Environnement lexical et contexte d’exécution

Cette leçon est très importante si vous voulez comprendre de nombreuses choses en `JavaScript`, aussi n'hésitez pas à y passer du temps et à y revenir plusieurs fois.

Elle vous permettra ensuite de comprendre **la pile d'exécution, les fermetures (`closures`), la portée des fonctions** et plein d'autres notions fondamentales.

L'environnement lexical

L'environnement lexical est l'endroit où sont stockés les identifiants, c'est-à-dire les noms des variables et des fonctions, et leurs valeurs (qui sont des références pour les objets et donc les fonctions).

L'environnement lexical a une référence à son environnement lexical parent.

Conceptuellement, cela ressemble à ça :

```
environnementLexical = {
  a: 25,
  obj: <référence de l’object>
  maFonction: <référence à la fonction>
  parent: <référence à l'environnement lexical parent>
}
```

Le contexte d'exécution

Le contexte d'exécution est une construction permettant de suivre l'exécution du code.

Comme le `JavaScript` est `single threaded`, **il n'y a toujours qu'un seul contexte d'exécution qui est en train d'exécuter du code à n'importe quel moment**.

Le moteur `JavaScript` va créer une pile d'exécution (appelée la `stack`).

Une pile est une structure de données de type `LIFO` (`Last In, First Out`).

Il est simple de se la représenter : imaginez une pile de papiers de tâches à faire. La dernière tâche qui a été mise sur la pile est la première à être exécutée et à s'enlever de la pile une fois l'exécution terminée.

Tout en bas de la pile, nous avons le **contexte d'exécution global**. A chaque fois qu'une fonction est invoquée, un **nouveau contexte d'exécution** est créé et ajoutée sur la pile.

Le nouveau contexte d'exécution est retirée lorsque l'exécution de la fonction est terminée.

La partie du code qui est en train d'être exécutée est toujours le contexte d'exécution au sommet de la `stack`.

Lien entre contexte d'exécution et environnement lexical

Pour chaque **contexte d'exécution** un **environnement lexical** est créé.

Cet environnement est ajouté dans une propriété interne `[[Environment]]` (auquel vous n'avez pas accès) sur chaque fonction.

Contexte d'exécution global et objet global

En `JavaScript`, un objet global est toujours créé par le moteur.

Dans un navigateur, les variables et les fonctions déclarées dans le contexte global sont créées sur l'objet global qui est `window`.

Dans `Node.js`, il y a bien un objet global mais les variables et les fonctions sont attachées à un module et non pas à cet objet.