JavaScript Closures

```
var a = 4;
function myFunction() {
    return a * a;
}
```

Variables created **without** the keyword **var**, are always global, even if they are created inside a function.

```
var add = (function () {
    var counter = 0;
    return function () {return counter += 1;}
})();

add();
add();
add();

// the counter is now 3
```

Example Explained

The variable **add** is assigned the return value of a self-invoking function.

The self-invoking function only runs once. It sets the counter to zero (0), and returns a function expression.

This way add becomes a function. The "wonderful" part is that it can access the counter in the parent scope.

This is called a JavaScript **closure. It makes it possible for a function to have "**private**" variables.**

The counter is protected by the scope of the anonymous function, and can only be changed using the add function.