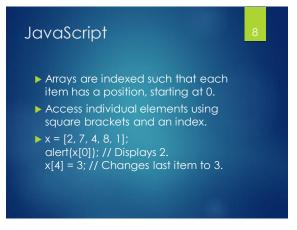






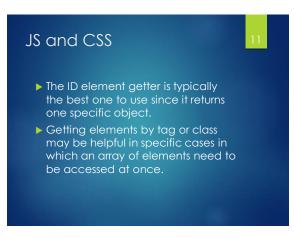
Variable values can be changed after they are first created. var course = "C\$1033"; document.write (course); course = "C\$2033"; document.write (course); Code is executed in top-to-bottom order, so this would first write "C\$1033" and then "C\$2033".

JavaScript There are several types of variables: String – text, surrounded by quotations x = "hello"; Integer – whole number x = 13; Float/Double – decimal number x = 7.3 Array – list of multiple items x = [2, 7, 4, 8, 1];









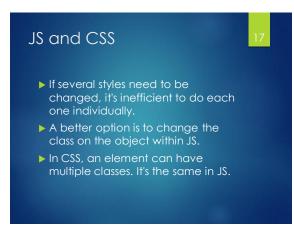






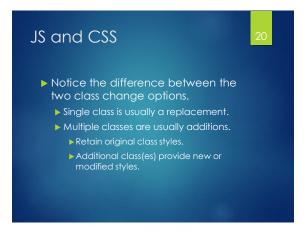




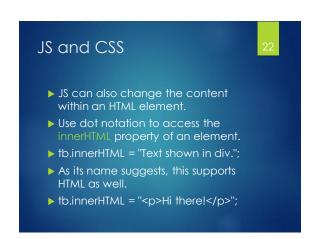












Event handling Style/content changes will typically be done as a result of an event. Events are a huge part of JS! User input events Mouse-based Keyboard-based Load event Timer events

Event handling Events are handled with event listeners applied to HTML elements. Event listeners are always watching for specific events to occur. When an awaited event occurs, the listener detects it and triggers the event's code.

Event handling There are two main ways to add event listeners to elements. Inline: attach the event listener as an attribute in the HTML element tag. Dot Notation: use the addEventListener function as a property on the element using dot notation.



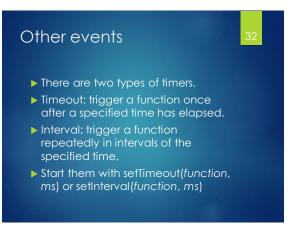




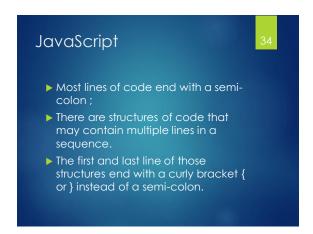
Event handlers > div id="box" onclick="document.getElementByl d("box").style.width = '300px"></div> In this example, the style change applies to the element itself, so we can use a shortcut: this > div id="box" onclick="this.style.width = '300px"></div>

Dot notation is the other method for creating event listeners. This can go in internal or external JavaScript code (after the element is created!) box.addEventListener("click", function() { this.style.width = "300px" });

Other events We talked about mouse events and keyboard events already. There are two other common event types in JS: the loading of the webpage and custom timers. The loading is simple. Just attach an onload listener to the body and it will trigger when everything loads.







Conditionals Conditionals are an important structure in programming. Portions of code will only execute if specific conditions are met. For example, checking if a number is less than 10 The if-else statement is used for conditionals in JavaScript.



Conditionals We can add multiple conditionals using the else if operator. if (x == 1) { // do stuff. } else if (x == 2) { // do different stuff. } else if (x == 3) { // do other different stuff. }

```
Conditionals
The else operator is a catch-all for any cases not yet accounted for.
if (x == 1) {
// do stuff.
} else if (x == 2) {
// do different stuff.
} else {
// do other different stuff.
}
```

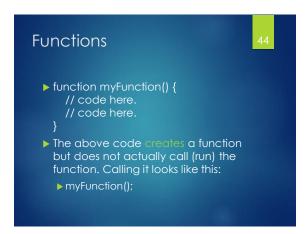
```
Conditionals
Note that once a condition is satisfied, the "else-if" statements below it will not be checked.
x = 2; if (x < 5) {
// do stuff.
} else if (x < 10) {
// do different stuff.
}</li>
```

```
Suppose we received the user's age online and then checked if the user was allowed to buy booze...
if (age >= 19) {
    canDrink = true;
} else {
    canDrink = false;
}
```

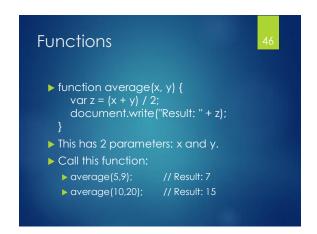
```
Conditionals
Conditionals might be composed of multiple sub-conditions that all have to be met.
We combine them with the && (and) operator.
if (age >= 19 && pregnant != true) { canDrink = true; } else { canDrink = false; }
```

```
Conditionals
There also might be sub-conditions such that at least one must be met.
In this case, use the || (or) operator.
if (age >= 19 || allowed == true) {
    canDrink = true;
} else {
    canDrink = false;
}
```

Functions Another common structure in code is a function. A function is a process that can be executed at any time, and any number of times. Great for routine processes that need to be used multiple times.



```
Functions
Many functions have input parameters which are placed within the parentheses.
Parameters make the function reusable and flexible to work in different scenarios.
To call a function with parameters, include the parameter values in the function call parentheses.
```



```
Another special code structure is the loop. This is used to run code repeatedly in a row.
There are two main types of loops: while-loop and for-loop but we will focus on the for-loop.
They contain 3 parts: variable initialization, condition, increment.
```

```
Loops

> for (x = 0; x < 5; x++) {
    alert(x);
    }

> They also work great with arrays.

> data = [9, 4, 7, 3];
    for (x = 0; x < data.length; x++) {
        alert(data[x]);
    }
</pre>
```