



# Module 4 Day 1

## Client-side Scripting with JavaScript

# What makes an application?

- Program Data

- ✓ Variables & .NET Data Types
- ✓ Arrays
- ✓ More Collections (list, dictionary, stack, queue)
- ✓ Classes and objects (OOP)

- Program Logic

- ✓ Statements and expressions
- ✓ Conditional logic (if)
- ✓ Repeating logic (for, foreach, do, while)
- ✓ Methods (functions / procedures)
- ✓ Classes and objects (OOP)
- ✓ Frameworks (MVC)

- Input / Output

- User
  - ✓ Console read / write
  - ✓ HTML / CSS
  - ❖ Front-end frameworks (HTML / CSS / JavaScript)
- Storage
  - ✓ File I/O
  - ✓ Relational database
  - ❑ APIs

# Module 4

- No pairs exercises during Module 4
- Student\_tutorial available most days, starting tonight
  - To enhance reading before lecture

# Client-side Scripting

- What
  - Code executes in the browser
  - Interacts with the HTML of the page (the DOM)
  - **JavaScript** is the language all browsers understand
- Why
  - Validation on the client
    - Immediate response to the user
    - Fewer posts to the server
  - Page interaction
    - Show or hide sections of the page dynamically
    - Dynamically update portions of the page (no refresh / scroll to top)
  - Event responses (button push)

# Client-side Responsibilities

- **HTML** provides the **content**
  - Server may dynamically generate the HTML
- **CSS** provides the **style**, or look-and-feel
  - Static style sheets usually provide the style
- **JavaScript** provides the **behavior** and user-interaction
  - Interacts with the HTML page, and may interact with the server

# Variables

- let and const
- var – Do not use (best practice)
  - Allows multiple declarations without warning
  - Function scope (vs. block scope)
- Names
  - Usually camel-case in JavaScript
  - Const may be upper-case

# Data Types

- Loosely typed
  - JavaScript infers the type from the value
  - Same variable can hold different types over time
- Number, String, Boolean, Object (includes array), undefined
- JavaScript does type coercion as necessary



# Null vs. Undefined

- `null` is a value of type *Object*
- `undefined` is a value of type *undefined*
- `null` must be assigned. It means *nothing*.
- `undefined` occurs from the `"let var_name;"` statement
  - It also may be assigned
- `null !== undefined`, but `null == undefined`



Let's  
Code



# Strict and loose equality

- === vs. ==
- === means types and values are equal (strict equality)
- == means values are equal (loose equality)
  - Types are coerced
- !== and != are the "not equal" equivalents
- **Falsy** values:
  - When coerced to Boolean, value is false
  - false, 0, "", null, undefined, NaN
  - All other values are **Truthy**
- More craziness: <https://codeburst.io/javascript-double-equals-vs-triple-equals-61d4ce5a121a>

Let's  
Code

# Branching

- ```
if (condition) {  
    // do something  
}
```
- ```
if (condition) {  
    // do something  
} else {  
    // do something else  
}
```
- ```
if (condition1) {  
    // do this  
} else if (condition2) {  
    // do that  
} else {  
    // do the other  
}
```
- **switch** also works

# Loops

- `for (let i = 1; i <= 5; i++) {  
 console.log("Hello World!");  
}`
- while and do also exist
- No foreach!
  - (at least, not yet 😊)

# Arrays

- `let scores = [];`  
`let grades = [90, 80, 70];`  
`scores[10] = 100;`
- Size is changeable!
  - `scores.length` gives the number of elements
- Functions:
  - `push`, `pop` (end of array)
  - `unshift`, `shift` (beginning of the array)
  - `indexOf`, `lastIndexOf` (finds an element)



Let's  
Code

# Objects

- { } denotes an object
- Key : value pairs, separated by commas

```
const person = {  
  firstName: "Bill",  
  lastName: "Lumbergh",  
  age: 42,  
  employees: [  
    "Peter Gibbons",  
    "Milton Waddams",  
    "Samir Nagheenanajar",  
    "Michael Bolton"  
  ]  
};
```

Let's  
Code

# Functions (think methods)

- **function** keyword
- Function name
  - Usually camel-case
- No return type
- Parameter names
  - No type defined
- **return** statement



Let's  
Code

# Built-in Functions

- String methods

- [https://www.w3schools.com/js/js\\_string\\_methods.asp](https://www.w3schools.com/js/js_string_methods.asp)
- [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Text formatting](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Text_formatting)

- Numbers, Math and Dates

- [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Numbers and dates](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Numbers_and_dates)
- [https://www.w3schools.com/js/js\\_number\\_methods.asp](https://www.w3schools.com/js/js_number_methods.asp)
- [https://www.w3schools.com/js/js\\_math.asp](https://www.w3schools.com/js/js_math.asp)
- [https://www.w3schools.com/js/js\\_dates.asp](https://www.w3schools.com/js/js_dates.asp)



Let's  
Code



# JavaScript vs C#

|                                          | JavaScript                                                                                                                        | C#                                                                              |
|------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| Variables                                | <code>let age = 35;</code> <ul style="list-style-type: none"><li>Also, <code>const</code>, <code>var</code> (don't use)</li></ul> | <code>int age = 35;</code>                                                      |
| Data Types                               | Number, String, Boolean, Symbol, Object, null, undefined                                                                          | int, short, long, float, decimal, single, double, string bool...                |
| Strings                                  | Single or double quotes                                                                                                           | Double quotes                                                                   |
| Undefined data type                      | "undefined" is both a datatype, and a value                                                                                       | Not applicable                                                                  |
| String interpolation (template literals) | <code>`My name is \${firstname} \${lastname}`</code>                                                                              | <code>\$"My name is {firstname} {lastname}"</code>                              |
| Equality Comparison                      | <code>==</code> , <code>===</code> , <code>!=</code> , <code>!==</code>                                                           | <code>==</code> , <code>!=</code>                                               |
| Array                                    | Size can change<br>Multiple types<br>Directly address even new elements                                                           | Size fixed and set when allocated<br>Single data type<br>Must address in-bounds |
| Functions / methods                      | <code>function myFunction(param1)</code>                                                                                          | <code>public int myMethod(string param1)</code>                                 |