香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

VARIABLES

PROF. DAVID ROSSITER

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

AFTER THIS PRESENTATION

- You'll understand different data types in JavaScript

WE WILL LOOK AT

`var`

`typeof`

DATA TYPES

- Number
- String
- Boolean
- Other e.g. Object

NUMBER

- JavaScript has only one type of number
- Can be written with or without a decimal place

```
var number1 = 34.289;
var number2 = 100;
```

- Can use scientific notation

```
var big_number = 123e5;     //12300000
var small_number = 123e-5; //0.00123
```

STRING

- A *string* simply means text
- You can use single or double quotes

```
var name = "David";
var title = 'Professor';
```

- You can use quotes inside a string, as long as they don't match the quotes surrounding the string

```
var message = "It's alright";
```

BOOLEAN

- A Boolean value can only be `true` or `false`

```
var condition1 = true;
var condition2 = false;
```

- Do not confuse Boolean values with String values

```
var myBool = true;       //Boolean type
var myString = "true"; //String type
```

A VARIABLE TYPE CAN CHANGE

- If you do this

```
var storage = "David";
```

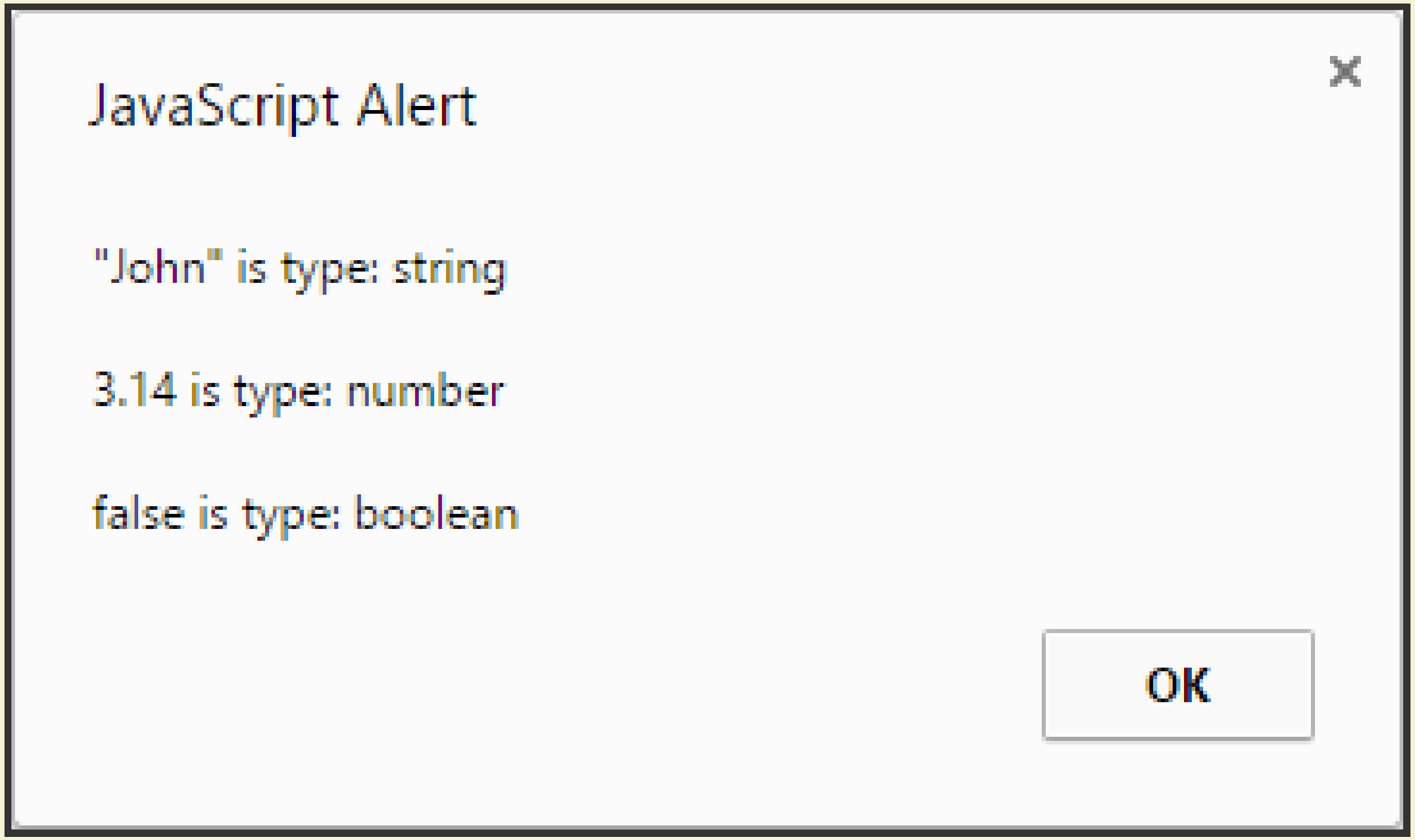
- And then this:

```
storage = 98;
```

- The type of the variable is immediately changed

USING TYPEOF

- You can use the `typeof` operator to check the type of a variable



```
"John" is type: string

3.14 is type: number

false is type: boolean
```

```
<!doctype html>
<html>
<head>
  <title>Variable Type Example</title>
</head>
<body>
  <script>
  alert( '"John" is type: ' + typeof "John" + "\n\n"
        + "3.14 is type: " + typeof 3.14 + "\n\n"
        + "false is type: " + typeof false ) ;
  </script>
</body>
</html>
```

COMMON CHANGES

Code	Quicker Typing
`count = count + 1`	`count++`
`count = count - 1`	`count--`
`count = count + 10`	`count += 10`
`hello = hello + "!"`	`hello += "!"`
`marks = marks - 20`	`marks -= 20`
`pigs = pigs * 5`	`pigs *= 5`
`cakes = cakes / students`	`cakes /= students`

FROM ONE TYPE TO ANOTHER

Function	Meaning
`parseInt()`	Converts to an integer
`parseFloat()`	Converts to a floating point number
`String()`	Converts the value of an object to a string

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

INTRODUCTION TO EVENTS AND FUNCTIONS

PROF. DAVID ROSSITER

THE DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

AFTER THIS PRESENTATION

- You'll appreciate the concept of events
- You'll understand how to use functions

WE WILL LOOK AT

Events	`onload`
Functions	`function`
	`return`

EVENTS

- An event is when something happens
- For example:
  - Click on something
  - Move the mouse
  - Press a key on the keyboard
- You can arrange for some code that you write to be executed when the event occurs

ONLOAD EVENT

- *onload* is triggered when the object has loaded

```
<body onload="alert('Hello!')">
```

. . . the main web page content goes here . . .

```
</body>
```

when body finishes loading code, it will show

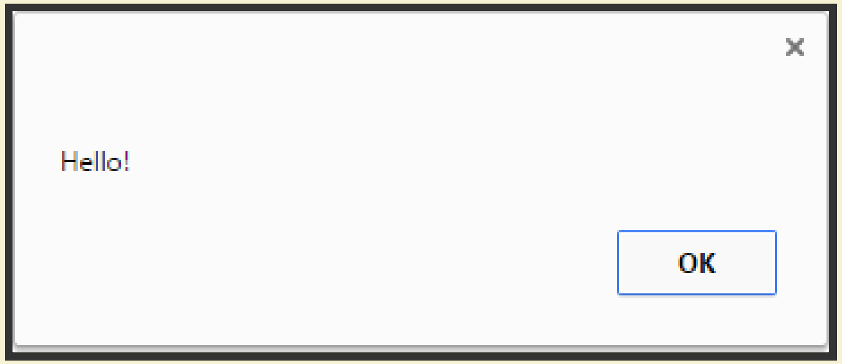
EXAMPLE

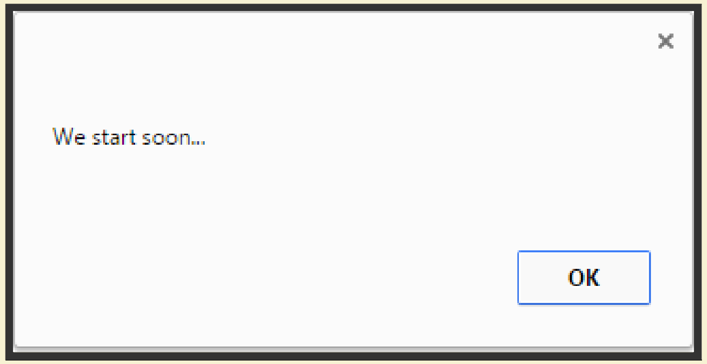
```
<!doctype html>
<html>
    <body onload="alert('Hello!')">
        <p>
            A message is shown as soon
            as the page is loaded.
        </p>
    </body>
</html>
```

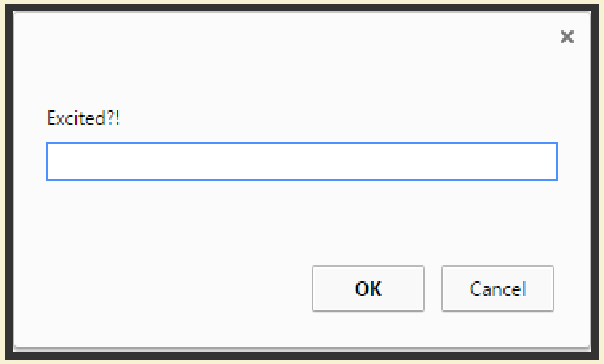
You can execute as much code as you like

```
<!doctype html>
<html>
    <body onload="alert('Hello!');
            alert('We start soon...');
            prompt('Excited?!') ">
        <p>
            3 popup windows are shown as
            soon as the page is loaded.
        </p>
    </body>
</html>
```

通常不这么做，而是将多个语句放在function里


Hello!


We start soon...


Excited?!

FUNCTIONS

- A function is a group of code:

```
function do_something() {
```

  . . . code goes here . . .

```
}
```

- Run the function like this:

```
do_something();
```

```html
<!doctype html>
<html>
    <head>
        <title>Example of a function</title>
        <script>
            function greet_the_user(){
                alert('Hello!');
                alert('We start soon...');
                prompt('Excited?!')
            }
        </script>
    </head>
    <body onload="greet_the_user()">
    </body>
</html>
```

function可以放在head里定义

用event调用

FUNCTION PARAMETERS

You can pass something to a function

```
function purchase( cats ) {
```

. . . *code here uses* cats . . .

```
}
```

- Run the function like this:

```
purchase( 10 );
```

FUNCTION RESPONSE

You can get a response from a function

```
function do_something() {
```

. . . *code here stores something in* answer . . .

```
    return answer; }
```

- Use the function like this:

```
result = do_something();
```

```
<!doctype html>
<html><body onload="check_user_age()" style="position:absolute">
    <h1>This is my naughty home page.</h1>
    <script>
        function check_user_age(){
            if (age_of_user() < 18)
                alert("Please go to another page.");
        }
        function age_of_user(){
            var age_text, age;
            age_text=prompt("What is your age?");
            age=parseInt(age_text);
            return age;
        }
</script></body></html>
```

在一个函数里调用
另一个函数

A RECURSIVE FUNCTION

A function can call itself 递归

```
function do_something( control_value ) {
```

. . . *code here calls* do_something(. . .)

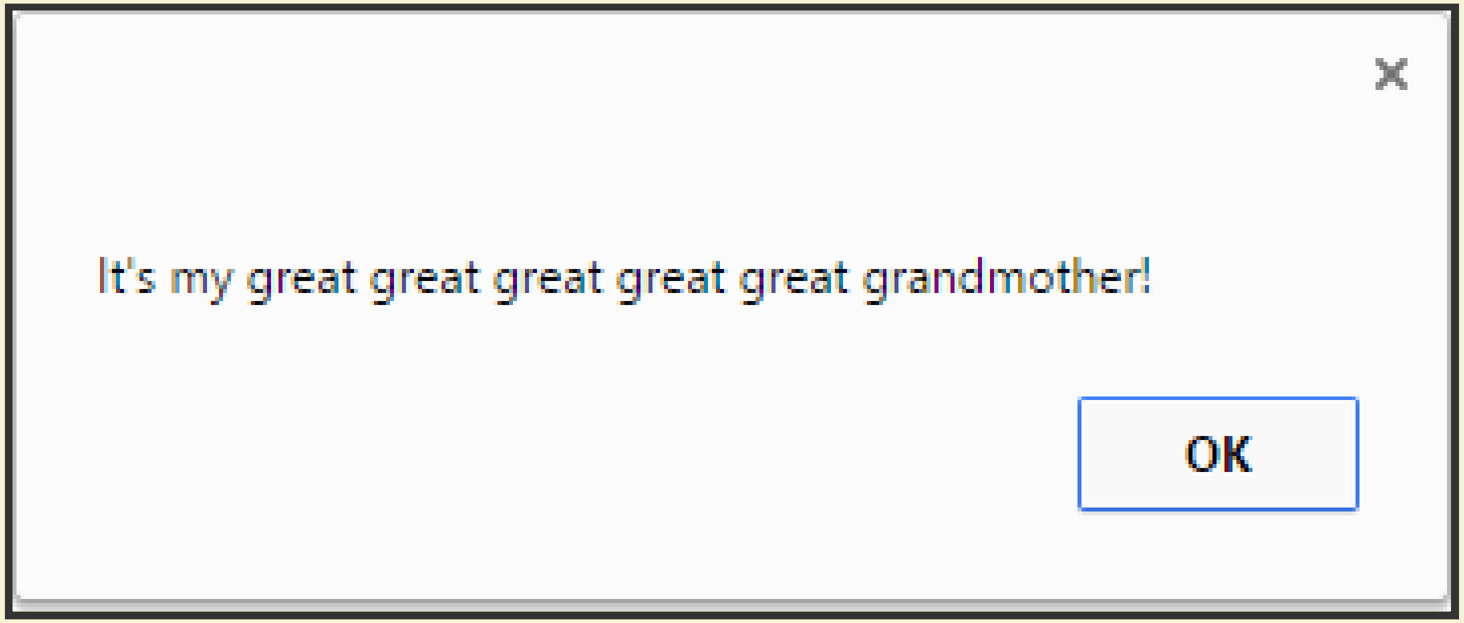
```
}
```

- Start the function like this:

```
result = do_something( 10 );
```

```
<!doctype html>
<html><body>
  <script>
    alert("It's my " + build_great(5) +
          "grandmother!");

    function build_great( depth ) {
      if (depth > 0)
        return "great " + build_great( depth - 1 );
      else
        return "";
    }
</script>
</body></html>
```


It's my great great great great great grandmother!

用console.log(“…”) 代替alert(“…”)，
使得打印只在console发生，用看不到