COMP4021
Internet Computing

Introduction to JavaScript

Gibson Lam and David Rossiter

JavaScript

- JavaScript (JS) is a scripting language used inside a browser (also other places)
- Just to be clear: although they have a similar name, JavaScript and Java are two completely different languages

Putting JavaScript in Webpages

- You put JavaScript inside `<script>…</script>`, like this:

```
<script>
alert("How are you?");
</script>
```

This example has one line of JavaScript code

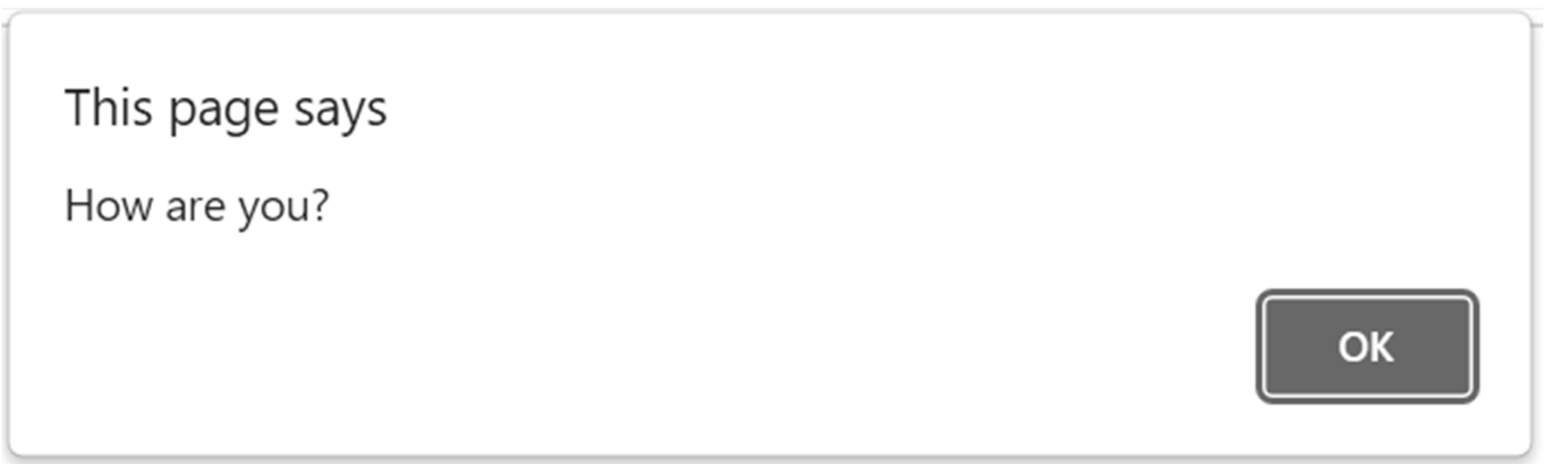
- The script can then go anywhere inside an HTML page, e.g. inside the head section or the body section of the page
- A simple example is shown on the next slide

A Simple Example

```
<!DOCTYPE html>
<html>
<head>
  ...
  <script>
  alert("How are you?");
  </script>
</head>
<body>
    ...
</body>
</html>
```

- Here we use JavaScript to show a small window that has a message when the HTML page is loaded in a browser

Showing a message using alert()



The Order of Running JavaScript

- If you put multiple JavaScript code inside an HTML file, the execution order follows the order that they are inside the file
- This happens no matter where you put the `<script>` tag, i.e. in the head or body section of the file

```
<!DOCTYPE html>
<html>
<head>
    ...
    <script>...</script>
    ...
    <script>...</script>
    ...
</head>
<body>
    ...
    <script>...</script>
    ...
    <script>...</script>
    ...
</body>
</html>
```

Execution order

Using Variables

- You create variables using `let`, for example:

```
let myname = "Genius";
let myIQ = 250;
```

- You do not need to worry about data types in JavaScript most of the time
- You can also create one without any initial value:

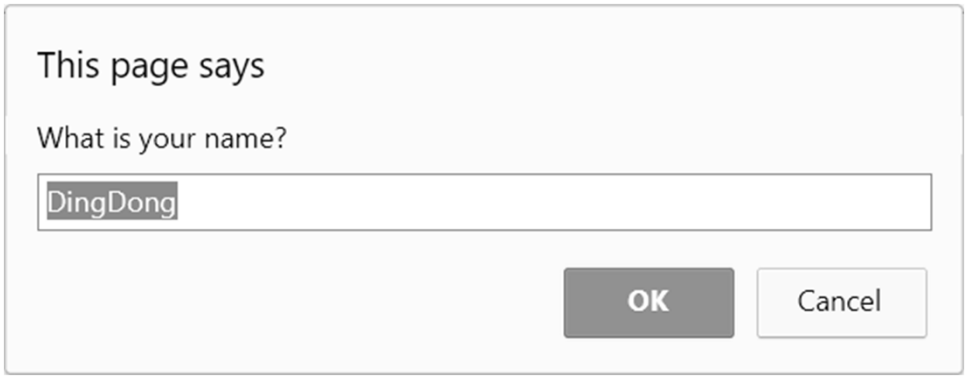
```
let mymoney;
```

Asking for Text Input

- A simple way to ask for text input is using `prompt()` which shows a small window for entering text, like this:

```
<script>
let name = prompt("What is your name?",
                  "DingDong");
alert("Hi! " + name + ".");
</script>
```

String concatenation



Converting To Numbers

- One common mistake is using variables with text content as if they are numbers, i.e.:

```
myage = "25";
myage = myage + 20; // expect 45,
                    // but get 2520!
```

- Just to be safe, you can convert things into integers using `parseInt()` or floating point numbers using `parseFloat()`, i.e.:

```
myage = "25";
myage = parseInt(myage) + 20; // 45
```

A Number? Not a Number?

- You may also check if a variable is a number or not before working with it, e.g.:

```
if (isNaN(myage)) {
    alert("You have a strange age!");
}
```

- `isNaN()` returns true if the input is **n**ot **a** **n**umber, i.e. NaN, and false otherwise

Basic Events

- It is often useful to start JavaScript when a particular *event* occurs
- A typical way to write code for an event is to:
  1. Create a function containing the JavaScript code that you want to run when the event happens
  2. Assign the function to the event
- We will demonstrate two simple events:
  - The load event
  - The click event

Creating Functions

- You create JavaScript functions using the `function` keyword, like this:

```
function fullname(firstname, lastname) {
    let display = lastname + "," +
                  firstname;
    return display;
}
```

- You can then use the function for any event

A Load Event Example

- As you can see, this example is almost the same as the alert example
- The difference here is the alert box is shown when the entire page has been loaded

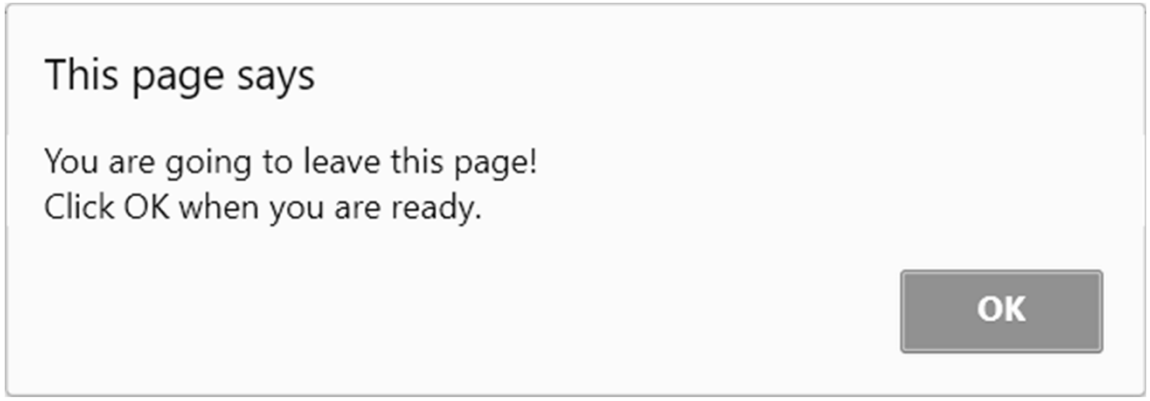
```
<!DOCTYPE html>
<html>
<head>
    ...
    <script>
    function show() {
        alert("How are you?");
    }
    </script>
</head>
<body onload="show()" >
    ...
</body>
</html
```

Run show() when the page is completely loaded

A Click Event Example

- In this example, when you click on the hyperlink, the click event runs `warning()`
- The function then shows an alert before the browser goes to the linked page

```
<script>
function warning() {
    alert("You are...");
}
</script>
...
<a onclick="warning()"
   href="https://cse.hkust.edu.hk">CSE Department</a>
```



Run warning() when the link is clicked

Making Random Numbers

- `Math.random()` gives you a random number from 0 to 1 but excluding 1, i.e. $[0, 1)$
- You can easily make a random integer within a range by combining `Math.random()` and `parseInt()`, i.e.:

```
// Make a random number between 1 and 10
let mynumber =
    parseInt(Math.random() * 10) + 1
```

A Quick Note About Debugging

- The Chrome developer tools have many useful debugging tools
  - You can set breakpoints, pause execution, watch expressions and so on
- You can also use `alert()` for a quick look in your code
- Perhaps a better way is to use `console.log()`
- It outputs content in the console window, which regular users of web page won't see it