JavaScript

Stephen P Levitt

School of Electrical and Information Engineering
University of the Witwatersrand

Outline

Functions as Values

Functions are first-class citizens (can be easily passed as arguments to other functions; can be returned from functions; can be assigned to variables or stored in data structures)

```
// illustrating a function *declaration*
function aFunction() {
    console.log("in a function");
};

aFunction(); // invoking the function
```

Function Declaration Hoisting

```
foo()

function foo () {
  let a = 2
  console.log(a)
}
```

Function Expressions

```
// illustrating a function expression
const f = function aFunction () {
  console.log('in a function')
}

f() // invoking the function
```

Anonymous Function Expressions

```
// function expressions can be anonymous
const f2 = function () {
  console.log('in a function')
}

f2() // invoking the function
```

Immediately Invokable Function Expressions - IIFE

```
// function is only ever called once
(function anotherFunction () {
  console.log('in a function')
})()
// // function expression is invoked immediately by trailing ()

anotherFunction() // error, aFunction not defined
```

> *Closure is when a function can remember and access its lexical scope even when it's invoked outside its lexical scope.*
>
> — Kyle Simpson in *You Don't Know JS*

> *A closure is the combination of a function and the lexical environment within which that function was declared.*

How are closures used in practice?

- For emulating private data leading to the module pattern
- For use with browser callbacks