not common in all the elements
  - href
  - name
  - src
  - value