Announcements

- Assignment 1 Due Soon (9/13)
  - Rubric present on Class Website
- GIT Demo Notes on class website
  - Please practice it
- Project Team Mentors will be assigned this week

From last class: GIT Demo

EGit

- GIT Plugin available for Eclipse
- The plugin can be downloaded at www.eclipse.github.com and can be installed in Eclipse

From last class: GIT Recap

LOCAL REPOSITORIES

CREATE

mkdir ProjectName
cd ProjectName
git init

MODIFY

git add foo.txt
git commit –m “message”

INSPECT

git log
git status
git diff
git diff HEAD
git show

REMOTE REPOSITORIES

COPY REPOSITORY

git clone <repo url>

- Creates a completely local copy of repository
- links it to remote repository (origin)

RECEIVE CHANGES

git pull

SEND CHANGES

git push

From last class: GitHub

- GIT hosting website. Get an account and create your remote repositories
- GitHub repository for your projects
- Provides easy-to-use FREE desktop clients for Mac and Windows (https://desktop.github.com)

- **GitHub Pages**:
  - One click to enable for your GitHub repo.
  - Hosted directly from your GitHub repository.
  - Just edit, push, and your changes are live.
  - This course's website is a GitHub Page.
  - Display both your projects as GitHub Pages at the end of this course. Great addition to your resume.

- ALWAYS SET YOUR GITHUB REPOSITORY TO BE PRIVATE, UNLESS YOU ARE ABSOLUTELY SURE YOU WANT IT PUBLIC !!!

CS3300 Introduction to Software Engineering

Lecture 05: Tools of the Trade #3

HTML, CSS, JS

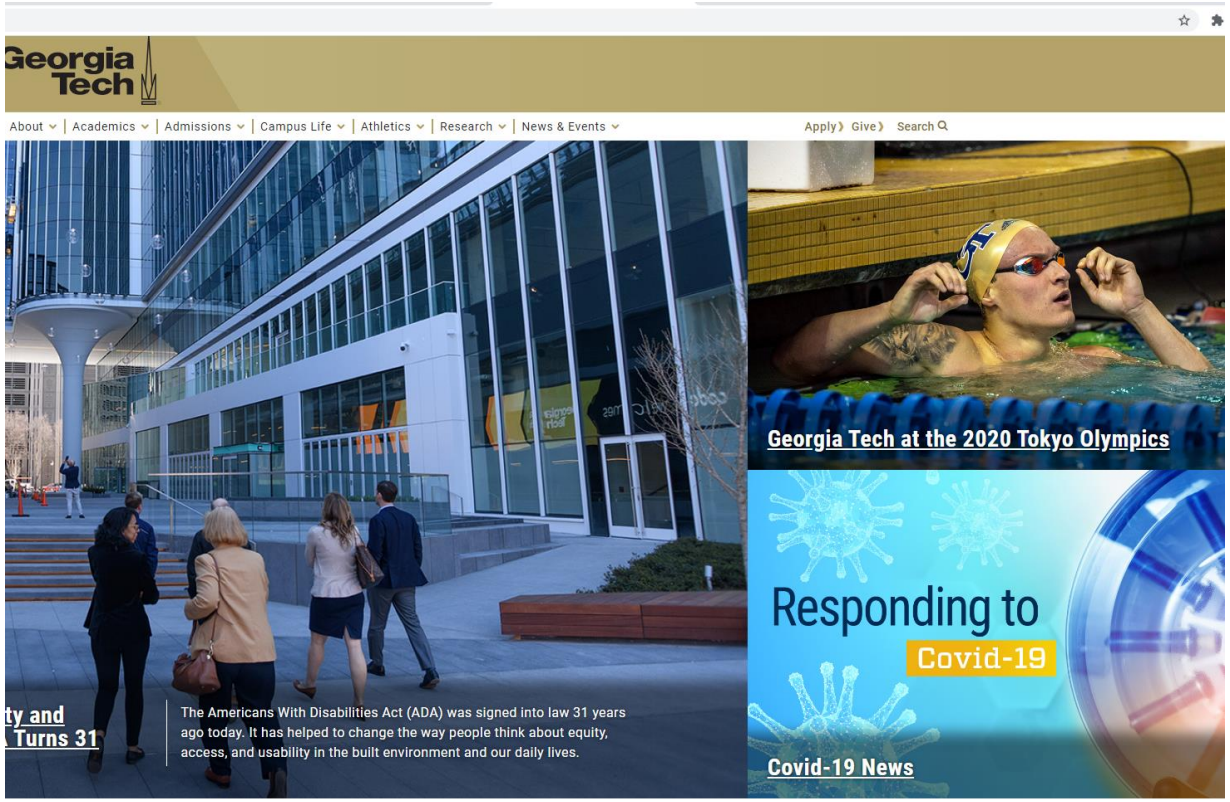
Nimisha Roy ▸ nroy9@gatech.edu

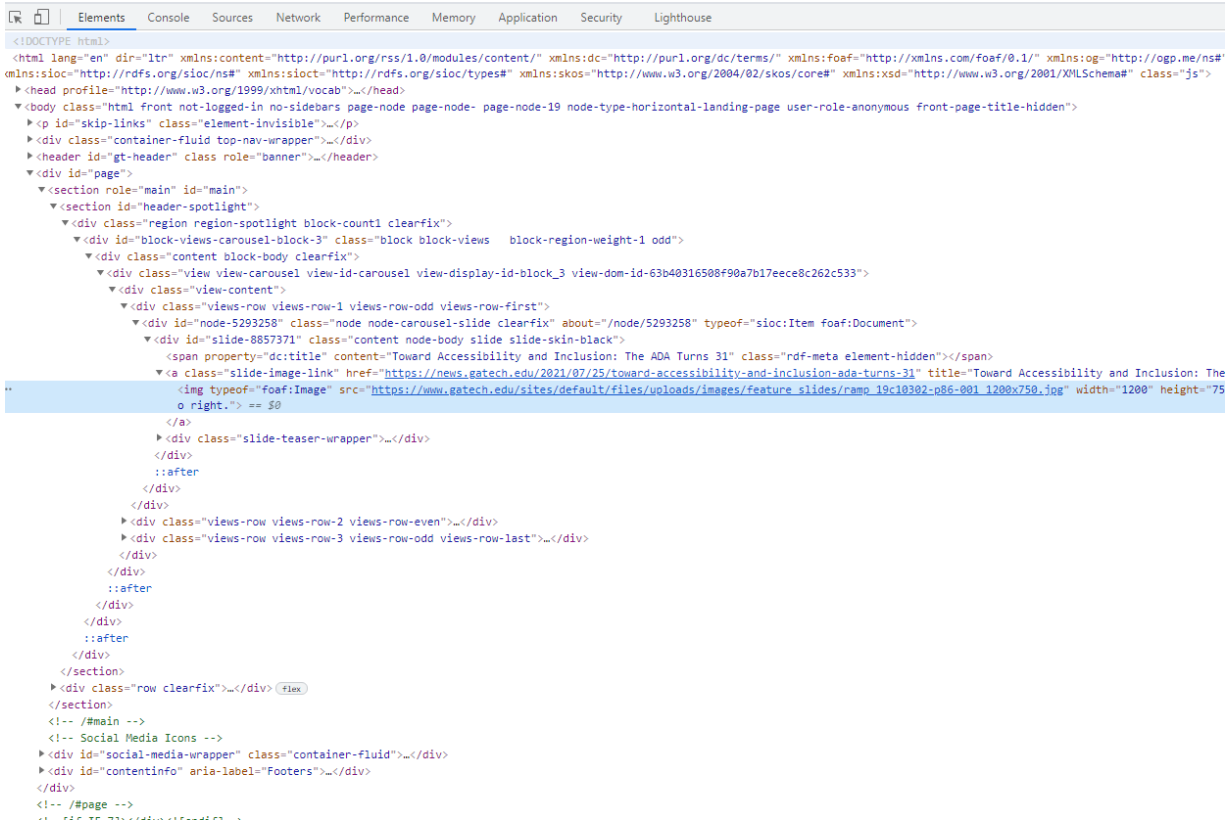
Slides adapted from Mahdi Roozbahani

Contents

- What is a Webpage made of?
- HTML
  - Common Tags (Headings, Paragraphs, Line Breaks, Links, Images, Divs)
- CSS
  - Structure
  - Positioning Elements
- Javascript
  - Statements, Variables, Operators, Datatypes, Functions, Objects, Events
- Visual Studio Code
- Demo

What is a webpage made of?





What is a webpage made of?

HTML

Hypertext Markup Language

Creates the Structure

- Describes the structure of a Web page
- Consists of a series of elements
- Tells the browser how to display the content

CSS

Cascading Style Sheet

Stylizes the webpage

- Applies style to the web page elements
- Targets various screen sizes to make the web page responsive
- Primarily handles the “look and feel” of a webpage

JS

JavaScript

Increases Interactivity

- Adds interactive component to webpage
- Handles complex functions and features
- Enhances functionality

HTML

An HTML element is defined by a start tag, some content, and an end tag

<tagname>Content goes here...</tagname>

The HTML **element** is everything from the start tag to the end tag:

<h1>My First Heading</h1>
My first paragraph.

HTML

A simple HTML document

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

The <!DOCTYPE html> declaration defines that this document is an HTML5 document
The <html> element is the root element of an HTML page
The <head> element contains meta information about the HTML page
The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
The <body> element defines the document's body; container for all the visible contents, like headings, paragraphs, images, hyperlinks, tables, lists.
The <h1> element defines a large heading
The <p> element defines a paragraph

HTML

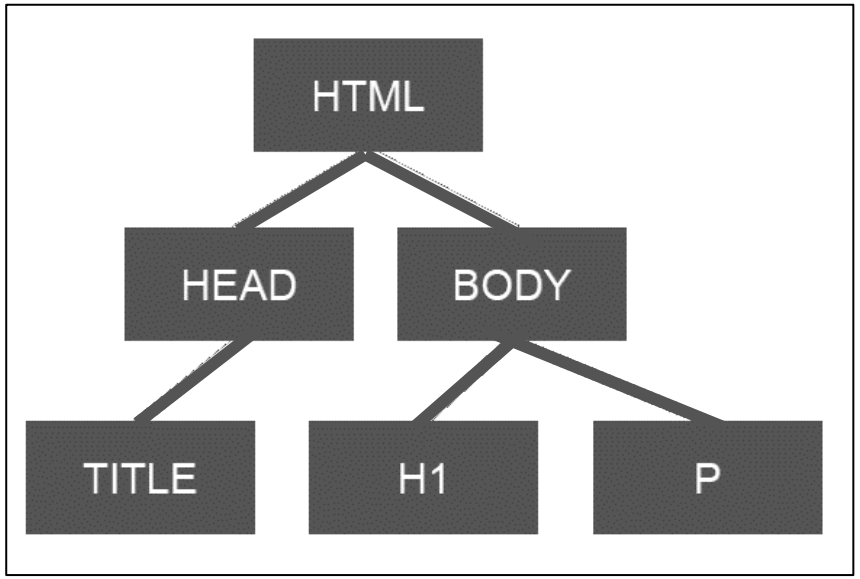
A simple HTML document

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

My First Heading

My first paragraph.

Tags are translated into a tree-like hierarchy



HTML: Common Tags - Headings

```
<h1>I'm a heading!</h1>
<h2>I'm a subheading!</h2>
<h3>I'm a subsubheading?</h3>
```

I'm a heading!

I'm a subheading!

I'm a subsubheading?

- Declared using h1, h2…..h6 tags
- h1 is the main heading
- Used by search engines to index content

HTML: Common Tags - Paragraphs

```
<p>First paragraph</p>
<p>Second paragraph</p>
```

First paragraph

Second paragraph

- Necessary for proper spacing of text
- HTML ignores whitespace

HTML: Common Tags – Line breaks

```
<p>
    First line<br>
    Second line
</p>
```

First line
Second line

- Creates a new line without adding space
- Interpreted differently by search engines/screen readers
-
 tag has no content. Empty elements do not have an end tag

HTML: Common Tags – Links

```
<a href="page2.html">Click here!</a>
```

Click here!

- Links to another HTML file on the internet or your computer
- Created using anchor tags
- Destination page can be specified as a relative or absolute path

HTML: Common Tags – Images

```
<img src="buzzbee.jpg" alt="GaTech mascot">
```

- Can point to any image the browser supports (.jpg, . png , .svg etc.)
- Alt text is required
- Height/Width can be controlled with CSS

HTML: Common Tags – Divs

```
<div>First div</div>
<div>Second div</div>
```

First paragraph

Second paragraph

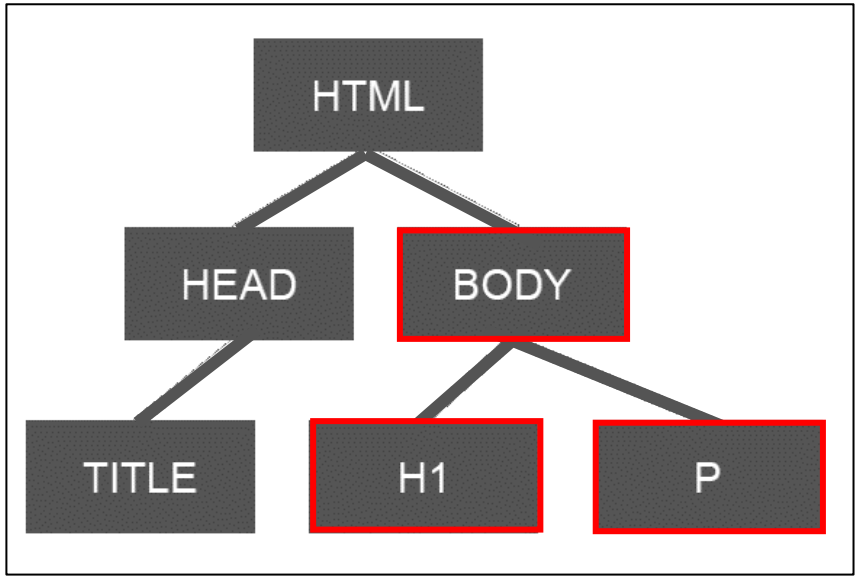
- Can be used to split page into defined rectangular regions
- Can be positioned and styled independently using CSS

CSS

- In HTML, CSS code is inserted between <style> and </style> tags, or as inline styles. <element style = “property : value;”>
- Code should be placed in the <head> section of an HTML page
- Scripts can also be placed in external files with extension *.css*
- External scripts can be referenced with a full URL or with a path relative to the current web page.

CSS

- CSS applies styles to HTML elements
- Cascades down in the hierarchy



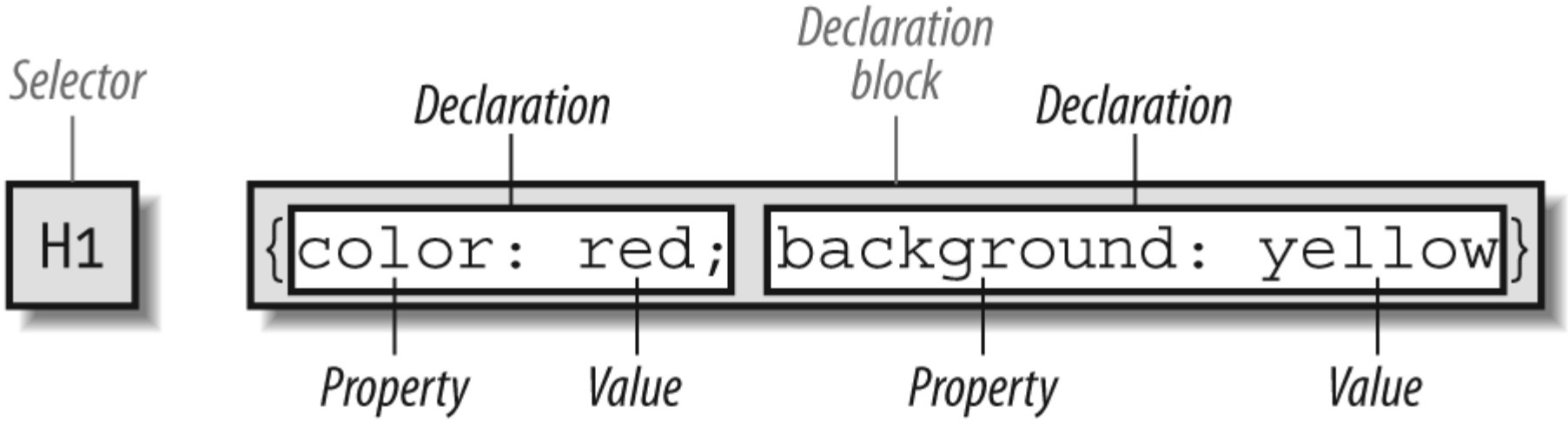

```
body {
    font-family: Arial, serif;
    font-weight: bold;
    font-size: 12px;
    background-color: red;
}
```

CSS

Referenced from an HTML using the LINK tag

```
<html>
    <head>
        <title>Hello World!</title>
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        <h1>I'm a heading!</h1>
        <p>I'm some content!</p>
    </body>
</html>
```

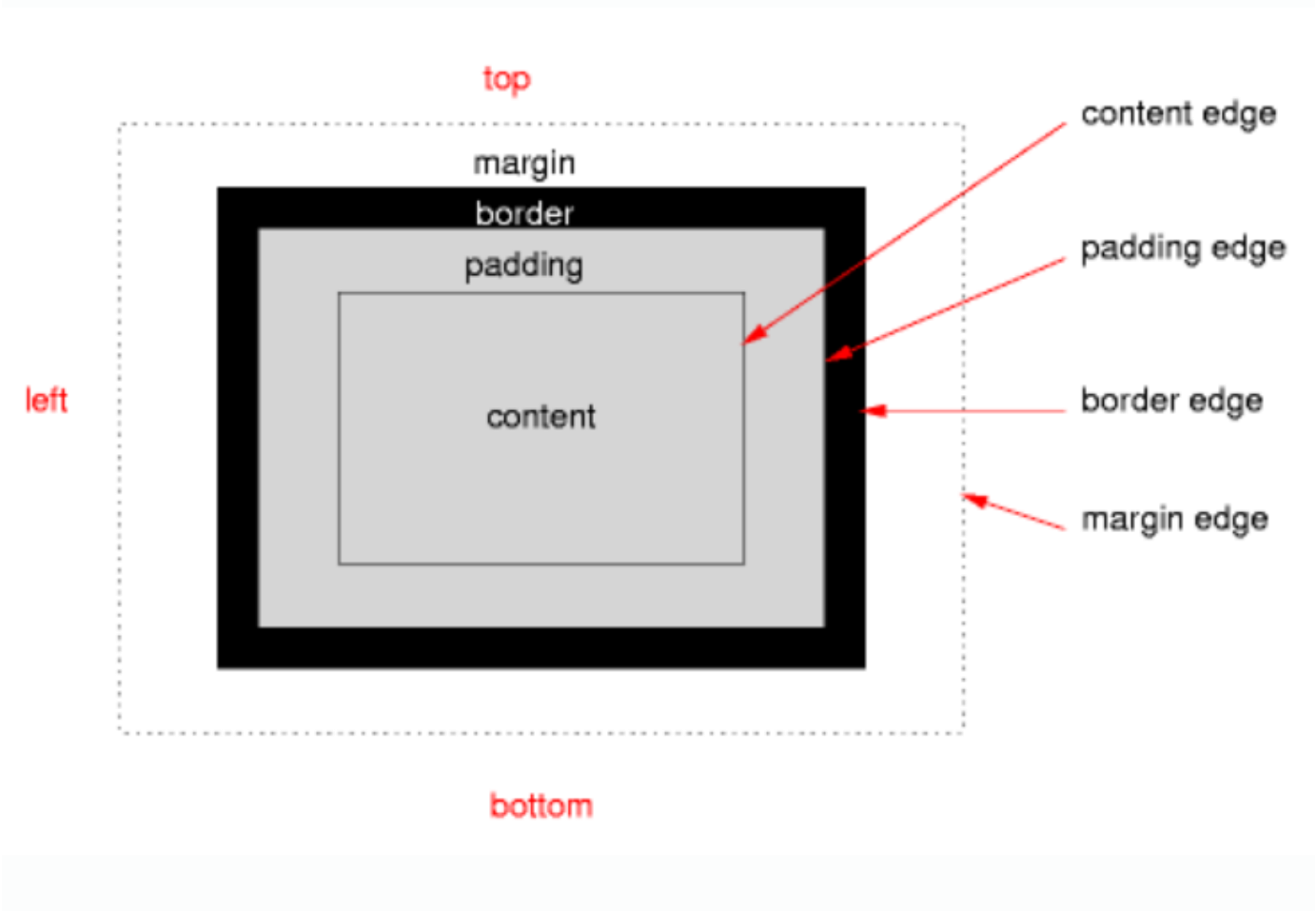
CSS: Structure



- CSS **selector** defines, identifies and targets the HTML elements to whom its set of style declarations apply.
- **Declaration block** starts and ends in curly braces
- A CSS **declaration** is a property-value pair, with both entities separated by a colon punctuation mark and ended by a semicolon.
- A CSS **property** is responsible for the exact style that you want to apply to the targeted element. (font-size, margin-top, color, padding-left, background-color, width, display, font-family, height etc.)

- **The Universal Selector** `* { }`
- **Tag or HTML Selectors** `h1 { }  p { }  body { }`
- **Class Selectors** `.imageframes { }`
- **ID Selectors** `#main-navigation { }`
- **Selecting in context** `h1 strong { }`
- **Attribute Selectors** `li img[title] { }`

CSS: Positioning Elements



- Margin : Amount of space around element
- Padding: Space between content and element border
- width/height: Uses units of em , px , or %

JavaScript

- In HTML, JavaScript code is inserted between <script> and </script> tags.

- Scripts can be placed in the <body>, or in the <head> section of an HTML page, or in both.

- Scripts can also be placed in external files with extension *.js*
  - Separated HTML and JS codes
  - easier to read and maintain
  - Cached JS files can speed up page loads

- External scripts can be referenced with a full URL or with a path relative to the current web page.

JavaScript –What can it do?

- Can change HTML content
  - Use one of the methods getElementById() to find an HTML element with a particular ID and change the element content
- Can change HTML Attribute Values
- Can change styles (CSS)
- Can hide/show HTML Elements
- Can display data using “innerHTML”, “document.write()”, “window.alert()”, “console.log()”

JavaScript – Statements

JavaScript statements are composed of: Values, Operators, Expressions, Keywords, and Comments.

Keywords: (reserved word part of the vocabulary of the language)

Keyword	Description
var	Declares a variable
let	Declares a block variable
const	Declares a block constant
if	Marks a block of statements to be executed on a condition
switch	Marks a block of statements to be executed in different cases
for	Marks a block of statements to be executed in a loop
function	Declares a function
return	Exits a function
try	Implements error handling to a block of statements

JavaScript – Variables

Can be described using **var**, **let** or **const**

- **var** x=5;
- **let** cannot redeclare a variable. **let** provides Block Scope

```
let x = "John Doe";

let x = 0;

// SyntaxError: 'x' has already been declared
```

```
var x = 10;
// Here x is 10

{
var x = 2;
// Here x is 2
}

// Here x is 2
```

```
let x = 10;
// Here x is 10

{
let x = 2;
// Here x is 2
}

// Here x is 10
```

- Variables defined with **const** cannot be redeclared, reassigned and have block scope

```
const PI = 3.141592653589793;
PI = 3.14;      // This will give an error
PI = PI + 10;   // This will also give an error
```

```
// You can create a constant array:
const cars = ["Saab", "Volvo", "BMW"];

// You can change an element:
cars[0] = "Toyota";

// You can add an element:
cars.push("Audi");
```

```
const cars = ["Saab", "Volvo", "BMW"];

cars = ["Toyota", "Volvo", "Audi"];    // ERROR
```

JavaScript – Operators

Arithmetic operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

Assignment Operators

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

JavaScript – Operators

Comparison operators

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

Bitwise Operators

Operator	Description	Example	Same as	Result	Decimal
&	AND	5 & 1	0101 & 0001	0001	1
|	OR	5 | 1	0101 | 0001	0101	5
~	NOT	~ 5	~0101	1010	10
^	XOR	5 ^ 1	0101 ^ 0001	0100	4
<<	Zero fill left shift	5 << 1	0101 << 1	1010	10
>>	Signed right shift	5 >> 1	0101 >> 1	0010	2
>>>	Zero fill right shift	5 >>> 1	0101 >>> 1	0010	2

JavaScript – Data Types

- Primitive Data Types
  - Numbers
  - Strings
  - Boolean (True, False)
  - Null
  - Undefined
  - Symbol

- Composite Data Types
  - Arrays
  - Objects

- JavaScript is *untyped*; It does not have explicit data types
  - For instance, there is no way to specify that a particular variable represents an integer, string, or real number
  - The same variable can have different data types in different contexts
  - Can use **typeof** operator to find type of JS variable

- JS has implicit datatype. Type coercion priority order:

1. String; 2. Number; 3. Boolean
   - 2 + “9” = “29”
   - false * 4 = 0

JavaScript – Functions

- Block of code designed to perform a particular task
- Invoked by (). Optional return
- Syntax:

```
function name(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

- Can be used as variable values

```
let text = "The temperature is " + toCelsius(77) + " Celsius";
```

- Variables declared within function become local to the function- Functional scope

```
// code here can NOT use carName

function myFunction() {
  let carName = "Volvo";
  // code here CAN use carName
}

// code here can NOT use carName
```

JavaScript – Objects

- Datatype written as **name:value** pairs (name and value separated by a colon).
- It is a common practice to declare objects with the **const** keyword. **var** can be used too.
- Spaces and line breaks are not important. An object definition can span multiple lines.
- In a function definition, **this** refers to the "owner" of the function.

```
const person = {
  firstName: "John",
  lastName : "Doe",
  id       : 5566,
  fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};
```

- A method of an object can be accessed by *objectName.methodName*()
- Avoid String, Number, and Boolean objects. They complicate your code and slow down execution speed.
- **Date** and **Math** are examples of built-in objects.

JavaScript – Events

- HTML events are **"things"** that happen to HTML elements.
- When JavaScript is used in HTML pages, JavaScript can **"react"** on these events.

<element event='some JavaScript'>

- Event handlers can be used to handle and verify user input, user actions, and browser actions:
  - Things that should be done every time a page loads; Things that should be done when the page is closed; Action that should be performed when a user clicks a button; Content that should be verified when a user inputs data

Event	Description
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

Visual Studio Code

- Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control.
- Built-in support for JavaScript
- Supported on macOS, Windows and Linux
- Provides great syntax highlighting; auto-complete with IntelliSense based on variable types, function definitions, and imported modules.
- Free
- Integrated with GitHub, Embedded GIT

DEMO TIME!!