DESIGN PATTERNS

this & that

Immediately Invoked Function Expressions (IIFE)

Closure pattern

Module pattern

Callbacks

“use strict”

THIS

```
var myFunction = function () {
  console.log(this); // this = global, [object Window]
};
myFunction();

var myObject = {};
myObject.myMethod = function () {
  console.log(this); // this = Object { myObject }
};

var nav = document.querySelector('.nav'); // <nav class="nav">
var toggleNav = function () {
  console.log(this); // this = <nav> element
};
nav.addEventListener('click', toggleNav, false);
```

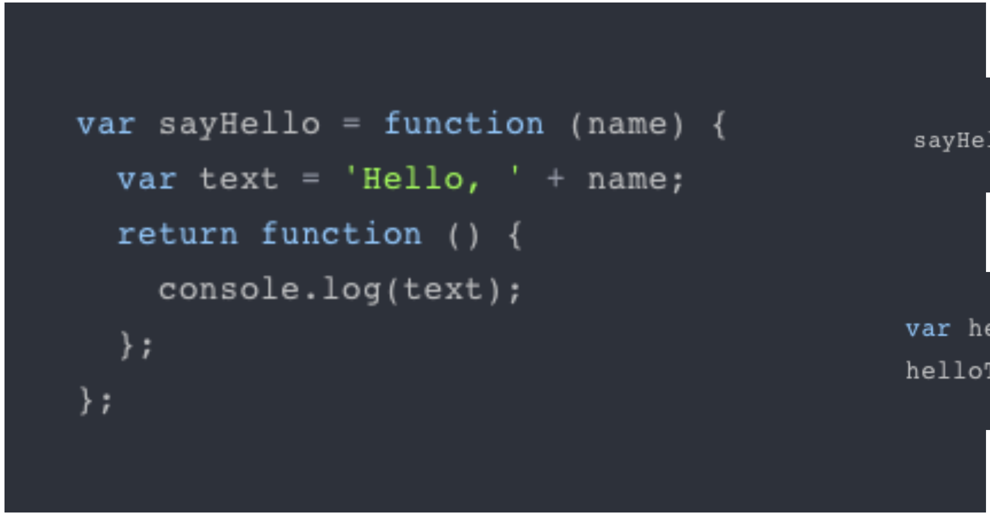
THIS PROBLEMOS

```
var nav = document.querySelector('.nav'); // <nav class="nav">
var toggleNav = function () {
  console.log(this); // <nav> element
  setTimeout(function () {
    console.log(this); // [object Window]
  }, 1000);
};
nav.addEventListener('click', toggleNav, false);
```

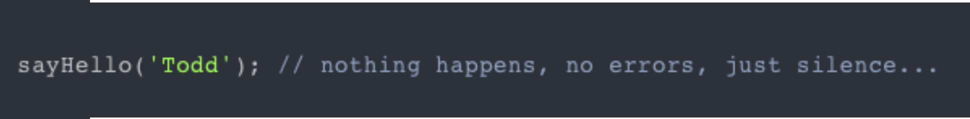
THAT SPRENDIMAS

```
var nav = document.querySelector('.nav'); // <nav class="nav">
var toggleNav = function () {
  var that = this;
  console.log(that); // <nav> element
  setTimeout(function () {
    console.log(that); // <nav> element
  }, 1000);
};
nav.addEventListener('click', toggleNav, false);
```

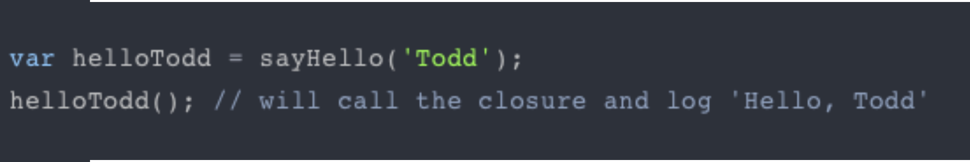
CLOSURE



```
var sayHello = function (name) {
  var text = 'Hello, ' + name;
  return function () {
    console.log(text);
  };
};
```



```
sayHello('Todd'); // nothing happens, no errors, just silence...
```



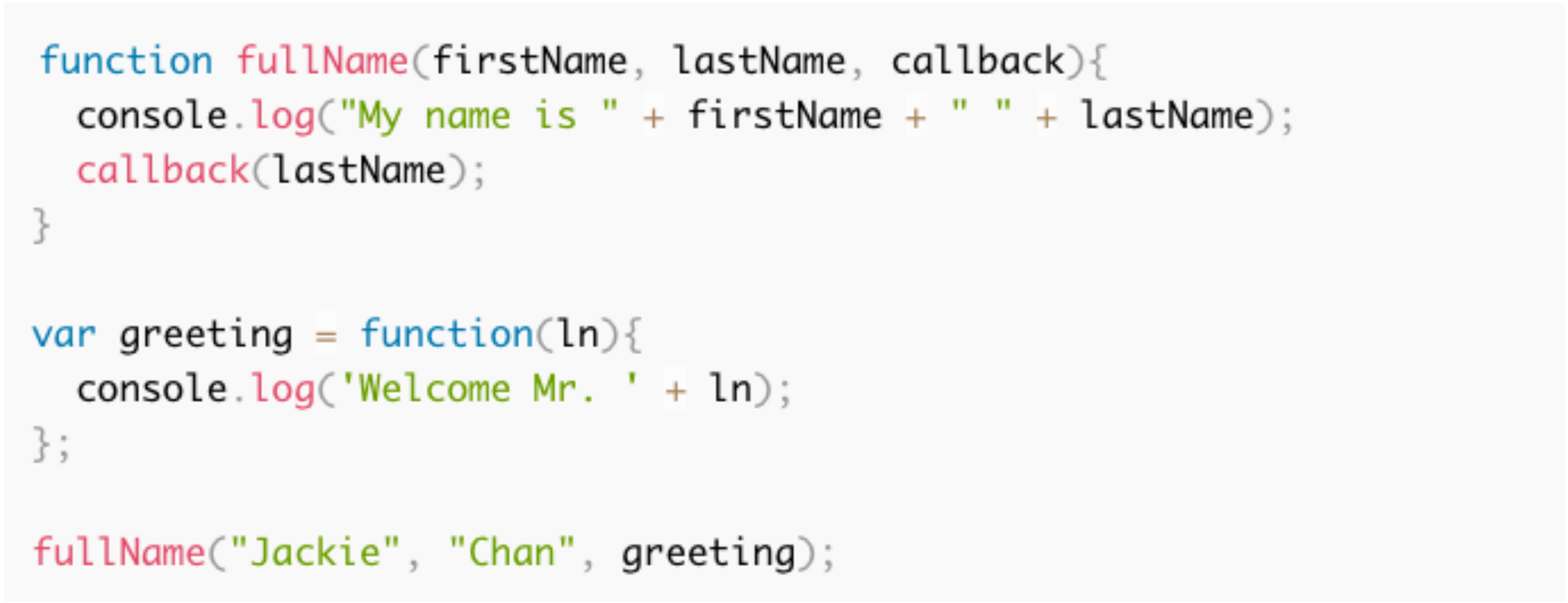
```
var helloTodd = sayHello('Todd');
helloTodd(); // will call the closure and log 'Hello, Todd'
```

CALLBACK

```
function showMessage(message){
  setTimeout(function(){
    alert(message);
  }, 3000);
}

showMessage('Function called 3 seconds ago');
```

CALLBACK



```
function fullName(firstName, lastName, callback){
  console.log("My name is " + firstName + " " + lastName);
  callback(lastName);
}

var greeting = function(ln){
  console.log('Welcome Mr. ' + ln);
};

fullName("Jackie", "Chan", greeting);
```

CALLBACK

```
function fullName(firstName, lastName, callback){
  console.log("My name is " + firstName + " " + lastName);
  callback(lastName);
}

fullName("Jackie", "Chan", function(ln){console.log('Welcome Mr. ' + ln);});
```

IIFE

```
(function () {
  alert(‘Woohoo!');    // transformuoja i expression
})();

var sayWoohoo = function () {
  alert(‘Woohoo!');          // jau yra function expression
}();
```

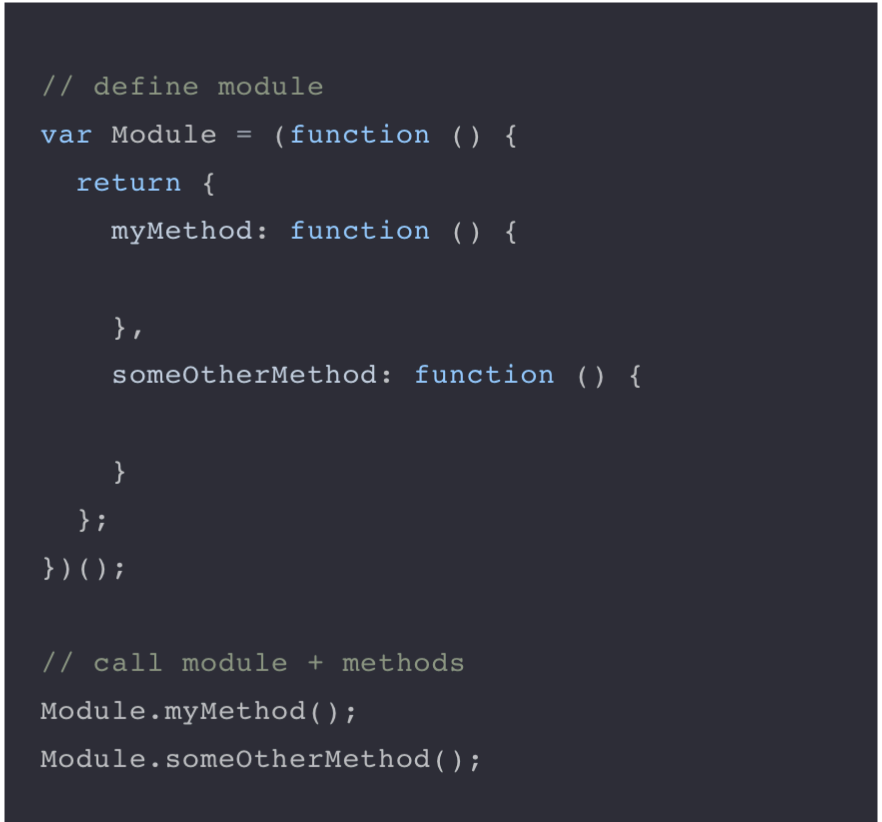
Tai nėra globalios funkcijos

PRIVAČIOS FUNKCIJOS

```
(function () {
  var myFunction = function () {
    // do some stuff here
  };
})();

myFunction(); // Uncaught ReferenceError: myFunction is not defined
```

METODAI IR PUBLIC FUNKCIJOS



```
// define module
var Module = (function () {
  return {
    myMethod: function () {

    },
    someOtherMethod: function () {

    }
  };
})();

// call module + methods
Module.myMethod();
Module.someOtherMethod();
```

```
var Module = (function () {
  var myModule = {};
  var privateMethod = function () {

  };
  myModule.publicMethod = function () {

  };
  myModule.anotherPublicMethod = function () {

  };
  return myModule; // returns the Object with public methods
})();

// usage
Module.publicMethod();
```