OBJECT SCOPE

Ultimate JavaScript Objects

INTRODUCTION

- JavaScript has three scopes – *window*, *function* and *eval*
- Object scope is function scope when that function is a method of an object
- All functions that contain the *this* keyword are basically methods as *this* does not evaluate logically outside of a method and usually creates a wrong result

UNDERSTANDING *THIS*

- Value of *this* keyword is equal to the object itself, when *this* is inside one of its bound methods
- Elsewhere, *this* is equal to the *window* variable (usually not what you want!)
- Value of *this* is always provided – no need to ever set it
- Though innocuous, a huge amount of new code constructs are possible
- The value of *this* can change again inside normal functions (but not arrow functions)

WHAT'S UP WITH… STRICT MODE

- Activates streamlined version of JavaScript
- Activated by putting "use strict" at the top of script or function
- The *this* keyword never refers to the window in strict mode

```
"use strict"
```

BINDING SCOPE

- All functions have a bind method which creates a copy of the function with a different value of *this*
- By binding a function to an object, we create a method that can refer to properties of the object itself and its other methods
- Variables available in parent scope of original function still available

OBJECT COMPOSITION

- Many object oriented languages offer inheritance – a hierarchical way of sharing methods between objects
- ES6 introduces classes with standard (classical) inheritance (see next chapter)
- JavaScript offers prototypical inheritance, which is useful but can have puzzling behaviors
- **Composition** is a third way of sharing methods between classes
- Shares methods between objects on a flat hierarchy using *this* and *bind*
- Not possible in many other languages and therefore worth investigating

CONCLUSION

- Using the *this* keyword allows methods to refer to the object of which they are part
- The value of *this* can be changed in any function using the function's bind method
- Unusual cases of the *this* keyword can be avoided by using strict mode
- Creating class-like objects by using many methods and the this keyword is called composition