T1b - Code JavaScript

Script de calcul de moyennes

Un enseignant qui s'intéresse à **JavaScript** cherche à utiliser ce langage pour calculer les moyennes de ses élèves et les afficher.

Création d'un nouveau projet

- Dans **VS Code**, utilisez le menu : **File** > **O pen Folder...** ou **Ctrl**+**K** **Ctrl**+**O**
- Naviguez jusqu'à votre **dossier de travail global** (celui qui contient déjà le projet **t1-bonjour**).
- Créez-y un nouveau dossier nommé **t1-moyennes**
- Sélectionnez-le pour l'ouvrir dans **VS Code**.

Script initial

L'idée de l'enseignant est de créer un tableau `eleves` qui contient autant d'items qu'il y a d'élèves dans la classe.

Chaque item sera un objet contenant le prénom d'un élève et la liste de ses notes.

- Créez un nouveau fichier nommé `groupe1.js`
- En tête, mettez la directive qui demande d'utiliser le mode strict.
  `'use strict'`
- Créez le tableau `eleves` contenant les élèves du groupe 1 :

```
const eleves = [
  { prenom: 'Aline', notes: [15, 13, 11] },
  { prenom: 'Brice', notes: [13, 14, 18] },
  { prenom: 'Cathy', notes: [11, 10, 6] },
  { prenom: 'Diego', notes: [10, 10, 13] },
]
```

- Écrivez l'instruction qui affiche le tableau `eleves`
- Exécutez le script pour vérifier son fonctionnement.

Utilisation d'une fonction constructeur

Ce code paraît un peu lourd à l'enseignant qui a l'habitude du **Java**. Il a lu quelque part qu'il n'y avait pas de classes en **JavaScript**, mais qu'on pouvait définir une fonction constructeur.

Licence Creative Commons BY-SA 4.0 - AMBLARD Emmanuel 2026

Voici la fonction correspondant aux objets du tableau **eleves** :

```
function Eleve(prenom, notes) {
  this.prenom = prenom
  this.notes = notes
}
```

Le mot clé **new** permet de l'utiliser comme constructeur :

```
new Eleve('Aline', [15, 13, 11])
```

L'usage est de mettre une majuscule au nom de la fonction afin de signaler qu'il s'agit d'un constructeur. Elle est considérée comme étant le type des objets qui sont créés grâce à elle.

- Ajoutez le code de la fonction **Eleve()** au début du script.
- Transformez le code qui définit le tableau **eleves** afin que la fonction **Eleve()** soit utilisée pour instancier les objets.

  Astuce On peut utiliser le multi-curseur pour gagner du temps :
  - Dans le tableau **eleves**, ajoutez un espace devant la note **6** de **Cathy**.
    Ainsi les 4 lignes seront parfaitement alignées verticalement.
  - Placez le curseur devant l'accolade ouvrante de la première ligne.
  - Appuyez 3 fois sur les touches **Ctrl**+**Alt**+↓
    Cela crée un curseur multi-ligne.
  - Effacez : `{ prenom :`
    et à la place, tapez : `new Eleve(`
  - Avec la touche → déplacez le curseur pour pouvoir effacer `notes :`
  - Déplacez de nouveau le curseur et remplacez `}` par `)`
  - Appuyez sur **Echap** pour quitter le mode multi-curseur
- Exécutez le script.

L'affichage produit est similaire au précédent, mais il indique en plus que les objets sont du type **Eleve**, c'est-à-dire que c'est la fonction **Eleve()** qui a été utilisée comme constructeur.

Cela ne garantit absolument pas que ces objets ont tous la même structure, car il est toujours possible d'ajouter ou de supprimer des propriétés à chacun d'eux, individuellement.

L'enseignant est satisfait, car il trouve que le code est plus concis et plus lisible.

Calcul de la moyenne

Boucle for…of

Pour parcourir un tableau on peut écrire une boucle avec la syntaxe **for...of** :

```
for (const item of tableau) {
  console.log(item)
}
```

Licence Creative Commons BY-SA 4.0 - AMBLARD Emmanuel 2026

Remarquez le mot-clé `const` devant `item`. On définit une nouvelle variable `item` à chaque itération.

- En suivant le modèle ci-dessus, écrivez le code qui affiche la liste des prénoms des élèves. Le résultat attendu est :

```
Aline
Brice
Cathy
Diego
```

- Puis, écrivez le code qui affiche les notes d'**Aline**. Le résultat attendu est :

```
15
13
11
```

Fonction getMoyenne()

- Écrivez une fonction qui calcule la moyenne d'un tableau de notes reçu en paramètre.

  Le résultat va ressembler à ceci :

```
function getMoyenne(notes) {

  ... ...

  return somme / notes.length
}
```

  Vous devez utiliser une boucle `for…of` pour calculer la `somme` des notes.

- Ensuite, avant l'instruction qui affiche le tableau `eleves`, écrivez le code qui ajoute à chaque objet `Eleve` de ce tableau, une propriété `moyenne` contenant le résultat du calcul.

- L'affichage produit doit être :

```
[
  Eleve { prenom: 'Aline', notes: [ 15, 13, 11 ], moyenne: 13 },
  Eleve { prenom: 'Brice', notes: [ 13, 14, 18 ], moyenne: 15 },
  Eleve { prenom: 'Cathy', notes: [ 11, 10, 6 ], moyenne: 9 },
  Eleve { prenom: 'Diego', notes: [ 10, 10, 13 ], moyenne: 11 },
]
```

Vous constatez qu'il a bien été possible d'ajouter une nouvelle propriété à chaque objet et que celui-ci est toujours défini comme étant de type `Eleve`. Ce serait pareil si on supprimait toutes les propriétés pour en faire un objet vide. Il serait toujours du type `Eleve`.

En fait, le type ne définit pas les propriétés, ni les méthodes propres à un objet, mais celles dont il hérite via l'héritage prototypique. (**NB.** Ici nous ne nous servons pas de ces concepts. Ils sont évoqués juste pour information)

Licence Creative Commons BY-SA 4.0 - AMBLARD Emmanuel 2026

Affichage final

- À la fin du script, écrivez le code qui affiche, pour chaque élève, son prénom suivi de sa moyenne. Le résultat attendu est :

```
Aline 13
Brice 15
Cathy 9
Diego 11
```

Il faut écrire une boucle `for…of` et utiliser le fait que **console.log()** peut accepter plusieurs paramètres.

Licence Creative Commons BY-SA 4.0 - AMBLARD Emmanuel 2026