

ΣΥΝΑΡΤΗΣΙΑΚΗ JAVASCRIPT

Διδάσκοντες: Π. Αγγελάτος, Δ. Ζήνδρος
Επιμέλεια διαφανειών: Δ. Ζήνδρος

Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών



Στόχος της ώρας

- Συναρτήσεις σε Javascript
- Συναρτήσεις ως τιμές
- Συναρτήσεις ως παράμετροι
- Επιστροφή συναρτήσεων
- Αντικειμενοστραφής προγραμματισμός σε Javascript
- `setTimeout/setInterval`
- Animations

Συνάρτηση

Όνομα συνάρτησης

```
function sayHello() {  
    alert( "Hello" );  
}
```

Σώμα συνάρτησης

```
sayHello();  
sayHello();
```

Κλήση συνάρτησης

Παράμετροι

Παράμετρος συνάρτησης

```
function greet( name ) {  
    var greeting = "Hello, ";  
    greeting += name;  
    alert( greeting );  
}
```

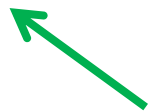
```
greet( "petros" );
```

Τιμή παραμέτρου

Επιστροφή τιμής

```
function Add( a, b ) {  
    return a + b;  
}
```

```
alert( Add( 5, 7 ) );
```



Αντικαθίσταται από 12

Προαιρετικές παράμετροι

- Τα ορίσματα που περνάμε σε μία συνάρτηση Javascript όταν την καλούμε μπορεί να είναι **περισσότερα** απ' όσα ορίζονται.
- `arguments`: Πίνακας που περιέχει τα ορίσματα που έχουν περάσει στη συνάρτηση

```
alert( add( 3, 5, 7 ) );  
alert( add( 10, 20, 7, -7 ) );
```

← Η κλήση μπορεί να γίνει
πριν τον ορισμό

```
function add() {  
    var len = arguments.length;  
    var sum = 0;  
  
    for ( var i = 0; i < len; ++i ) {  
        sum += arguments[ i ];  
    }  
    return sum;  
}
```

↑ Πίνακας παραμέτρων

↑ i-οστή παράμετρος

Τοπικές και καθολικές μεταβλητές

- Οι **καθολικές** μεταβλητές είναι προσβάσιμες από παντού
- Ορίζονται με **var** **εκτός** όλων των συναρτήσεων
- Η Javascript θεωρεί ότι μία μεταβλητή είναι **καθολική** όταν *δεν ορίζεται*
- Οι **τοπικές** μεταβλητές είναι προσβάσιμες μόνο μέσα στη συνάρτηση
- Ορίζονται με **var** **εντός** της συνάρτησης

Τοπικές και καθολικές μεταβλητές

`var a = 5;` ← **Καθολική μεταβλητή**

```
function foo() {  
    var a;      ← Τοπική μεταβλητή  
    a = 6;      ← Δεν επηρεάζει την καθολική μεταβλητή  
    alert( a );  
}
```

Καθολική μεταβλητή

`foo();`
`alert(a);`

Τοπικές και καθολικές μεταβλητές

`var a = 5;` ← Καθολική μεταβλητή

```
function foo() {  
    a = 6;  
    alert( a ); ← Καθολική μεταβλητή αλλάζει  
}
```

```
foo();  
alert( a ); ← Καθολική μεταβλητή
```

Αναδρομή


- Κλήση μίας συνάρτησης μέσα από τον εαυτό της

```
function factorial( n ) {  
    if ( n == 1 ) {  
        return 1;  
    }  
    return factorial( n - 1 ) * n;  
}
```

```
alert( factorial( 6 ) );
```

Ανώνυμες συναρτήσεις

- Η συνάρτηση είναι ένας ακόμη τύπος τιμής
 - Όπως boolean, number, string
- Οι συναρτήσεις δεν είναι απαραίτητο να έχουν όνομα
- Μία ανώνυμη συνάρτηση που προσθέτει:

Παράλειψη ονόματος

function(a, b) {
 return a + b;
}

Κλήση ανώνυμων συναρτήσεων

```
alert(  
    ( function ( a, b ) {  
        return a + b;  
    } )( 5, 7 )  
);
```

Τα () ορίζουν κλήση

Συναρτήσεις ως τιμές

```
var getThree = function () {  
    return 3;  
};  
var threeGetter = getThree; // αντιγραφή συνάρτησης  
var three = getThree(); // κλήση συνάρτησης  
  
alert( getThree );  
alert( threeGetter );  
alert( three );
```

Συναρτήσεις ως παράμετροι

- `$.get()`, `$.post()`, `click()`, `submit()` παίρνουν ως παράμετρο μία **συνάρτηση**

```
var foo = function () {  
    alert( "Clicked!" );  
    return true;  
};
```

```
$( "a" ).click( foo );
```



Περνάει η συνάρτηση

Συναρτήσεις ως παράμετροι

- Το `()` χρησιμοποιείται για την κλήση μίας συνάρτησης:

```
var foo = function () {  
    alert( "Clicked!" );  
    return true;  
};
```

```
$( "a" ).click( foo() );
```



Περνάει true

setTimeout

- Ορίζει ότι μία συνάρτηση θα τρέξει **ασύγχρονα** μετά από ένα χρονικό διάστημα:

```
setTimeout( συνάρτηση, χρόνος_milliseconds );
```

```
var sayHello = function() {  
    alert( "Hello, world!" );  
};
```

```
// τρέχει 1 φορά μετά από 5s  
setTimeout( sayHello, 5000 );
```

setInterval

- Ορίζει ότι μία συνάρτηση θα τρέχει **ασύγχρονα** κάθε ένα ορισμένο χρονικό διάστημα:

```
setInterval( συνάρτηση, χρόνος_milliseconds );
```

```
var sayHello = function() {  
    alert( "Hello, world!" );  
}
```

```
// τρέχει κάθε 5s
```


```
setInterval( sayHello, 5000 );
```


Συναρτήσεις ως παράμετροι

- Μπορούμε να φτιάξουμε δικές μας συναρτήσεις που παίρνουν συναρτήσεις ως παραμέτρους

```
var conditionalCall = function ( condition, f, g ) {  
    if ( condition ) {  
        f();  
    }  
    else {  
        g();  
    }  
};  
conditionalCall(  
    // παράμετρος παίρνει τιμή true ή false  
    x == 5,  
    // παράμετρος-συνάρτηση  
    function () { alert( "x is five" ); },  
    // παράμετρος-συνάρτηση  
    function () { alert( "x is not five" ); }  
);
```

```
var map = function ( data, transformer ) {  
    var result = [];  
    for ( var i = 0; i < data.length; ++i ) {  
        result.push( transformer( data[ i ] ) );  
    }  
    return result;  
};
```

 **Παράμετρος-συνάρτηση
που κάνει το μετασχηματισμό**

 **Εφαρμογή μετασχηματισμού
σε ένα στοιχείο**

```
// περνάει κάθε στοιχείο του data από την transformer
```

```
var cube = function ( x ) {  
    return x * x * x;  
};
```

```
// εμφανίζει [ 1, 8, 27 ]  
alert( map( [ 1, 2, 3 ], cube ) );
```

```
var filter = function ( data, decisionMachine ) {  
    var result = [];  
    for ( var i = 0; i < data.length; ++i ) {  
        if ( decisionMachine( data[ i ] ) ) {  
            result.push( data[ i ] );  
        }  
    }  
    return result;  
};
```

// επιλέγει ποια στοιχεία θα κρατήσει με την decisionMachine

```
var isOdd = function ( x ) {  
    return x % 2 == 1;  
};
```

```
alert( filter( [ 1, 2, 3, 4, 5, 6, 7 ], isOdd ) );
```

```
var reduce = function ( data, soFar, aggregator ) {  
    if ( data.length == 0 ) {  
        return soFar;  
    }  
    var first = data.shift();  
    soFar = aggregator( soFar, first );  
    return reduce( data, soFar, aggregator );  
};
```

Παράμετρος-συνάρτηση που κάνει την «συνένωση»

«συνένωση» δύο στοιχείων

```
alert( reduce( [ 1, 2, 3, 4, 5 ], 0, function ( x, y ) {  
    return x + y;  
} ) );
```

Τιμές που θα «συνενωθούν»

Αρχική τιμή

- Κάθε στοιχείο του πίνακα data προστίθεται στην τιμή soFar.
- Η soFar αρχικά είναι 0
- Συνεχίζει μέχρι να τελειώσουν τα στοιχεία
- Υπολογίζει το **άθροισμα** όλων των στοιχείων

soFar	data
0	[1, 2, 3, 4, 5]
1	[2, 3, 4, 5]
3	[3, 4, 5]
6	[4, 5]
10	[5]
15	[]

```
var reduce = function ( data, soFar, aggregator ) {  
    if ( data.length == 0 ) {  
        return soFar;  
    }  
    var first = data.shift();  
    soFar = aggregator( soFar, first );  
    return reduce( data, soFar, aggregator );  
};
```

```
alert( reduce( [ 'Hello', ',', 'world', '!' ], '',  
function ( x, y ) {  
    return x + y;  
} ) );
```


soFar	data
“	[‘Hello’, ‘,’, ‘world’, ‘!’]
‘Hello’	[‘,’, ‘world’, ‘!’]
‘Hello, ‘	[‘world’, ‘!’]
‘Hello, world’	[‘!’]
‘Hello, world!’	[]

```
alert(  
    reduce(  
        [  
            [ 1, 2, 3 ],  
            [ 4, 5, 6 ],  
            [ 7, 8, 9 ]  
        ],  
        [],  
        function ( x, y ) {  
            return x.concat( y );  
        }  
    )  
);
```

soFar	data
[]	[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
[1, 2, 3]	[[4, 5, 6], [7, 8, 9]]
[1, 2, 3, 4, 5, 6]	[[7, 8, 9]]
[1, 2, 3, 4, 5, 6, 7, 8, 9]	[]

Συναρτήσεις ως τιμή επιστροφής

- Οι συναρτήσεις είναι ένας ακόμη απλός τύπος
- Άρα μπορούν να επιστρέφονται από συναρτήσεις

```
function makeFunction() {  
    var displayHello = function () {  
        alert( "Hello, world!" );  
    };  
    return displayHello;  
}
```

```
var hello = makeFunction();  
hello();
```

```
var makeAdder = function () {  
    var adder = function ( x, y ) {  
        return x + y;  
    };  
    return adder;  
};
```

```
var addTwoValues = makeAdder();  
alert( addTwoValues );  
alert( addTwoValues( 5, 7 ) );
```

```
var makeMultiplier = function ( factor ) {  
    var multiplier = function ( x ) {  
        return factor * x;  
    };  
    return multiplier;  
};
```

```
var double = makeMultiplier( 2 );  
var triple = makeMultiplier( 3 );  
var quadruple = makeMultiplier( 4 );
```

```
alert( double );  
alert( double( 5 ) );  
alert( triple( 4 ) );
```

Κλεισίματα

- Οι συναρτήσεις που ορίζονται μέσα σε άλλες συναρτήσεις έχουν πρόσβαση στις μεταβλητές των εξωτερικών συναρτήσεων
- Αυτό ονομάζεται **κλείσιμο**
- Οι αναφορές αφορούν: Τα **στιγμιότυπα** των εξωτερικών μεταβλητών **τη στιγμή δημιουργίας** της εσωτερικής συνάρτησης

```
var makeMultiplier = function ( factor ) {  
    var multiplier = function ( x ) {  
        return factor * x;  
    };  
    return multiplier;  
};
```



```
var makeCounter = function () {  
    var count = 0;  
  
    return function () {  
        count++;  
        return count;  
    };  
};  
  
var countCows = makeCounter();  
var countChicken = makeCounter();  
alert( countCows() ); alert( countCows() );  
alert( countChicken() ); alert( countChicken() );  
alert( countChicken() );  
alert( countCows() );
```

Javascript

- Προστακτική γλώσσα (όπως C, Pascal)
- Αντικειμενοστραφής (όπως C++, Java)
- Συναρτησιακή (όπως Haskell, LISP, ML)

Αντικείμενα σε Javascript

- Αντικείμενο = Λεξικό
- Περιέχουν ιδιότητες και μεθόδους
- Βολεύουν για να έχουμε μία ομάδα από συναρτήσεις που αφορούν ένα θέμα.

Ένα κενό αντικείμενο

```
var myObject = { };
```

Προσθήκη ιδιοτήτων/μεθόδων

- Μπορούμε να **προσθέσουμε** ιδιότητες και μεθόδους **μετά** την δημιουργία ενός αντικειμένου
- Ακόμη και σε αντικείμενα που δεν δημιουργήσαμε εμείς

```
var links =  
document.getElementsByTagName( 'a' );  
var link = links[ 0 ];
```

```
link.coolnessFactor = "maximum";  
alert( link.coolnessFactor );
```

Αντικείμενο με ιδιότητες

```
var dionyziz = {  
    name: "Dionysis Zindros",  
    age: 107,  
    sex: "yes, please",  
    twitter: "http://twitter.com/dionyziz",  
    site: "dionyziz.com"  
};  
  
alert( dionyziz.age ); // εμφανίζει 107  
alert( dionyziz.site ); // εμφανίζει dionyziz.com
```

Αντικείμενο με μεθόδους

```
var dog = {  
    name: "puppy",  
    bark: function () {  
        alert( "Arf!" );  
    }  
};  
  
dog.bark(); // εμφανίζει Arf!
```

Η λέξη κλειδί “this”

- Αναφέρεται στο αντικείμενο στο οποίο «**ανήκει**» η μέθοδος που καλέσαμε

```
var parrot = {  
    name: "John",  
    sayName: function () {  
        alert( "My name is " + this.name );  
    }  
};
```

```
parrot.sayName(); // Εμφανίζει My name is John
```


Η λέξη κλειδί “this”


Συναρτήσεις που ορίζονται εκτός αντικειμένων «ανήκουν» στο window:

```
function sayHello() {  
    alert( 'Hello, world!' );  
    alert( this ); // εμφανίζει window  
}
```

Πώς πυροδοτεί ο browser τα events

```
var link = document.getElementsByTagName( 'a' )[ 0 ];
```

```
link.onclick();
```

 this = link

```
link.onclick = function () {  
    alert( "Let me take you to " + this.href );  
};
```

```
function navigateAway() {  
    alert( "Let me take you to " + this.href );  
};
```

```
link.onclick = navigateAway;
```



μετά την αντιγραφή:
this = link



πριν την αντιγραφή:
this = window

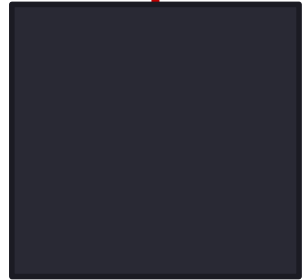
```
<a href="navigateAway()">Test</a>
```



To this **μέσα** στην
navigateAway θα είναι window

Animations

- Μία σταδιακή αλλαγή στην εμφάνιση ενός αντικειμένου
 - Στη θέση
 - Στο χρώμα
 - Στο μέγεθος
 - κλπ.



```
div {  
  background-color: #222;  
  border: 2px solid black;  
  width: 100px;  
  height: 100px;  
  position: absolute;  
  top: 10px;  
  left: 10px;  
}
```

```
<div></div>
```

Πώς θα πετύχουμε κίνηση;

```
var position = 0;  
var div = document.getElementsByTagName( 'div' )[ 0  
];  
  
while ( position < 100 ) {  
    position += 1;  
    div.style.top = position + 'px';  
}
```

Δεν είναι σταδιακή! Το κουτί μεταφέρεται ξαφνικά από το 0 στο 100!

Χρησιμοποιούμε `setInterval/setTimeout`

```
var position = 0;
var div = document.getElementsByTagName( 'div' )[ 0
];

setInterval( function () {
    position += 1;
    div.style.top = position + 'px';
}, 50 );
```



```
var position = 0;  
var div = document.getElementsByTagName( 'div' )[ 0  
];
```

```
function animate() {  
    position += 1;  
    div.style.top = position + 'px';  
    if ( position < 100 ) {  
        setTimeout( animate, 50 );  
    }  
}
```

```
setTimeout( animate, 50 );
```

Μάθαμε

- Συναρτήσεις σε Javascript
- Συναρτήσεις ως τιμές
- Συναρτήσεις ως παράμετροι
- Επιστροφή συναρτήσεων
- Αντικειμενοστραφής προγραμματισμός σε Javascript
- setTimeout/setInterval
- Animations

Συγχαρητήρια!

- Μπορείτε πλέον να χρησιμοποιείτε τις συναρτησιακές και αντικειμενοστραφείς δυνατότητες της Javascript!



Την επόμενη φορά...

- Χρήση SVN για έλεγχο εκδόσεων
- Ανταλλαγή κώδικα σε ομάδες