Cornell Webdev Club

Workshop #2: Introduction to JavaScript for Interactivity & Git and GitHub

March 4, 2025

Attendance

Agenda:

1.	• JavaScript Fundamentals
2.	• DOM Manipulation
3.	• Git and GitHub Basics
4.	• Homework

Goal:

Learn about Git and GitHub Environment and the basics of JavaScript to add dynamic behavior to web pages

1. JavaScript Fundamentals

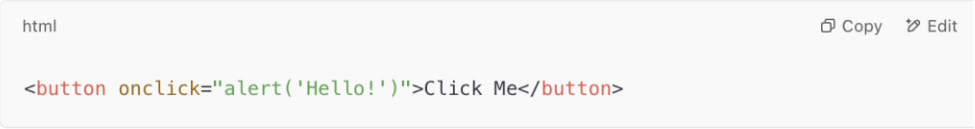
1. JavaScript Fundamentals

JavaScript = "a scripting or programming language that allows you to implement complex features on web pages"

- Adds interactivity and functionality to web pages
- Runs in the browser and supports event-driven programming

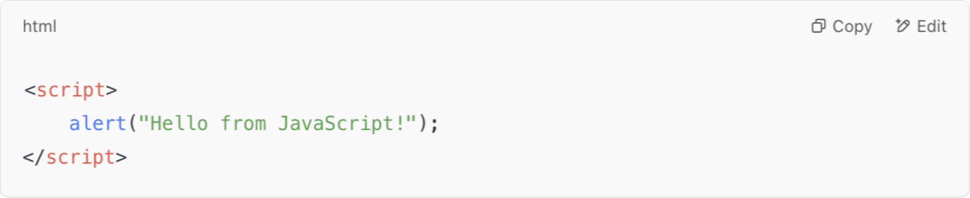
Embedding JS into HTML:

- Inline HTML



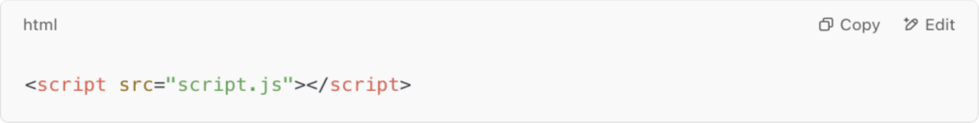
```html
<button onclick="alert('Hello!')">Click Me</button>
```

- Internal JS



```html
<script>
    alert("Hello from JavaScript!");
</script>
```

- External JS



```html
<script src="script.js"></script>
```

  - Create a new .js file in the root of repo
  - Add this html element to embed JS into HML

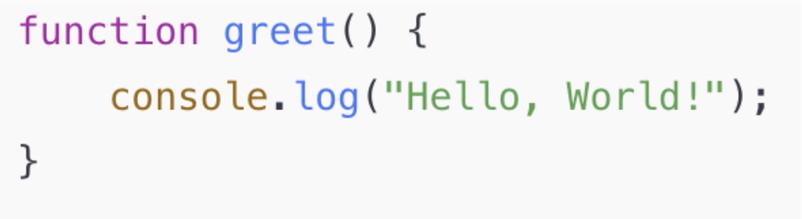
1. JavaScript Fundamentals

JavaScript Variables = a container that stores data values

- Use:
  - var = global or function-scoped, can be re-declared
  - let = block-scoped, cannot be re-declared in the same scope
    - Block-scoped: only accessible within the block where they are defined
  - const = block-scoped, must be assigned a value upon declaration and CANNOT change

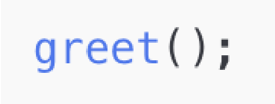
JavaScript Functions = a reusable block of code designed to perform a specific task. It helps organize and reuse code efficiently

- Function Declaration



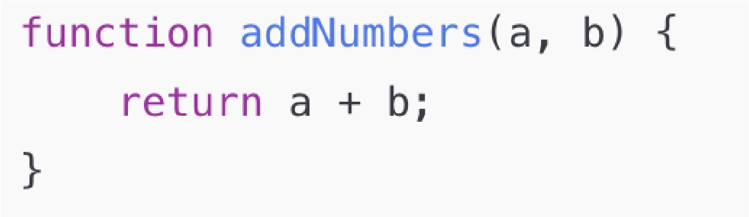
```
function greet() {
    console.log("Hello, World!");
}
```

- Function Call: invoking the function



```
greet();
```

- Function with Parameters and return value



```
function addNumbers(a, b) {
    return a + b;
}
```

  - Parameters = inputs that receives value when function is called
  - Return = returns statement in server

1. JavaScript Fundamentals

JavaScript Events = action or occurrence detected by JS, often triggered by user interaction (clicking, typing, scrolling)

JS Event Types:

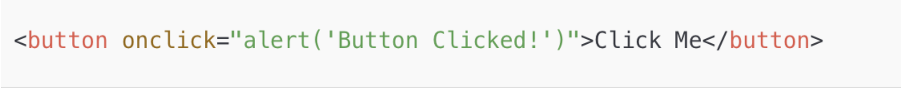
- Mouse Events
  - Use "click", "dblclick", "mouseover", "mouseout", "mousedown", "mouseup"
- Keyboard Events
  - Use "keydown", "keyup", "keypress"
- Form Events
  - Use "submit", "change", "focus", "blur"
- Window Events
  - Use "load", "resize", "scroll", "unload"

1. JavaScript Fundamentals

JavaScript Event Handling = process of responding to the events created by user interactions

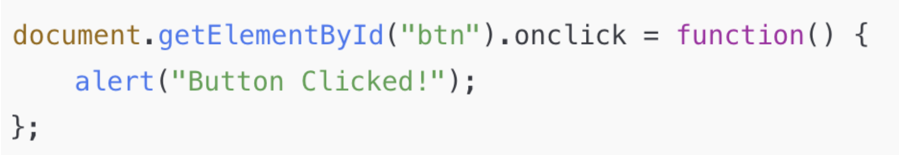
Event Handling Methods:

- Inline Event Handling (not recommended)



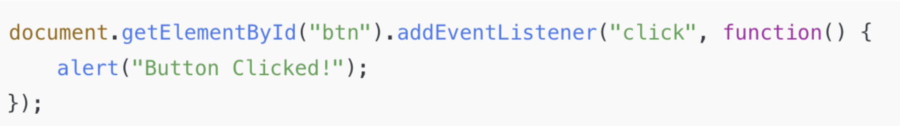
```
<button onclick="alert('Button Clicked!')">Click Me</button>
```

- JavaScript Property



```
document.getElementById("btn").onclick = function() {
    alert("Button Clicked!");
};
```

- "addEventListener()" (best practices)



```
document.getElementById("btn").addEventListener("click", function() {
    alert("Button Clicked!");
});
```

Explanation:

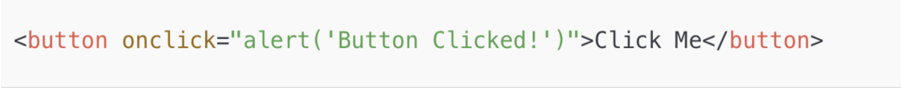
- All of these methods do the same actions of alerting the browser of a message "Button Clicked" when the button element is clicked
- HOWEVER, the "addEventListener()" is best because:
  - Supports multiple Event Handlers
  - Works on any event type
  - Supports Event Capturing & Bubbling

1. JavaScript Fundamentals

JavaScript Event Handling = process of responding to the events created by user interactions

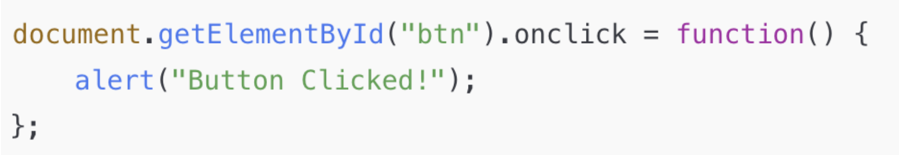
Event Handling Methods:

- Inline Event Handling (not recommended)



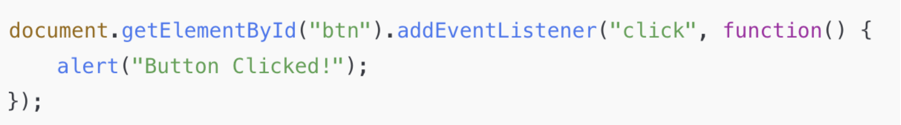
```
<button onclick="alert('Button Clicked!')">Click Me</button>
```

- JavaScript Property



```
document.getElementById("btn").onclick = function() {
    alert("Button Clicked!");
};
```

- "addEventListener()" (best practices)



```
document.getElementById("btn").addEventListener("click", function() {
    alert("Button Clicked!");
});
```

Explanation:

- All of these methods do the same actions of alerting the browser of a message "Button Clicked" when the button element is clicked
- HOWEVER, the "addEventListener()" is best because:
  - Supports multiple Event Handlers
  - Works on any event type
  - Supports Event Capturing & Bubbling

Demo: Practicing Event Handling to Modify Content & Style

2. DOM Manipulation

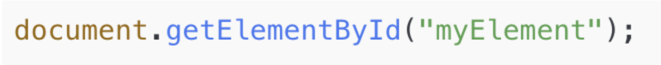
2. DOM Manipulation

Document Object Model (DOM) = structured representation of an HTML document

- JS can interact with the DOM to dynamically update content, styles, and behavior on web page

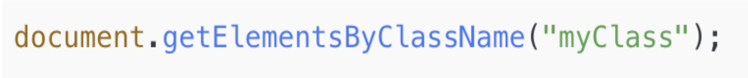
Key Concepts to DOM Manipulation

- Selecting Elements
  - By ID



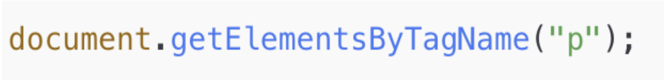
```
document.getElementById("myElement");
```

  - By Class



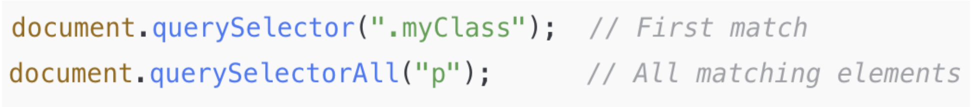
```
document.getElementsByClassName("myClass");
```

  - By Tag Name



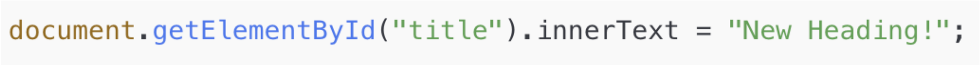
```
document.getElementsByTagName("p");
```

  - By Query Selector



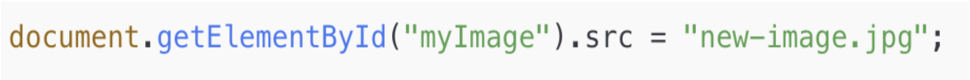
```
document.querySelector(".myClass");  // First match
document.querySelectorAll("p");      // All matching elements
```

- Modifying Elements
  - Change Content



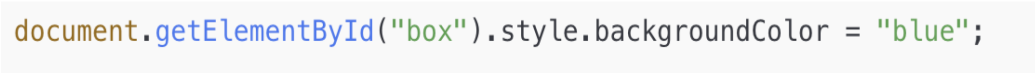
```
document.getElementById("title").innerText = "New Heading!";
```

  - Modify Attributes



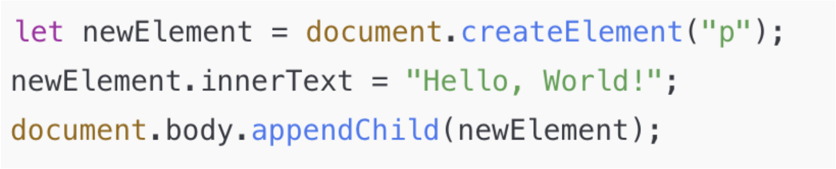
```
document.getElementById("myImage").src = "new-image.jpg";
```

  - Change Styles



```
document.getElementById("box").style.backgroundColor = "blue";
```

  - Create & Append Elements

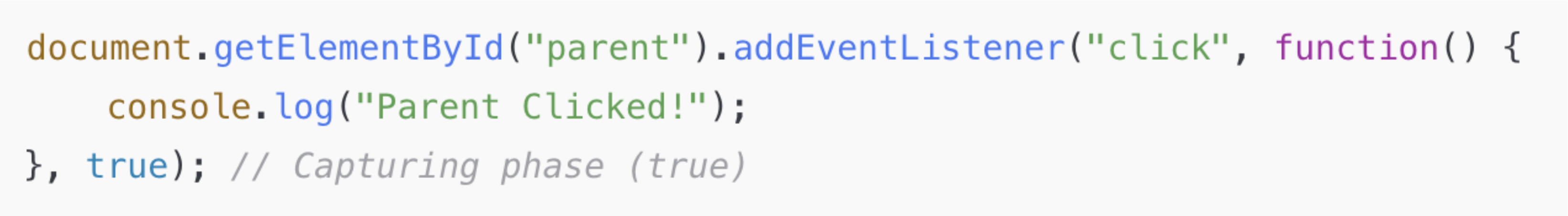


```
let newElement = document.createElement("p");
newElement.innerText = "Hello, World!";
document.body.appendChild(newElement);
```

2. DOM Manipulation

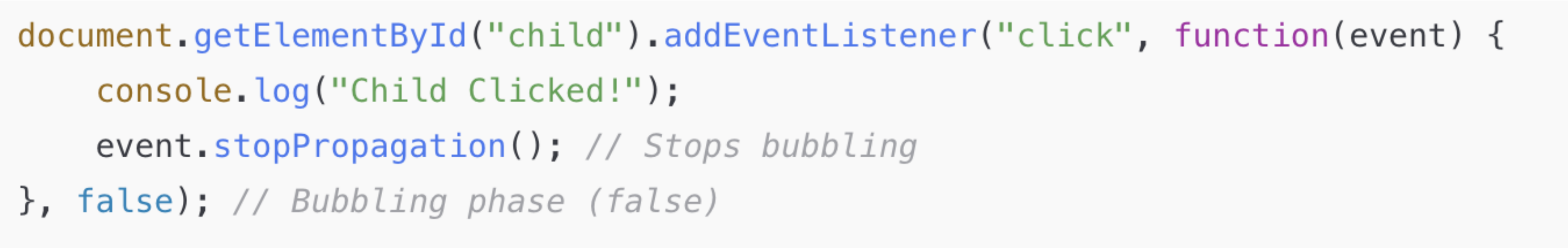
Key Concepts to DOM Manipulation (continued):

- Event Propagation
  - Event Propagation = the process of how events travel through the DOM
    - **Bubbling:** event moves **from the target element up** to its parent



```
document.getElementById("parent").addEventListener("click", function() {
    console.log("Parent Clicked!");
}, true); // Capturing phase (true)
```

- **Capturing:** event moves **from the parent down** to target element



```
document.getElementById("child").addEventListener("click", function(event) {
    console.log("Child Clicked!");
    event.stopPropagation(); // Stops bubbling
}, false); // Bubbling phase (false)
```

- **Stopping Propagation:** use `event.stopPropagation()` to prevent further event propagation

3. Git and GitHub Basics

3. Git and GitHub Basics

Git = “a free, open-source version control system that tracks changes to computer files”

- Helps developers collaborate and manage project history
- Works locally on your computer

GitHub = cloud-based Git hosting service founded in 2008 that makes tools which integrate with git

- Enables remote collaboration and code sharing
- Provides features like pull requests, issue tracking, and CI/CD.

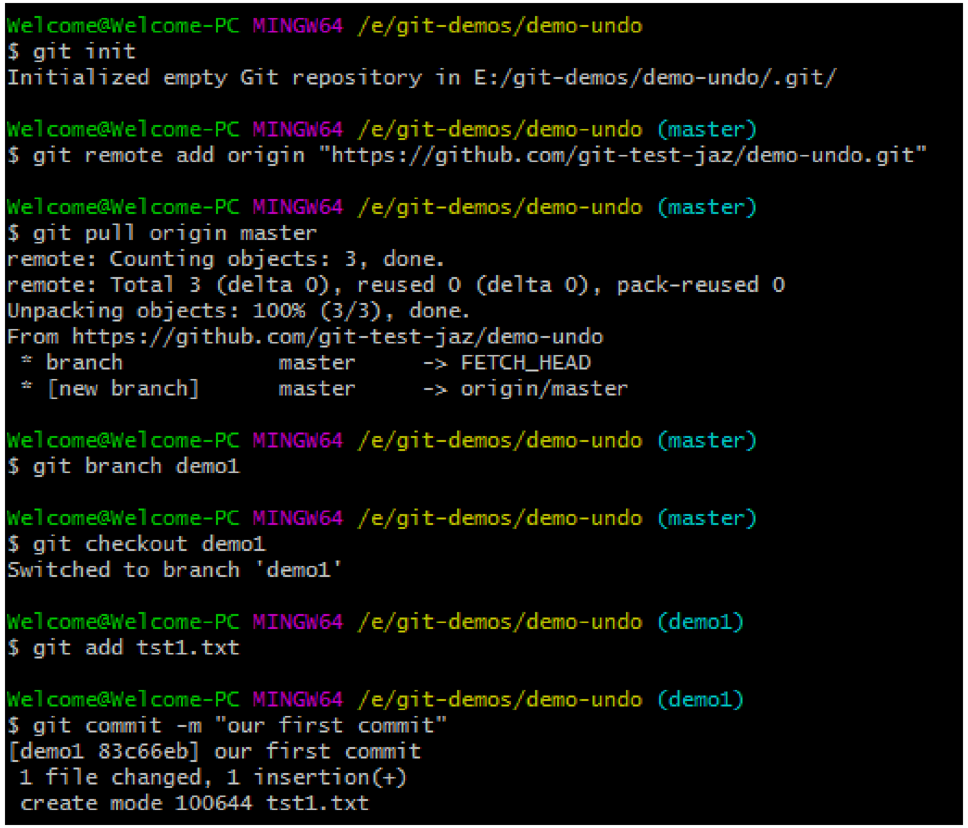
Parts to Git:

- Repository (Repo): database containing all the information needed to retain and manage the revisions and history of a project
- Commit: A snapshot of changes made to files
- Branch: A separate line of development
- Merge: Combining changes from different branches
- Remote & Local: Local repo is on your computer; remote is on GitHub

3. Git and GitHub Basics

How to Commit Your Work:

1. git init
2. git add .
3. git commit -m "initial commit"
4. git remote add origin [LINK]
5. git push -u origin main



```
Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo
$ git init
Initialized empty Git repository in E:/git-demos/demo-undo/.git/

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (master)
$ git remote add origin "https://github.com/git-test-jaz/demo-undo.git"

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (master)
$ git pull origin master
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/git-test-jaz/demo-undo
 * branch            master     -> FETCH_HEAD
 * [new branch]      master     -> origin/master

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (master)
$ git branch demo1

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (master)
$ git checkout demo1
Switched to branch 'demo1'

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (demo1)
$ git add tst1.txt

Welcome@Welcome-PC MINGW64 /e/git-demos/demo-undo (demo1)
$ git commit -m "our first commit"
[demo1 83c66eb] our first commit
 1 file changed, 1 insertion(+)
 create mode 100644 tst1.txt
```

Demo: How to Set Up Git and GitHub Account

4. Homework

4. Homework

Homework: Enhancing Bio Webpage

Instructions:

- Enhance your bio webpage by:
  - Adding a button
  - When button clicked, updates content or styling
  - (OPTIONAL) Add other JS features

- Stage, Commit, and Push Website on GitHub