

JAVASCRIPT

Functions



Functions in JavaScript

2

- A JavaScript function is a block of code designed to perform a particular task.
- The function is executed when “something” invokes it (calls it)
- $f(x) = x + 2$

```
function multiply(param1, param2)
{
    return param1 * param2
}
```

Function won't
execute unless it
is invoked



Functions in JavaScript

3

- A function can be named or it can be anonymous

```
function(param1, param2)
{
    return param1 * param2
}
```

- The params are items of data that the function needs to perform its task
- Not passing a required parameter will result in an error
- Can have zero parameters but still requires empty parameters

ES6 Arrow functions

4

ES6

```
(param1, param2) =>  
{  
    return param1 * param2;  
}
```

ES5

```
function(param1, param2)  
{  
    return param1 * param2;  
}
```

- Arrow functions are more concise, developer can achieve the same functionality with fewer lines of code.
- Support concise function expressions

Assign function to a variable

5

```
let multiply = (param1, param2) =>
{
  return param1 * param2;
}
multiply(3,3)
```

```
let multiply = function(param1, param2)
{
  return param1 * param2;
}
multiply(3,3)
```

Returning from a function

6

- Functions can return values to their calling environments
- Use the ***return*** statement to do this

```
function divide(numerator, denominator)  
{  
    return numerator / denominator  
}
```

Returning from a function


7

- The returned value from a function can be assigned to a variable.

```
function incrementAge(myAge)
{
    myAge++;
    return myAge;
}
```

```
let incAge = incrementAge(26);
alert("Incremented age is " + incAge);
```

Invoking the function
and passing
arguments



Functions

8

- ❑ Functions consist of
 - ❑ Unique name (cannot be keywords)
 - If they are named!
 - ❑ Parameters (again cannot be keywords)
 - ❑ **Don't** declare variables as parameters (**let** param1)
 - ❑ Code block to execute
 - ❑ Will only return once, but can use conditional statements to control the execution of the code and have multiple return statements

Invoking a function from HTML

9

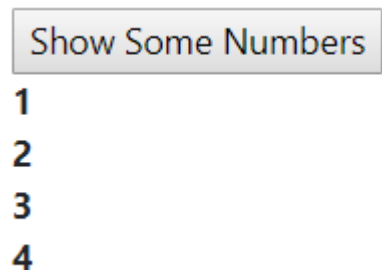
```
<script>
function numbers() {
  let sHTML = "<b>";
  sHTML += 1 + "<br>" + 2 + "<br>" + 3 + "<br>" + 4 + "<br>";
  sHTML += "</b>";
  clientSideContent.innerHTML = sHTML;
}
</script>
<button onClick="numbers()">Show Some Numbers</button><br>
<div id="clientSideContent">Something here</div>
```

DOM Manipulation

Inline JS within our pages

-Note we can also link to a JS file too!

-Try and reproduce the same functionality but this time by using a separate JS file



Exercise: Functions

10

- ❑ Copy the JS code from the previous slide and place it into a separate JavaScript file.
- ❑ Modify the function “numbers” to accept a param
- ❑ If the argument passed in is 1 then the numbers printed out should be 1, 2, 3, 4.
- ❑ If the argument passed in is something else, then the numbers printed out should be 2, 4, 6, 8

JAVASCRIPT

Events



Events

12

- Actions that can be responded to by JavaScript
- Every element on a page has certain events which can trigger some JavaScript code
 - ▣ We can identify when a user clicks a button with the **onClick** event
 - ▣ Can then assign a function to run when the event is identified
 - ▣ Events defined as an attribute in the **HTML Tag**

Examples of Events

13

- ❑ A mouse click
- ❑ A web page or an image loading
- ❑ Moving the mouse over a hot spot on the web page



A login form enclosed in a dotted border. It contains the following elements:

- A label "Name:" followed by a text input field.
- A label "Pass:" followed by a text input field.
- A label "Save Password:" followed by an unchecked checkbox.
- A "Submit" button.

Other Mouse Events

14

- **onClick**

- Triggered when the mouse clicks an element

- **onMouseDown**

- Triggered when the mouse button is pressed

- **onMouseUp**

- Triggered when the mouse button is released.

Selecting and De-Selecting Elements

15

- All three normally used with form input elements (text boxes, buttons etc.)
- onFocus
 - ▣ Triggered when an element gets focus
 - ▣ E.g. an element that is clicked is said to be in focus
- onBlur
 - ▣ Triggered when an element loses focus
- onChange
 - ▣ Triggered when the content of an element changes

As the mouse moves over HTML elements

16

□ **onMouseOver**

- Triggered for an element when the mouse cursor is moved over that element
- e.g. moving the mouse over an image ('rollover')

□ **onMouseOut**

- Triggered for an element when the mouse cursor is moved away from that element
- e.g. moving the mouse out of the image

Other Keyboard Events

17

- `onKeyDown`
 - ▣ Triggered when a keyboard key is pressed
- `onKeyUp`
 - ▣ Triggered when a keyboard key is released
- `onKeyPress`
 - ▣ Triggered when a keyboard key is pressed or held
- `onSelect`
 - ▣ Triggered when text is selected

Exercise: Multiply numbers from text input

18

- Place a text input on your web page
- Ask the user to pop in a number into the textbox using the placeholder attribute
- Place a button underneath it called “Multiply”
- When the user clicks on the button a function is invoked which takes the value from the text box, multiplies it by 3 and pops it back into the same text box
- Note that if you place an ID on the input, you can access the value the user enters via `myelement.value`