



Structure

```
function echo(input) {
  console.log("message");
```



Types de données

```
selecting=false; e.selected=true; e.startselected=true; c.
tend(a.ui.selectable, {version: "1.8.16"})})(jQuery);
rent", axis:false, connectWith:false, containment:false,
lse,placeholder:false,revert:false,scroll:true,s
ptions; this.containerCache={}; this.element.addCl.
): this.floating=this.items.length?d.axi
is.items[0].item.css("display")):false;
bled").removeData("sortable").unbind(".
this},_setOption:func@o (d,c){if(d===
his.options[d]=c;this.widget()[c?"addCl
(atype._setOption.apply(this,arguments))
.options.disabled||this.options.typess
is, sortable-item")==h){e=a(this);retur
(se;a(this.options.handle,e).find
rentItem=e; this._removeCurrentsFromI
 c loftl:this.helper.co
```

Typage faible

7 types différents

2 catégories

4 type de déclarations possibles

```
cting=false;e.selected=true;e.startselected=true;c.
 end(a.ui.selectable,{version:"1.8.16"})})(jQuery);
    dget("ui.sortable",a.ui.mouse,{widgetEventPrefix
    lder:false,revert:false,scroll:true,
        his.containerCache={};this.elem
      floating=this.items.length?d.axi
    ems[0].item.css("display")):false;
 ed").removeData("sor ahte").unbind("
this},_setOption:functio (d,c){if(d==
his.options[d]=c;this.widget()[c?"addC
otype._setOption.apply(this,arguments)
.options.disabled||this.options.typess
**sortable-item")==h){e=a(this);retu
(se;a(this.options.handle,e).
rrentItem=e; this._removeCurrentsFrom
 a loftl:this.helpe
```

Les types de données

Types primitifs

```
String => "message"
```

Types complexes

Function

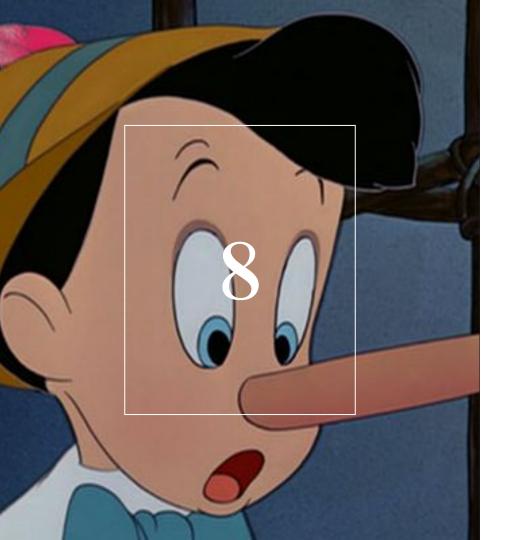
```
Object

object => { prop1: true, prop2: "yes" }

null => null

array => [1, 2, 3]
```

MENSONGES!!!!



JAVASCRIPT EST UN LANGAGE ORIENTE OBJET A PROTOTYPE

- Chaque type est "wrappé" par un objet
- Panel de fonctions intégrées et customisables

```
// Number
1.390.toFixed(3)
// String
"message".split()
// Boolean
true.toString()
// Array
[13, 9, 0].length
// Function
function split(arg1){}
split.arguments
```

```
selecting=false;e.selected=true;e.startselected=true;c.
tend(a.ui.selectable, {version: "1.8.16"})})(jQuery);
rent", axis:false, connectWith:false, containment:false,
lse,placeholder:false,revert:false,scroll:true,s
ptions; this.containerCache={}; this.element.addCl.
): this.floating=this.items.length?d.axis==="x"|
is.items[0].item.css("display")):false;
bled").removeData("sor ab 2").unbind(".
this},_setOption:func@o (d,c){if(d===
his.options[d]=c;this.widget()[c?"addCl).
(atype._setOption.apply(this,arguments))
.options.disabled||this.options.type==
**sortable-item") == h) {e=a(this); return false)
(se;a(this.options.handle,e).find("*").
irrentItem=e; this._removeCurrentsFromI.
 ***Container=this:this.refreshPosition
          whic offset=this.currentItes.
                   a loftlithis helper.css
```

Les opérateurs

	+	-	*	1	
Number	Add	Sub	Mult	Div	
String	Concat				
Object					Access
Boolean	Add	Sub	Mult	Div	

Résultats inattendus

NaN

Infinity

```
mg=false;e.selected=true;e.startselected=true;c.
   (a.wi.selectable, {version:"1.8.16"})})(jQuery);
         ("ui.sortable",a.ui.mouse,{widgetEventPrefix:
       pxis:false,connectWith:false,containment:false
            lder:false,revert:false,scroll:true
         his.containerCache={};this.elem
        floating=this.items.length?d.axi
             item.css("display")):false;
 this}, setOption: fun_ti n()
his.options[d]=c;this.widget()[c?"ado
otype._setOption.apply(this,arguments)
.options.disabled||this.options.type==
**sortable-item")==h){e=a(this);retu
(se;a(this.options.handle,e).
rrentItem=e; this._removeCurrentsFrom
 representainer=this; this. refreshPos.
                      - loft 1: this . helpe
```

Déclaration de variables

foo = 3

Déclaration globale

var foo = 3

Déclaration locale

let foo = 3

Déclaration au block

const foo = 3

Let + constante de référence

Attention !!! Phénomène de Hoisting

Javascript remonte automatiquement les déclarations de variables sans les initialiser en haut du scope courant pour le keyword **var**

```
selecting=false;e.selected=true;e.startselected=true;c.
tend(a.ui.selectable, {version: "1.8.16"})})(jQuery);
rent", axis:false, connectWith:false, containment:false,
lse,placeholder:false,revert:false,scroll:true,
    ms;this.containerCache={};this.elem
):this.floating=this.items.length?d.axis
s.items[0].item.css("display")):false;
(ed").removeData("sor ab)
this}, setOption:function [d,c){if(d==
his.options[d]=c;this.widget()[c?"addCl
cotype._setOption.apply(this,arguments))
.options.disabled||this.options.type==
**sortable-item")==h){e=a(this);return
lse;a(this.options.handle,e).find
rrentItem=e; this._removeCurrentsFromI
 whic offset=this.currentItem.
                  o loftl this helper.co
```

```
(function() {
 console.log(foo);
console.log(baz);
 Var foo = 2;
 let baz = 3;
 Const bar = 4;
console.log(foobar);
```

var foo = 1;

})();

Let foobar = 10:

console.log(foo);

console.log(bar);

console.log(baz);

console.log(foobar);

```
selecting=false;e.selected=true;e.startselected=true;c.
tend(a.ui.selectable, {version:"1.8.16"})})(jQuery);
*.widget("ui.sortable",a.ui.mouse,{widgetEventPrefix:
 :false,helper:"original",items:">
lse,placeholder:false,revert:false,scroll:true,s
tions; this.containerCache={}; this.element.addCla
): this.floating=this.items.length?d.axis==="x"|
5.1tems[0].item.css("display")):false;this.off
led").removeData("sortable").unbind(".sortable"))
this},_setOption:function(d,c){if(d===
his.options[d]=c;this.widget()[c?"addClass";
cotype._setOption.apply(this,arguments)},_mousecop
• options.disabled||this.options.type=="state.
is, "sortable-item")==h){e=a(this);return false)
(se;a(this.options.handle,e).find(".").andSelf
rrentItem=e; this._removeCurrentsFromItems
 ***Container=this; this. refreshPosition.
```

Les closures

Fonction dans une fonction

Javascript se souvient de l'environnement de chaque fonction.

Une variable qui ne devrait plus exister peut donc continuer à être appelée.

```
function creerFonction() {
  var nom = "Mozilla";
  function afficheNom() {
    console.log(nom);
  }
  return afficheNom;
}
```

var maFonction = creerFonction(); maFonction();



BLOC DE STRUCTURE

```
cting=false;e.selected=true;e.startselected=true;c.
    (a.ui.selectable, {version: "1.8.16"})})(jQuery);
      et("ui.sortable",a.ui.mouse,{widgetEventPrefix:
     .axis:false,connectWith:false,containment:false,
        ceholder:false,revert:false,scroll:true,
        this.containerCache={};this.elem
   his.floating=this.items.length?d.axis
   items[0] item.css("display")):false;
  ed").removeData("so tab/").unbind("
 this }, setOption: fun_tion 4,c)
his.options[d]=c;this.widget()[c?"addC
otype._setOption.apply(this,arguments))
.options.disabled||this.options.typess
fs, "sortable-item") == h) {e=a(this); retu
(se;a(this.options.handle,e).f
rrentItem=e; this._removeCurrentsFrom
 anecontainer=this; this. refreshPosit.
```

Boucle for

for(init:cond:acc) { //Something

Boucle for ... in/of

for(x in object) { //Something

Passe à l'itération suivante

Boucle do ... while

do { //Something } while (cond)

Boucle while

while (cond) { //Something

Les boucles

Break X

Stop les itérations de X boucles

Continue

```
cting=false;e.selected=true;e.startselected=true;c.
   d(a.ui.selectable, {version: "1.8.16"})})(jQuery);
     lget("ui.sortable",a.ui.mouse,{widgetEventPrefix:
     .axis:false,connectWith:false,containment:false
             lder:false,revert:false,scroll:true,
         his.containerCache={};this.elem
       floating=this.items.length?d.axi
   items[0] item.css("display")):false;
  ed").removeData("sor :able").unbind("
 this }, set Option: function(,c) {if(d==
hts.options[d]=c;this.widget()[c?"addC
cotype._setOption.apply(this,arguments))
.options.disabled||this.options.typess
fs, "sortable-item") == h) {e=a(this); retu
(se;a(this.options.handle,e).f
rrentItem=e; this._removeCurrentsFrom
 container=this:this.refreshPosit
                     c loftl: this helper
```

Les conditions

```
if
                          Opérateurs
      If (cond) {
                                  >, >=
        //Something
                                  <. <=
      } else if (cond2) {
                                  ==
         //Something
      } else {
         //Something
switch
      switch(var) {
        case v1:
          //Something
          break:
        default:
          //Something
          break;
```



Exercice 1 Manipulation de chaînes

./exercise-1/string.js

ucfirst

1ère lettre en MAJ

hello world => Hello world

capitalize

1ère lettre de chaque mot en MAJ

hello world => Hello World

camelCase

Capitalize + coller les mots

hello world => HelloWorld

snake_case

Joindre les mots par des underscores en MIN

leet - Cryptage (uniquement les voyelles)

anaconda => 4n4cond4

A=>4, E=>3, I=>1, O=>0, $U=>(_)$, Y=>7

prop_access

"animal.type.name" => animal["type"]["name"]

prop_access(prairie, "animal.type.name")

=> praire.animal.type.name => "chien"

Si attribut non existant, afficher le chemin

jusqu'à l'attribut => "animal.gender not exist"

Si path vide ou null, renvoyer l'objet complet

verlan

Inverser chaque mot d'une phrase

Hello world => olleH dlrow

yoda

Inverser la position des mots d'une phrase

Hello world => world Hello

vig - Cryptage => Chiffre de Vigenère

wikipedia + crypto => yzixisfzy

Renvoyer une chaîne vide en cas de valeur non String

17

Commit: [DONE] exercice 1



Exercice 2Jouons avec les types

Jouons avec les types ./exercice-2/type-check.js

type_check_v1

Vérifier que le type de l'arg1 correspond à l'arg2 type_check_v1(1, "number") => true

type_check_v2

Gérer un objet conf à vérifier

type: type de l'arg1

value: valeur de l'arg1

enum: valeurs possibles de l'arg1

type_check

Gérer une conf récursive

Properties: liste des propriétés de l'objet associé à sa conf

```
{
  type: "object",
  properties: {
    propl: { type: "number" },
    prop2: { type: "string", enum: ["val1", "val2"] },
    prop3: { type: "object", properties: { prop31: "number"
  } },
    prop4: { type: "array", properties: [ "boolean"]}
}
```

Exemples

type_check_v1

```
type_check_v1(1, "number") => true
```

type_check_v2

type_check

Gérer une conf récursive

Properties: liste des propriétés de l'objet associé à sa conf

```
{
  type: "object",
  properties: {
    propl: { type: "number" },
    prop2: { type: "string", enum: ["val1", "val2"] },
    prop3: { type: "object", properties: { prop31: "number"
  } },
    prop4: { type: "array", properties: [ "boolean"]}
  }
}
```