CC JS : Session 1 - Durée 1h30 - Sur machine

Téléchargez l'archive **cc-22-js-sujet.tgz** de l'espace Moodle. Après extraction, renommez le répertoire obtenu **nom-prenom** où **nom** et **prenom** sont vos nom et prénom écrits en minuscules et sans espaces (par ex. **delafontaine-jean**). A l'issue de l'examen, archivez ce répertoire (`tar czf nom-prenom.tgz nom-prenom`) et déposez l'archive sur Moodle.

Le sujet comporte 4 exercices indépendants. Pour visualiser ce qui est attendu, un démonstrateur est à votre disposition : https://leria-info.univ-angers.fr/~a.jamin/l2_dw/cc-22-js/.

Exercice 1. Manipulation de tableaux et objets

Dans cet exercice, vous allez manipuler un tableau contenant l'ensemble des bornes et poteaux incendie à Angers Loire Métropole. Pour ce faire, nous avons stocké, dans le fichier **data.js**, le contenu de ce tableau dans une variable nommée `bornesIncendie` que vous manipulerez par la suite dans le fichier **bornes-incendie.js** afin de répondre aux questions suivantes. La Figure 1 montre l'ensemble des éléments à obtenir pour chaque question.

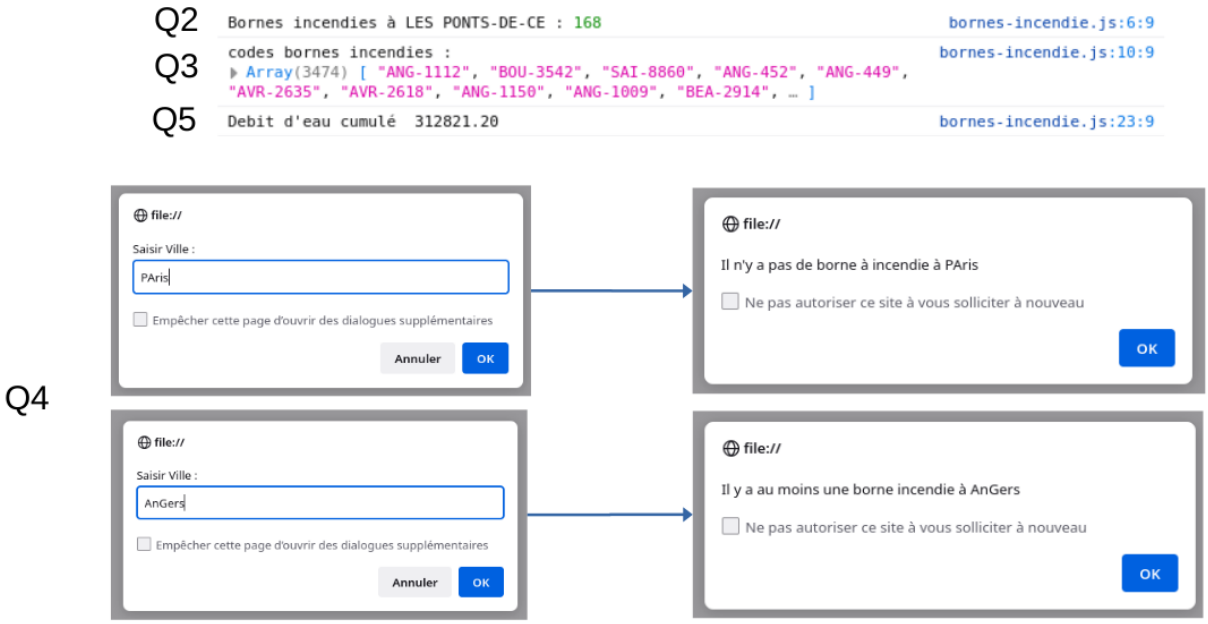

FIGURE 1 – Éléments à obtenir pour l'exercice.

1. Supprimez le premier et le dernier élément du tableau `bornesIncendie`. Après ces suppressions, la taille de `bornesIncendie` est de 3474.

2. Affichez dans la console le nombre de bornes à incendie présentes au sein de la commune dénommée `LES PONTS-DE-CE`.

3. Créez un tableau contenant un code, créé sur mesure, pour chaque borne. Le code sera composé des 3 premières lettres de la commune suivies d'un tiret et du numéro pompier (`num_pompier`).

4. Affichez dans une popup s'il existe au moins une borne incendie dans une ville saisie par l'utilisateur. Attention : les noms de communes sont stockés en majuscules dans le tableau `bornesIncendie`.

5. Affichez dans la console le débit d'eau cumulé des bornes à incendie du tableau `bornesIncendie`. Attention : pour certaines bornes, le débit prend la valeur `null`.

Exercice 2. Constructeurs et prototypes

Complétez le fichier **lieu.js** pour répondre aux questions qui suivent.

1. Créez un objet littéral `maine_et_loire` possédant une propriété entière dénommée `département` de valeur `49` et une méthode `toText` pour obtenir à la console le résultat illustré en Figure 2.

2. Implémentez un constructeur `Lieu` acceptant 2 arguments `type` et `département` - le premier étant une chaîne de caractères dénotant un type de lieu (p. ex. `"université"`, `"palais"`), le second, optionnel, étant un entier dénotant un numéro de département (p. ex. 49) - pour obtenir à la console le résultat illustré en Figure 4. Assurez-vous que chaque objet construit avec `Lieu` :

(1) possède une propriété propre dénommée `"type"` et égale au premier argument

(2) ait pour prototype l'objet `maine_et_loire`,

(3) possède une propriété propre dénommée `"département"` uniquement dans le cas où le second argument n'est pas passé à la construction.

3. Ajoutez une méthode `toText()` au constructeur `Lieu` pour obtenir à la console le résultat illustré en Figure 3.

```
>> maine_et_loire.toText();
<- "Département : 49"
```

FIGURE 2 –

```
>> ua.toText();
<- "type : université - Département : 49"
>> elysee.toText();
<- "type : palais - Département : 75"
```

FIGURE 3 –

```
>> var ua = new Lieu("université");
<- undefined
>> ua.département;
<- 49
>> ua.type;
<- "université"
>> var elysee = new Lieu("palais", 75);
<- undefined
>> elysee.département;
<- 75
>> elysee.type;
<- "palais"
```

FIGURE 4 –

Exercice 3. DOM : manipulation d'attributs HTML et mise en forme CSS

Le fichier **mendeleiev.html** importe le fichier **mendeleiev_3.js**. Complétez ce dernier, sans modifier le premier, pour répondre aux questions qui suivent.

1. Appliquez la police `Arial` au corps de la page et incorporez le texte "Tableau périodique des éléments chimiques" dans l'élément d'identifiant `titre-principal` (voir Figure 5).

2. Cochez toutes les cases à cocher (voir Figure 5).

3. Colorez les éléments de classe `Gaz_noble` en `lightblue` sauf les libellés des cases à cocher qui sont dans cette classe (voir Figure 5).

4. Affichez en console les données - numéro, nom, symbole et masse - contenues dans la $5^{ème}$ cellule du tableau (le Béryllium).

5. Implémentez la fonction `decolorer()` qui encadrera chaque cellule du tableau appartenant à (au moins) une classe par une bordure de style `"1px solid black"`. Testez la fonction en l'appelant dans la console ou en cochant le bouton "Noir et blanc" : le résultat attendu est illustré en Figure 6.

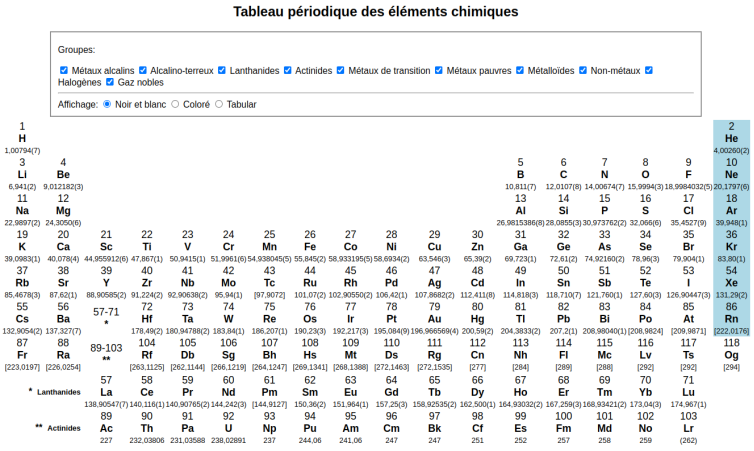

FIGURE 5 – Coloration des gaz nobles.

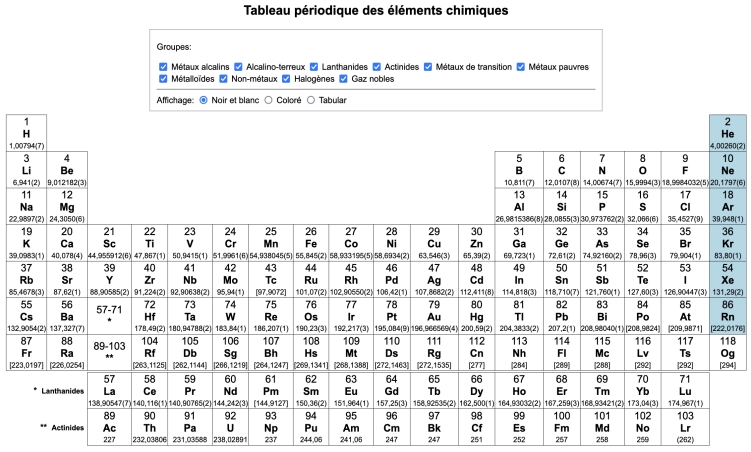

FIGURE 6 – Contour des cellules du tableau.

Exercice 4. DOM : Manipulation et création d'éléments HTML

Le fichier **mendeleiev.html** importe le fichier **mendeleiev_4.js**. Complétez ce dernier, sans modifier le premier, pour répondre aux questions qui suivent.

1. Complétez l'implémentation de la fonction `estElement` qui prend en argument une cellule du tableau et détermine si elle contient ou non des données. Précisément, la fonction retournera `true` si :

(1) la cellule n'est pas vide (autrement dit, elle a au moins 1 noeud enfant), et

(2) le nom indiqué dans la cellule est différent de `*` et `**`.

Testez la fonction en console qui retournera `true` sur la $1^{ère}$ cellule du tableau et faux sur la $59^{ème}$ cellule.

2. La fonction `transformerTableau` vise à remplacer le tableau périodique par un nouveau tableau HTML contenant une ligne par élément atomique, chaque ligne étant décomposée en 4 cellules contenant, respectivement, le numéro, nom, symbole et masse de l'élément. L'implémentation partielle qui vous est donnée crée le tableau et sa ligne d'en-têtes. Complétez-la pour obtenir la page illustrée en Figure 7 lorsque l'on exécute la fonction dans la console ou l'on coche le bouton "Tabular". Votre implémentation devra donc récupérer les données correspondant à chaque élément, créer et ajouter la ligne de l'élément au tableau, ajouter ce dernier à la page, puis supprimer le tableau d'origine. Utilisez la fonction `estElement` développée en question précédente pour ne traiter que les cellules "non vides" du tableau d'origine.

Numero	Nom	Symbole	Masse
1	Hydrogène	H	1,00794(7)
2	Hélium	He	4,00260(2)
3	Lithium	Li	6,941(2)
4	Béryllium	Be	9,012182(3)
5	Bore	B	10,811(7)
6	Carbone	C	12,0107(8)
7	Azote	N	14,00674(7)
8	Oxygène	O	15,9994(3)
9	Fluor	F	18,9984032(5)
10	Néon	Ne	20,1797(6)
11	Sodium	Na	22,9897(2)
12	Magnésium	Mg	24,3050(6)
13	Aluminium	Al	26,9815386(8)
14	Silicium	Si	28,0855(3)
15	Phosphore	P	30,973762(2)
16	Soufre	S	32,066(6)
17	Chlore	Cl	35,4527(9)

FIGURE 7 – Transformation du tableau périodique par appel à `transformerTableau()`.