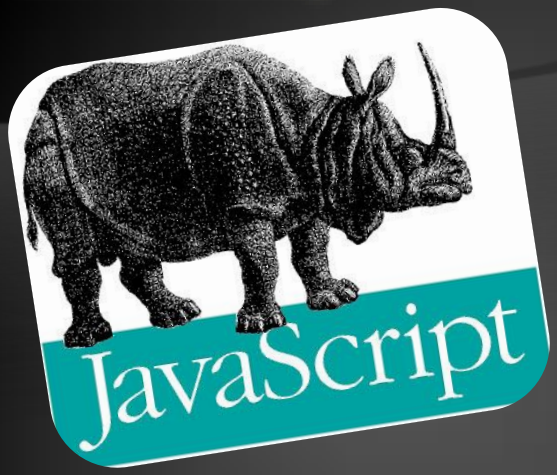
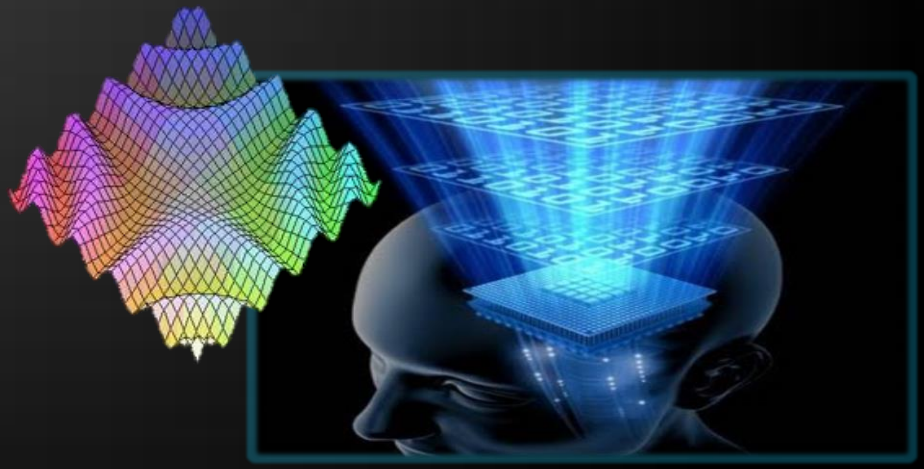
Telerik Academy



JavaScript Patterns

Private/Public fields, Module, Revealing Module



Telerik Software Academy
Learning & Development Team
http://academy.telerik.com

Telerik Academy

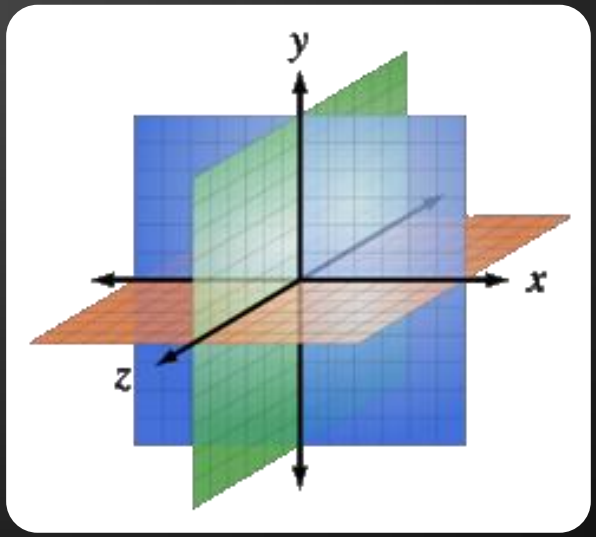
Table of Contents

1. Public/Private fields in JavaScript
2. Module pattern
   1. Pros and cons
   2. Structure
   3. Live demo
3. Revealing module
   1. Pros and cons
   2. Structure
   3. Live demo

Telerik Academy

Public/Private fields

Using the function scope



Public/Private Fields

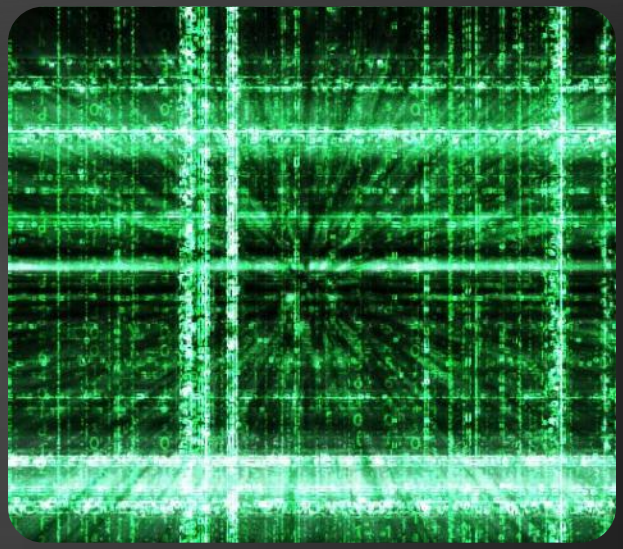
- Each variable is defined:
  - In the global scope (Public)
  - In a function scope (Private)

```
var global = 5;

function myFunction() {
	var private = global;

	function innerFunction(){
		var innerPrivate = private;
	}
}
```

Telerik Academy



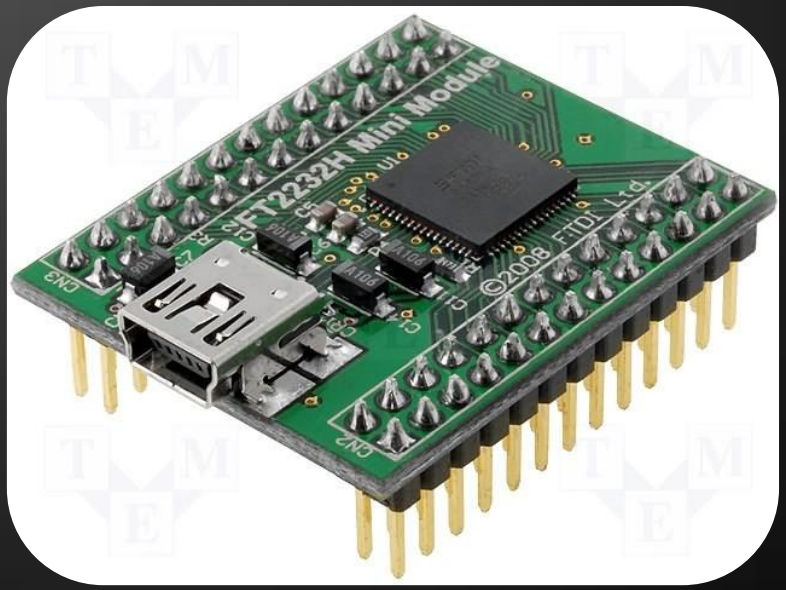
Public/Private fields

Live Demo

Telerik Academy

The Module Pattern

Hide members



Pros and Cons

- Pros:
  - "Modularize" code into re-useable objects
  - Variables/functions not in global namespace
  - Expose only public members
- Cons:
  - Functions may be duplicated across objects in memory
  - Not easy to extend
  - Some complain about debugging

Structure

- Structure:

```
var module = (function() {
        //private variables
        //private functions

        return {
                //public members
        };
});
```

- Usage
  - With the new keyword

Example

- Example:

```
var Calculator = (function(eq) {
	var eqCtl = document.getElementById(eq);

	return {
		add: function(x,y) {
			var val = x + y;
			eqCtl.innerHTML = val;
		}
	};
});

var calculator = new Calculator('eq');
calculator.add(2,2);
```

Summary

- Module pattern provides encapsulation of variables and functions
- Provides a way to add visibility (public versus private) to members
- Each object instance creates new copies of functions in memory
- Extending objects can be difficult since no prototyping is used

Telerik Academy

Module Pattern

Live Demo

Telerik Academy

The Revealing Module Pattern

Reveal the most interesting members

Pros and Cons

- Pros:
  - "Modularize" code into re-useable objects
  - Variables/functions taken out of global namespace
  - Expose only public members
  - "Cleaner" way to expose public members
- Cons:
  - Functions may be duplicated across objects in memory when not using singleton
  - Not easy to extend
  - Some complain about debugging

Structure

- Structure:

```
var module = function() {
        //private variables
        //private functions

        return {
                //public members
        };
}();
```

- Usage
  - Without the new keyword

Example

- Example:

```
var calculator = function(eq) {
       var eqCtl = document.getElementById(eq),
            doAdd = function(x,y) {
                     var val = x + y;
                     eqCtl.innerHTML = val;
                     };

       return {
              add: doAdd
       }; //Expose public member
}('eqCtl');

calculator.add(2,2);
```

Summary

- Module pattern provides encapsulation of variables and functions
- Provides a way to add visibility (public versus private) to members
- Each object instance creates new copies of functions in memory
- Extending objects can be difficult since no prototyping is used

Telerik Academy

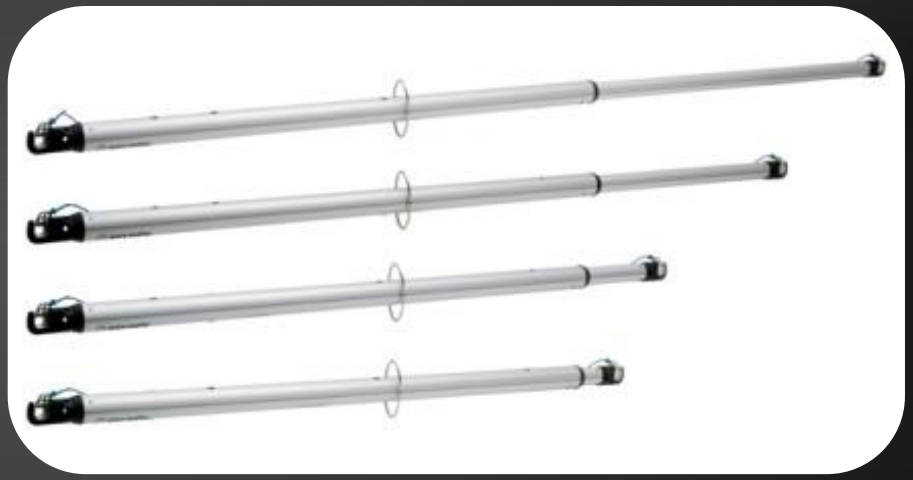
```
var salary = function () {


}();
```

Revealing Module Pattern

Live Demo

Telerik Academy



Extending Existing Module

Live Demo

Telerik Academy

Arrays

Questions?

http://csharpfundamentals.telerik.com

Free Trainings @ Telerik Academy

- "C# Programming @ Telerik Academy
  - csharpfundamentals.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com

Homework

Create the Snake game using the Revealing module pattern. Design the game such that it has at least three modules.