PARCOURS FRONT-END

Javascript

Ateliers ES2015

Notes :

- Pour chacun de ces exercices, créer un fichier index.html et un fichier app.js
- Le fichier app.js contiendra le code de votre application JS
- Créer une balise script dans le fichier index.html faisant référence au fichier app.js
- Pour tester, ouvrir la page index.html dans le navigateur et se servir de l'outil de développement pour l'affichage/debug

Exercice « let/const »

- Sans l'exécuter, indiquer si ce code est valide. Qu'affiche-t-il ? Pourquoi ?

```
for (var i = 0; i < 5; i++) {

   setTimeout(function () {
       console.log(i);
   }, 500);
}
// On voudrait la suite : 0 1 2 3 4
// Ici le script affiche : 5 5 5 5 5
```

Exercice « Object shortcuts »

- Considérer le code suivant :

```
const checklist = getChecklist(
  "Faire les courses",
  "Faire le ménage",
  "Promener le chien"
);

console.log(checklist);

// {
//     'Faire les courses': true,
//     'Faire le ménage': true,
//     'Promener le chien': true
// }
```

- Votre objectif est de coder la fonction **getChecklist** afin d'obtenir le même résultat que dans le commentaire.
- Pour obtenir l'objet résultat, vous ne devez qu'utiliser l'un des raccourcis vus dans le cours.

Exercice « String templates »

- Faire un script qui générera (en utilisant les string templates) la présentation d'une personne en fonction des informations suivantes : nom, prénom, age, ville, profession.

Exercice « Fonctions fléchées »

- Considérer le code suivant :

```
function Prefixer(prefix) {
    this.prefix = prefix;
}

Prefixer.prototype.prefixArray = function (arr) {
    return arr.map(function (x) {
        // Doesn't work:
        return this.prefix + x;
    });
};

let p = new Prefixer("pre-");
console.log(p.prefixArray([1, 2, 3]));
```

- Trouver un moyen de faire fonctionner ce script, l'objectif étant que la méthode prefixArray retourne un tableau, contenant les mêmes éléments que ceux passés dans le tableau en paramètre, mais tous préfixés d'une chaîne de caractère (« pre- » dans l'exemple).
- Ainsi, s'il fonctionne, ce code est censé renvoyer : ["pre-1", "pre-2", "pre-3"].
- **Astuce :** Il faut utiliser les fonctions fléchées… Comment ça c'est pas une astuce ..? ☹

Exercice « POO »

- Écrire une classe Animal avec des propriétés. Puis créer deux classes Chien et Chat, qui héritent de Animal et qui ayant chacune au moins une propriété spécifique.
- Profiter de cet exercice pour tester tout ce que vous venez de voir.

Exercice « Fetch »

- Récupérer le dernier lancement de fusée de SpaceX (API : https://api.spacexdata.com/v4/launches/latest) et afficher le nom de la roquette utilisée pour ce lancement.
- Utiliser bien évidemment les promesses (et fetch est de rigueur ! Désolé pour les férus de XMLHttpRequest) !

Exercice « Async/Await »

- Reprendre l'exercice précédent en mode async/await !

Exercice « Module »

- Reprendre l'exercice de la POO et séparer les classes en différents fichiers.
- Utiliser les ES modules et organiser les imports/exports de manière à faire fonctionner les tout.
- Astuce : Utiliser le standard des ES module (attribut type="module" sur balise script) ! Demander à l'intervenant pour leur utilisation !

Exercice « Bookstore »

- Reprendre l'exercice du Bookstore et l'améliorer en utilisant les nouvelles fonctionnalités vues sur ES2015.
- De plus, à l'initialisation, le bookstore devra afficher une liste de livres dont les informations sont présentes à l'adresse suivante (au format json) : https://gist.githubusercontent.com/spirival/55b20e602aad32f3fe697074d82665a4/raw/0540d907dbc21fe6322540953529f623b07f10d0/bookstore.json