Cahier d'exercices JavaScript
PSB
Année 2020
V1.0

A/ Exercice de manipulation de base avec le langage

1/

```
test='valeur'
test+23
a='23"
b=16
a-b
a+b

console.log(1 + "2" + "2");
console.log(1 + "2" + "2");
console.log(1 +-"1" + "2");
console.log(+"1"+ "1" + "2");
console.log( "A"- "B" + "2");
console.log( "A"- "B" + 2);

012+0123
95+025
```

```
Chaîne de caractères :
var arr1="john".split(') ;
var arr2=arr1.reverse() ;
var arr3="jones3".split(') ;

arr2.push(arr3) ;

arr1 ?    arr2 ?   pourquoi ?
```

Conversions :

```
paseInt('12')
parseInt('12',8)
parseInt('12',16
parseInt('111',2)
parseInt('blabla')

parseFloat('12,32)
Affichage 1 chiffre derrière la virgule
```

2/ Tableaux :

Manipulation au niveau des tableaux

var monTableau=[12,51,10,9,8,7,6] ;
Veuillez modifier la position 2 à 5 avec la valeur 99 ?
Veuillez trouver l'instruction pour trier dans l'ordre croissant et décroissant ce tableau.
On souhaite le copier dans une autre variable monTableau2 ?

Récupérez une valeur à un indice dans la tableau et au delà ... que renvoie la fonction ?

Veuillez ensuite modifier tous les éléments du tableau est les mettre à -1 ?
On souhaite rajouter une entrée test='PSB'

Veuillez modifier le contenu de ce tableau
Quelle est la longueur de ce tableau ? expliquez pourquoi ?

3/ Fonctions :

3-1/ function initiale()
Veuillez définir une fonction qui reçoit un string et renvoie le même string avec une majuscule pour la première lettre et le reste en minuscules.

3-2 / Cette même fonction on souhaite la définir directement dans le prototype de String.
veuillez faire l'aménagement.

3-3/ On souhaite sur une chaine de caractères modifier les numériques et les mettre entre parenthèses
Veuillez définit la fonction pour le faire ?

Exemple : demain il fait 32 à l'ombre demain il fait (32) à l'ombre

3-4/ Veuillez implémenter les fonctions suivantes

Ecriture d'une fonction de sum avec arguments

Ecrire une fonction qui doit fonctionner dans les cas suivants :
console.log(sum(2,3)); // Outputs 5
console.log(sum(2)(3)); // Outputs 5

```
function sum(x) {
	if (arguments.length == 2) {
		return arguments[0] + arguments[1];
	} else {
		return function(y) { return x + y; };
	}
}
```

Ecriture d'une fonction qui détecte si la chaîne est un palindrome et renvoie true

```
console.log(isPalindrome("level")); // true
console.log(isPalindrome("levels")); // false
console.log(isPalindrome("A car, a man, a maraca")); //true
```

```
function isPalindrome(str) {
	str = str.replace(/\W/g, '').toLowerCase();
	return (str == str.split('').reverse().join(''));
}
```

4/ Prototype et manipulation des classes

Soit l'exercice suivant à réaliser directement sous la console de votre navigateur

```
function Vehicule() {
        this.type='non renseigné' ;
        this.marque='non renseigné' ;
}
```

```
function Voiture() {
        this.type='voiture' ;
        this.modele='non renseigné' ;
        this.moteur='non renseigné' ;
        this.puissance='non renseigné' ;
}
```

```
function Renault() {
        this.marque='Renault' ;
}
```

```
Voiture.prototype=new Vehicule ;
Renault.prototype=new Voiture ;
```

Veuillez saisir les différents éléments dans votre environnement pour constater le comportement qu'offre Javascript en terme d'héritage :

on utilise nos définitions :

```
var maVoiture=new Renault();
var maVoiture2=new Renault();
maVoiture2.kilometrage='124521' ;
```

Quels sont les attributs de maVoiture2 ?

Veuillez ajouter un nouvel attribut à la classe Voiture par exemple nombre de place .

Quels deviennent les attributs de maVoiture2 ? après ce changement ?

Exploration d'une DOM

Visite d'une page Web
Créer une fonction qui reçoit la DOM de la page chargée et qui visitera la page elle-même et tous les descendants et pas seulement les descendants immédiat. Pour chaque élément visité, la fonction devra fournir une fonction callback pour afficher le contenu.

```
function Traverse(p_element,p_callback) {
        p_callback(p_element);
        var list = p_element.children;
        for (var i = 0; i < list.length; i++) {
                Traverse(list[i],p_callback); // recursive call
        }
}
```

Essayez d'utiliser cette fonction sur une page simple avec en paramètre
```
var affiche=function(e) {console.log(e);}
Traverse(document, affiche) ;
```

Manipulation de la DOM d'une page

Veuillez charger la page de l'exercice PSB/Chapitre4 sur votre ordinateur et charger la page html dans votre navigateur ....

Utilisez la fonction traverse pour analyser le contenu de cette page.
Que contiennent les variables
- window
- document

On souhaite modifier l'affichage de la page chargée :

Veuillez mettre au point ce traitement
- Voir cours Page 42
- Voir cours page 43
- Voir cours Page 44

Gestion des événements : Page 51

Lorsqu'on passe avec la souris sur les articles , on souhaite un changement de style et de taille des Articles.

Functional javascript

```
var colors = [’blue’, ’black’, ’red’];
var cars = ...;

A partir de ce code, construire un véhicule par couleur
```

```
var colors = ['blue', 'black', 'red'];
var cars = colors.map(buildCar);

function buildCar (color) {
        return new Car(color);
}
```

Objets :
Utilisation d'une fonction privée .... attention chaque objet créé aura sa propre fonction privée ...

```
var Employee = function (name, company, salary) {
	this.name = name || "";
	//Public attribute default value is null
	this.company = company || ""; //Public attribute default value is null
	this.salary = salary || 5000; //Public attribute default value is null
	// Private method
	var increaseSalary = function () {
		this.salary = this.salary + 1000;
	};
	// Public method
	this.displayIncreasedSalary = function() {
		increaseSalary();
		console.log(this.salary);
	};
};

// Create Employee class object
var emp1 = new Employee("John","Pluto",3000);
// Create Employee class object
var emp2 = new Employee("Merry","Pluto",2000);
// Create Employee class object
var emp3 = new Employee("Ren","Pluto",2500);
```

Les fermetures
Exercice 1/ avec changement de couleur toutes les secondes par article

Charger le fichier exercice_1-4.html et veuillez modifier via la navigateur les paramètres de telle sorte que lors du click sur le bouton un lkes paragraphes les uns après les autres changent de couleur.

Exercice 2/

Elaboration d'un compteur
Veuillez élaborer une fonction qui permet d'offrir la fonctionnalité compteur. A chaque appel de cette fonction on incrémente à 1. Contrainte on ne veut pas de variable globale qui soit incrémentée directement.

Des tentatives ont été développées :

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

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

Résultat et commentaire :

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

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

Résultat et commentaire :

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

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

Résultat et commentaire :

Une solution avec la notion de closure (fermeture) est envisagée ce qui nous donne

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

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

Résultat et commentaire :

Exercice 3

On souhaite créer un formulaire avec trois champs à saisir et une aide adaptée à ces champs
Champs

Aide :
E-Mail :
Nom :
Age :

Veuillez définir le programme qui permet d'effectuer cette saisir

```
<html>
	<head>
	</head>
	<body>
		<p id="aide">Des aides seront affichées ici</p>
		<p>E-mail : <input type="text" id="email" name="email"></p>
		<p>Nom : <input type="text" id="nom" name="nom"></p>
		<p>Âge : <input type="text" id="âge" name="âge"></p>
	<script>
		function afficherAide(aide) {
			document.getElementById('aide').innerHTML = aide;
		}
		function preparerAide() {
			var texteAide = [
				{'id': 'email', 'aide': 'Votre adresse e-mail'},
				{'id': 'nom', 'aide': 'Vos prénom et nom'},
				{'id': 'âge', 'aide': 'Votre âge (plus de 16 ans requis)'}
			];
			for (var i = 0; i < texteAide.length; i++) {
				var item = texteAide[i];
					document.getElementById(item.id).onfocus = function() {
						afficherAide(item.aide);
					}
			}
		}
		preparerAide();
	</script>
	</body>
</html>
```

Testez ce programme et veuillez le corriger pour qu'il fonctionne ... affiche l'aide en fonction du focus sur le champ correspondant :

Corrections :

Tableaux :

```
monTableau.sort((a,b)=>{return Number(a)>Number(b)?-1:Number(a)<Number(b) ?1:0 }) ;
```

Tableau de liste d'objet que l'on souhaite classer

On peut le faire avec une fonction du même genre et passer la fonction en paramètres pour le classement d'objets selon l'attribut handle de l'objet.

```
function compare( a, b ) {
  if ( a.handle < b.handle ){
    return -1;
  }
  if ( a.handle > b.handle ){
    return 1;
  }
  return 0;
}
```

Fonction pour les parenthèses :

```
function numentreparentheses(texte) {
        function encadrer(hit,numgroupe,str){
                return (numgroupe>0 ?'(' :'')+hit+(numgroupe>0 ?')' :'') ;
        }
        return texte.replace(/([0-9]*)/g,encadrer) ;
}
```

Réf exercice : https://developer.mozilla.org/fr/docs/Web/JavaScript/Closures

Exercice sur les fermetures :

```
<html>
	<head>
	</head>
	<body>
		<p id="aide">Des aides seront affichées ici</p>
		<p>E-mail : <input type="text" id="email" name="email"></p>
		<p>Nom : <input type="text" id="nom" name="nom"></p>
		<p>Âge : <input type="text" id="âge" name="âge"></p>
	<script>
	function afficheAide(aide) {
		document.getElementById('aide').innerHTML = aide;
	}
	function creerCallbackAide(aide) {
		return function() {
			afficheAide(aide);
		};
	}
	function prepareAide() {
		var texteAide = [
			{'id': 'email', 'aide': 'Votre adresse e-mail'},
			{'id': 'nom', 'aide': 'Votre prénom et nom'},
			{'id': 'âge', 'aide': 'Your age (you must be over 16)'}
		];
		for (var i = 0; i < texteAide.length; i++) {
			var item = texteAide[i];
			document.getElementById(item.id).onfocus = creerCallbackAide(item.aide);
		}
	}
	prepareAide();
	</script>
	</body>
</html>
```

Manipulation DOM

Exercice de manipulation et de création d'objets dans la page actuelle
On souhaite ajouter 5 boutons dans la page avec un comportement donné (affichage dans la console)
Voici la première version du code

```
for (var i = 0; i < 5; i++) {
        var btn = document.createElement('button');
        btn.appendChild(document.createTextNode('Button ' + i));
        btn.addEventListener('click', function(){ console.log(i); });
        document.body.appendChild(btn);
}
```

Solution proposée ;

```
for (var i = 0; i < 5; i++) {
        var btn = document.createElement('button');
        btn.appendChild(document.createTextNode('Button ' + i));
        btn.addEventListener('click', (function(i) {
                return function() { console.log(i); };
                                })(i));
        document.body.appendChild(btn);
}
```

Expliquez la solution ....

Explications :

Exercice 1 : Web Socket
Élaboration d'un chat en ligne avec Socket.io et Express
Création des répertoires et préparation de l'environnement.

```
$mkdir socket-app
$ cd socket-app
$ npm init
```

Remplir les différents éléments :
nom de l'auteur
fichier app.js
lancement npm start

```
$ npm install socket.io express --save
$ npm install jquery --save
```

Regardez le contenu du fichier packages.json qui définit les dépendances de l'application.

Création du fichier de démarrage de l'application nodejs

```
// app.js
var express = require('express');
var app = express();
var server = require('http').createServer(app);
var io = require('socket.io')(server);

app.use(express.static(__dirname + '/node_modules'));
app.get('/', function(req, res,next) {
    res.sendFile(__dirname + '/index.html');
});

server.listen(4200);
```

```
<!doctype html>
<html lang="en">
   <head>

   </head>
   <body>
      <h1>Hello World!</h1>
      <div id="future"></div>
      <form id="form" id="chat_form">
         <input id="chat_input" type="text">
         <input type="submit" value="Send">
      </form>
       <script src="/jquery/dist/jquery.js"></script>
      <script src="/socket.io/socket.io.js"></script>
   </body>
</html>
```

Pour le moment rien de fantastique ....

Mise en place de l'échange via la web socket

Ajout dans le fichier html

```
io.on('connection', function(client) {
    console.log('Client connected...');

    client.on('join', function(data) {
        console.log(data);
    });
});
```

```
    <script src="/jquery/dist/jquery.js"></script>
<script src="/socket.io/socket.io.js"></script>
<script>
        var socket = io.connect('http://192.168.1.26:4200');
        socket.on('connect', function(data) {
                socket.emit('join', 'Hello World from client');
        });
</script>
```

Les deux parties se connectent via la web socket

Le serveur renvoie un message sur le client

```
io.on('connection', function(client) {
   console.log('Client connected...');

   client.on('join', function(data) {
      console.log(data);
      client.emit('messages', 'Hello from server');
   });
```

```
// index.html
...
        socket.on('messages', function(data) {
              alert(data);
        });
```

Pour finir notre application chat :

```
// app.js
...
io.on('connection', function(client) {
   console.log('Client connected...');

   client.on('join', function(data) {
      console.log(data);
   });

   client.on('messages', function(data) {
         client.emit('broad', data);
         client.broadcast.emit('broad',data);
   });

});
```

```
<!--index.html-->
...
<script>
 var socket = io.connect();
 socket.on('connect', function(data) {
    socket.emit('join', 'Hello World from client');
 });
 socket.on('broad', function(data) {
        $('#future').append(data+ "<br/>");
   });

 $('form').submit(function(e){
     e.preventDefault();
     var message = $('#chat_input').val();
     socket.emit('messages', message);
 });
</script>
```

Vous pouvez tester votre chat .....Simple
Pour un fonctionnement plus évolué voir les exemples de socket.io

Contenu du fichier packages.json

```
{
 "name": "socket-app",
 "version": "1.0.0",
 "description": "Application de demo de chat",
 "main": "app.js",
 "scripts": {
   "start": "node ./app.js"
 },
 "author": "Y.Stroppa",
 "license": "ISC",
 "dependencies": {
   "express": "^4.17.1",
   "jquery": "^3.4.1",
   "socket.io": "^2.3.0"
 }
```

Pour exécuter : npm start

Autres exercices

soit la fonction suivante :

```
function readHugeList(l) {
	var liste=[];
	for (var i =0 ; i<l;i++) {
		liste.push("valeur"+ i) ;
	}
	return liste ;
}
```

soit la fonction suivante :

```
var list=readHugeList() ;
var nextListItem=function() {
	var item=list.pop() ;
	if (item) {
		// process pour traiter les elements item
		console.log(item) ;
		nextListItem() ;
	}
} ;
```

soit la fonction suivante :

```
var list=readHugeList() ;
var nextListItem=function() {
	var item=list.pop() ;
	if (item) {
		// process pour traiter les elements item
		setTimeout(nextListItem(),0) ;
	}
} ;
```