

Cours 1

Environnement de travail et introduction à JavaScript

Intro. à la programmation - Aut. 2021



❖ Présentation

- ◆ Plan de cours et fonctionnement 
- ◆ Évaluations
- ◆ Matériel

❖ Plateformes scolaires

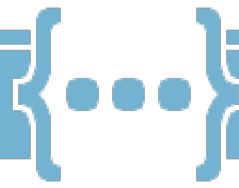
- ◆ Léa

❖ Environnement de travail

- ◆ Dossiers, fichiers et compression

❖ Introduction à JavaScript

- ◆ Opérations mathématiques, expressions et variables

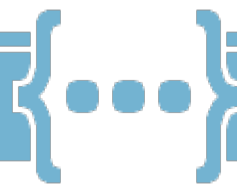


❖ Plan de cours 🤪

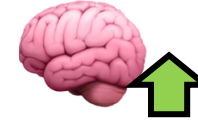
- ◆ Survolons-le rapidement
 - Emphase sur les **évaluations** et les **règles du cours**
- ◆ Vous pouvez le consulter plus en détails de votre côté

❖ Matériel

- ◆ Une clé USB de **8 Go** sera suffisante pour ce cours
 - Alternatives : Dropbox, Google Drive, espace de stockage Omnivox
 - Pas de manuel pour ce cours! On économise notre \$



❖ Semaine typique en Intro à la programmation



Deux fois par semaine 🤔

Une fois par semaine

Théorie 🧠

Quiz Kahoot 🎲

Exercices 🖋️

Remise des
exercices ✉️

15 à 45 minutes pour
aborder des concepts
théoriques.

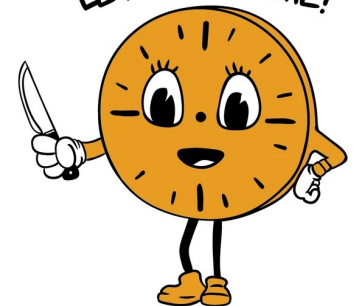
10 à 20 minutes pour
faire un **quiz anonyme
interactif** (qui n'est pas
évalué) pour pratiquer
les notions théoriques
en groupe

1h à 2h pour réaliser
des exercices en classe
(et terminer en dehors
des cours si nécessaire)

Les exercices réalisés
lors des deux cours de
la semaine doivent être
remis le **samedi
suivant avant 23h59**

Interchangeables

LET'S KILL TIME!





❖ Omnivox - Léa

◆ Petite démo pour...

- Récupérer les notes de cours
- Récupérer les laboratoires
 - Remettre les laboratoires

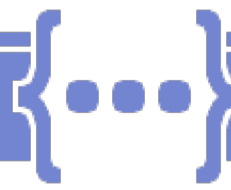
❖ Ordinateurs du cégep

◆ Tous les logiciels dont on a besoin sont installés dessus!

- Petite démo : où ranger nos fichiers pour ne pas les perdre?

◆ Installer le logiciel WebStorm à la maison

- Voir capsule vidéo sur le site du cours pour la procédure

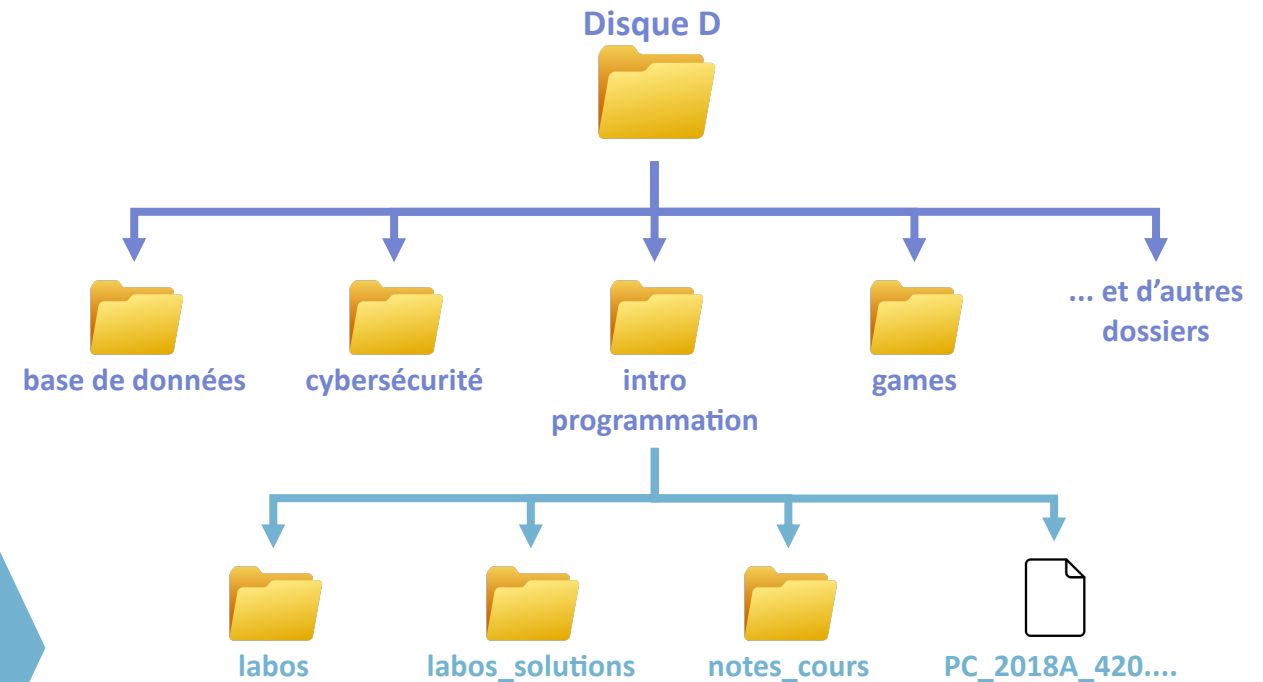


❖ Arborescence de dossiers

- ◆ Dans un système d'exploitation, les dossiers et fichiers sont organisés en arborescence (ou en hiérarchie...)

Dossier « Disque D »

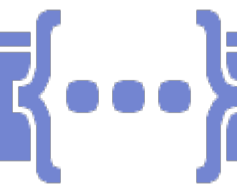
Ce PC > Disque D (D:) >		
Nom	Modifié le	Type
base de données	2020-12-19 13:00	Dossier de fichiers
cybersécurité	2021-04-23 19:13	Dossier de fichiers
games	2021-02-06 18:33	Dossier de fichiers
intro programmation	2021-07-30 15:58	Dossier de fichiers
Introduction aux bases de données	2021-04-09 00:51	Dossier de fichiers
logiciels de bureautique	2021-06-10 14:04	Dossier de fichiers
Programmation Web Serveur	2021-06-05 14:14	Dossier de fichiers



Dossier « intro programmation »

Ce PC > Disque D (D:) > intro programmation >		
Nom	Modifié le	Type
labos	2021-07-29 20:18	Dossier de fichiers
labos_solutions	2021-07-29 20:18	Dossier de fichiers
notes_cours	2021-07-30 16:04	Dossier de fichiers
PC_2018A_420-905-EM_Département-Inf...	2021-07-28 20:34	Document Micros...

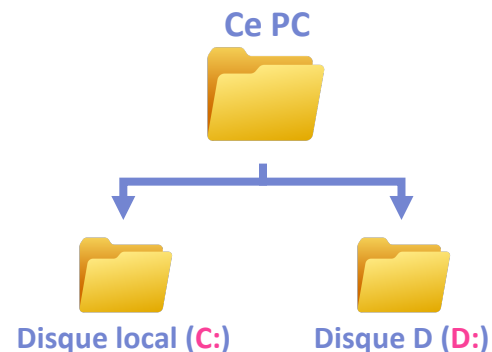
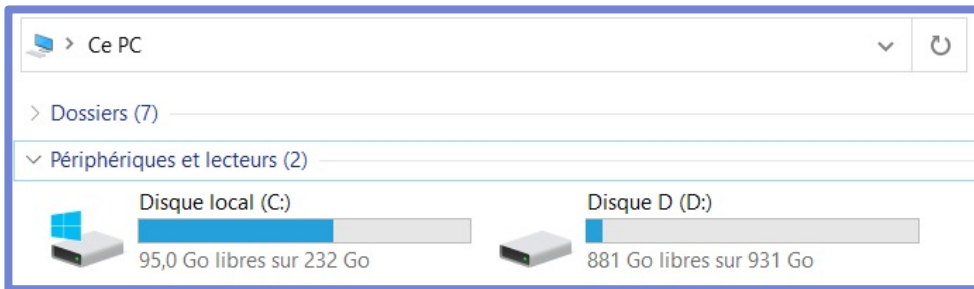




❖ Arborescence de dossiers

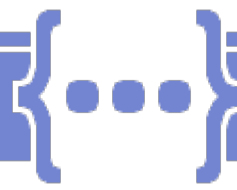
- ◆ La *racine* : c'est le tout début de l'arborescence, le « dossier qui contient tous les dossiers »
 - Sur **Windows 10**, il est nommé « **Ce PC** », par exemple

Dossier racine : « Ce PC »



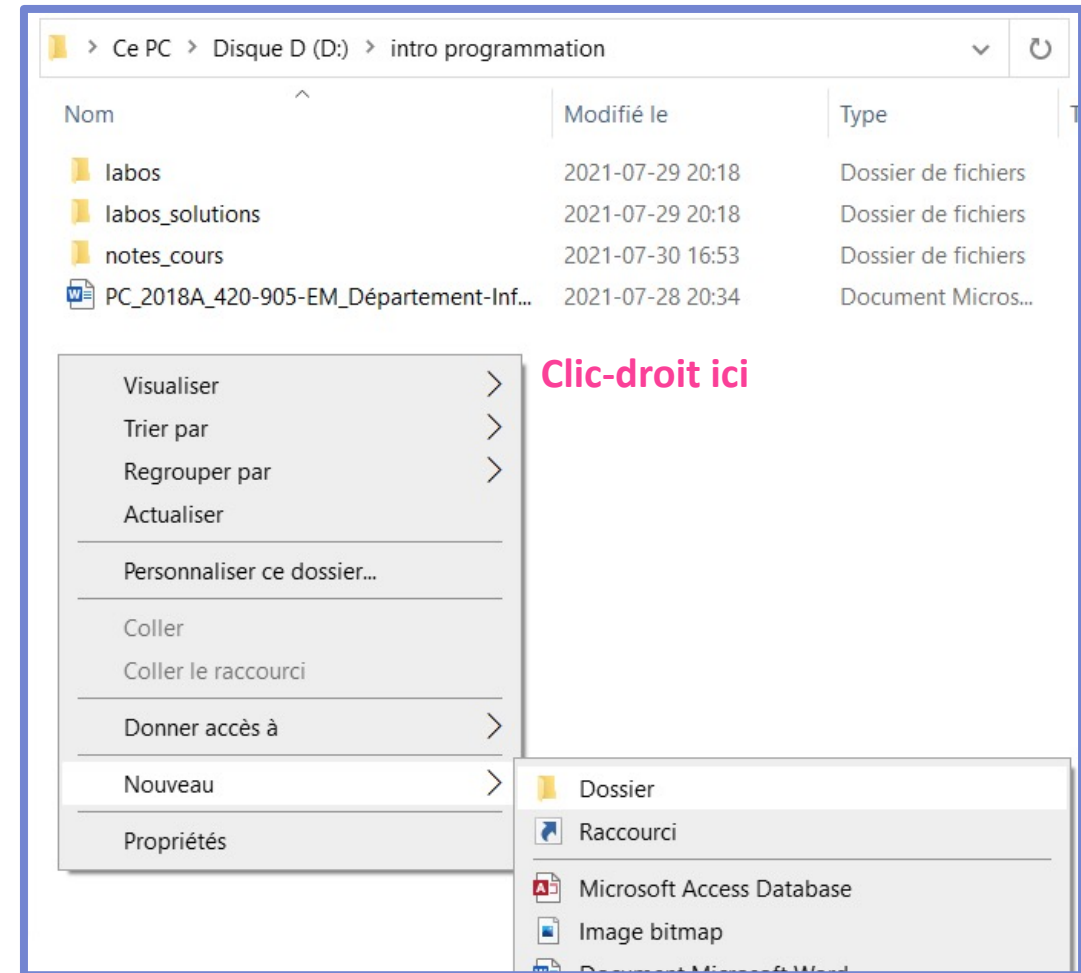
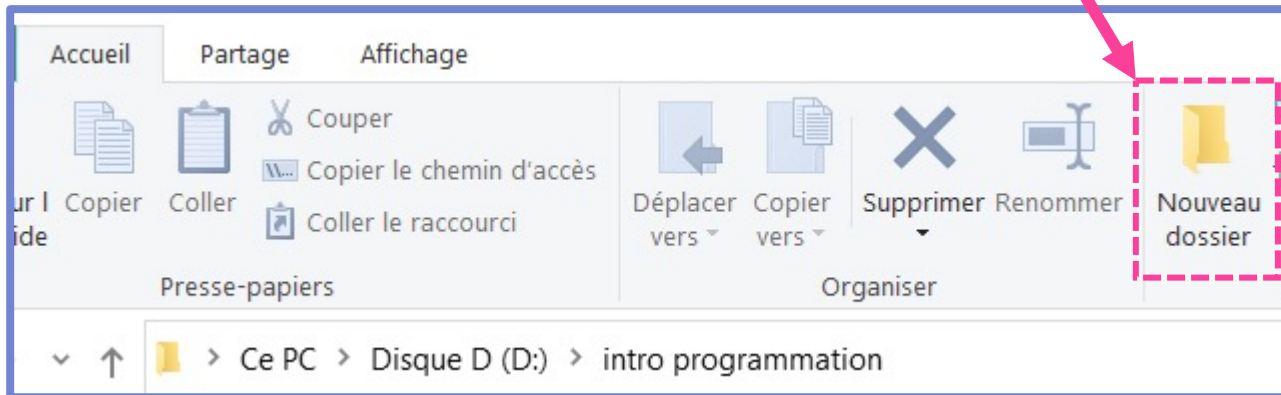
❖ Dans cet exemple, il y a deux « Disques » (**C:** et **D:**)

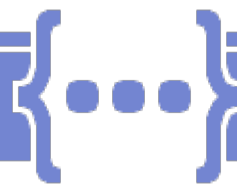
- ◆ Ce sont les deux disques qui stockent les données de l'ordinateur!
- ◆ Généralement, il n'y a qu'un seul disque (le **C:**)
- ◆ Si on branchait une **clé USB** dans l'ordinateur, on verrait qu'un nouveau « **Disque** » apparaîtrait. (**E:**, **F:**, **G:**, ou autre ...)
 - On pourrait accéder à son contenu ici



❖ Créer un dossier

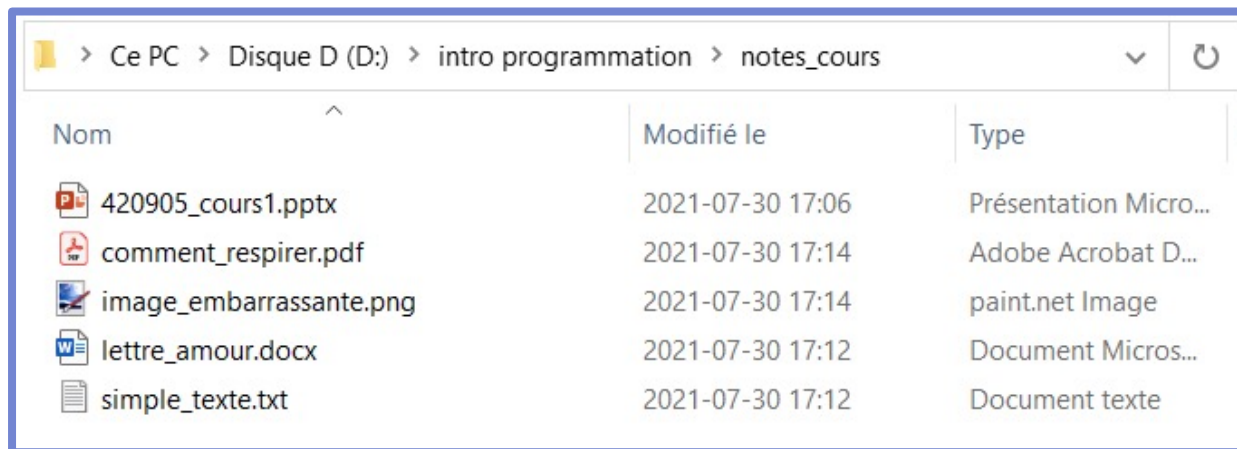
- ◆ Se rendre au dossier dans lequel on souhaite ajouter un dossier
- ◆ Faire un **clic-droit** dans un espace vide
 - Choisir « Nouveau → Dossier »
 - Nommez-le !








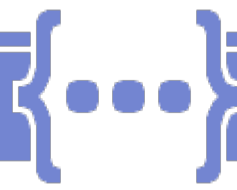


❖ Extensions de fichier

- ◆ Indiquent le **type de fichier** d'un document. Quelques exemples...
 - **.pptx** : Présentation Microsoft Powerpoint
 - **.pdf** : Document de rendu, de format vectoriel
 - **.png** (ou .jpeg, .bmp, .gif, etc.) : Image
 - **.docx** : Traitement de texte avec Microsoft Word (ce n'est pas un fichier texte!)
 - **.txt** (ou .html, .css, .js, etc.) : Simple fichier de texte (ça, ce sont des fichiers texte!)
- ◆ Les icônes à droite des fichiers peuvent également indiquer le type

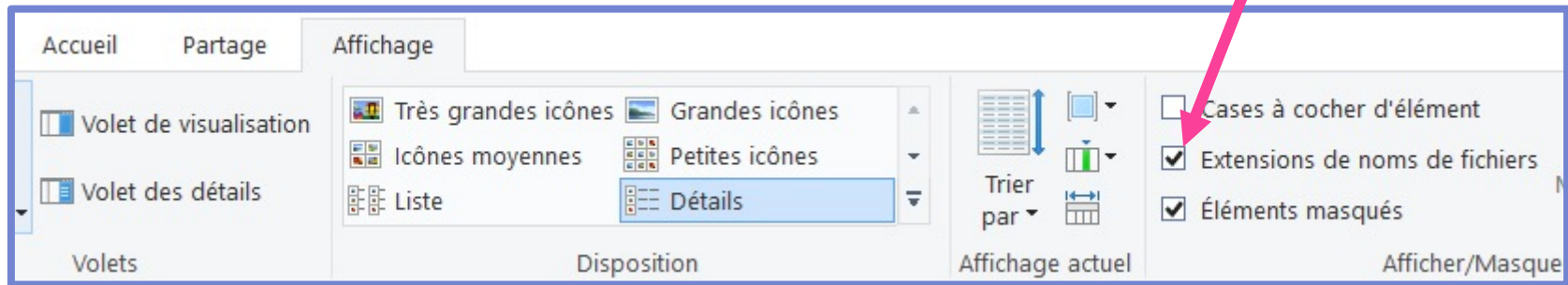


Ce PC > Disque D (D:) > intro programmation > notes_cours			
Nom	Modifié le	Type	
 420905_cours1.pptx	2021-07-30 17:06	Présentation Micro...	
 comment_respirer.pdf	2021-07-30 17:14	Adobe Acrobat D...	
 image_embarrassante.png	2021-07-30 17:14	paint.net Image	
 lettre_amour.docx	2021-07-30 17:12	Document Micros...	
 simple_texte.txt	2021-07-30 17:12	Document texte	



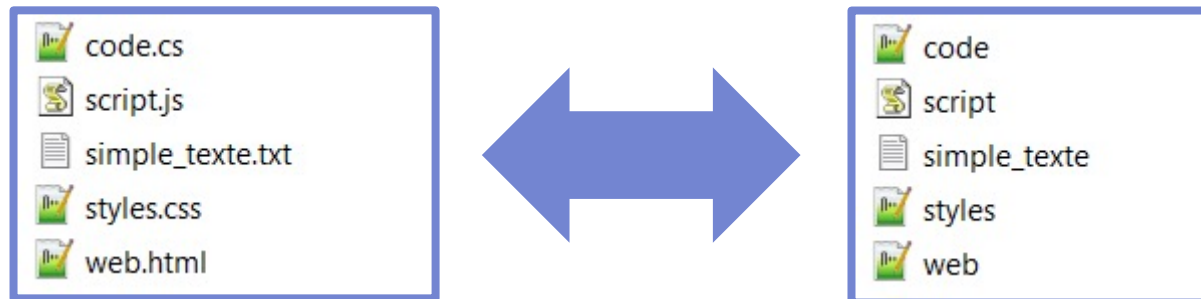
❖ Extensions de fichier

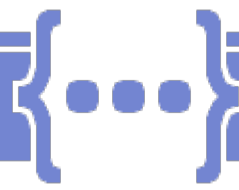
- ◆ Assurez-vous d'afficher les extensions de fichier s'ils sont cachés



Le menu « **Affichage** » est disponible depuis n'importe quel dossier!

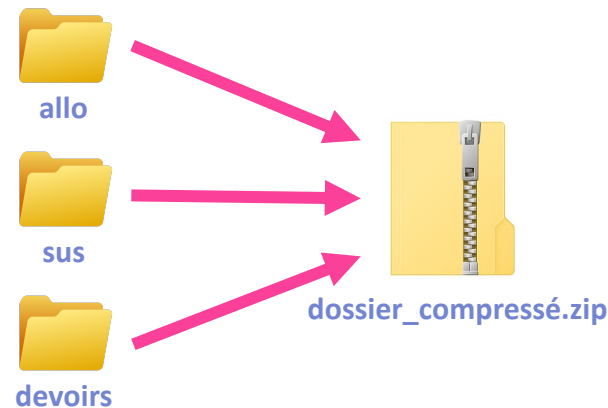
- ◆ Sinon il peut être difficile de différencier certains types...

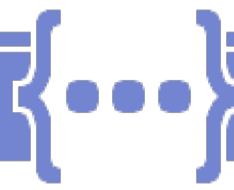




❖ Compression de fichier

- ◆ C'est une action qui permet de « **regrouper des fichiers/dossiers** »
 - Réduit potentiellement **leur taille** (en données)
 - Permet de « **Partager** » / « **Envoyer** » un ou plusieurs dossiers. Par exemple on peut :
 - Le téléverser (*upload*) sur Léa
 - L'envoyer par courriel
 - Le stocker dans le Cloud (Dropbox, Google Drive, etc.)
 - Lorsque compressés, les fichiers ne peuvent pas être utilisés!
 - Il faut d'abord « **décompresser** »

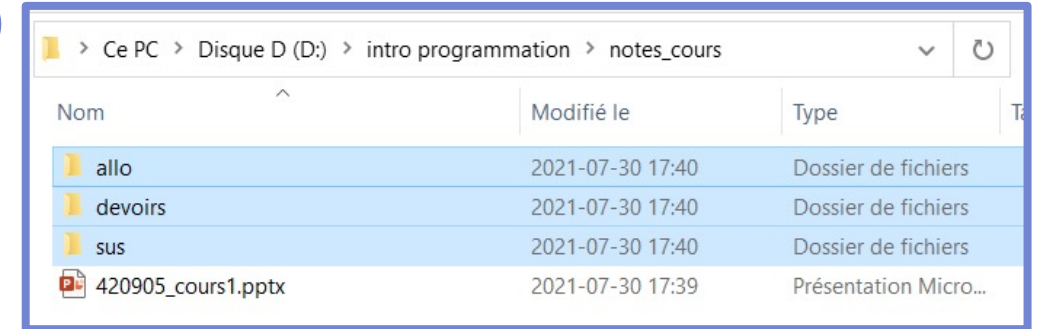




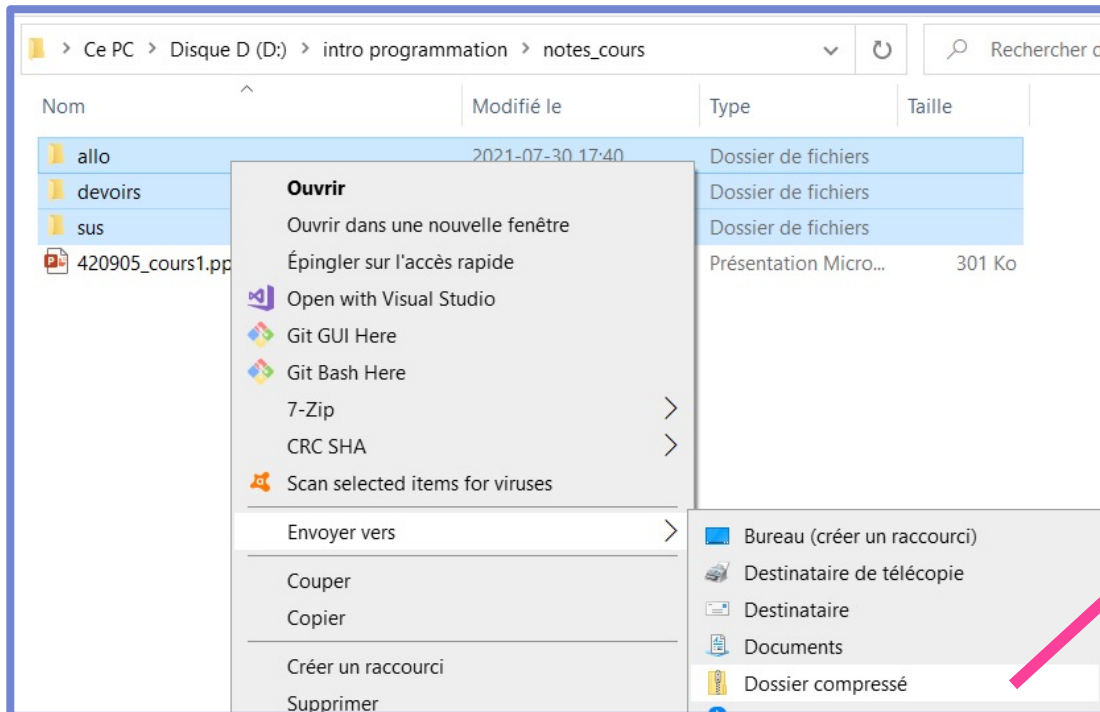
❖ Compresser des fichiers

- 1) Sélectionner les fichiers et/ou dossiers
- 2) Clic-droit sur l'un d'eux → Envoyer vers → Dossier compressé

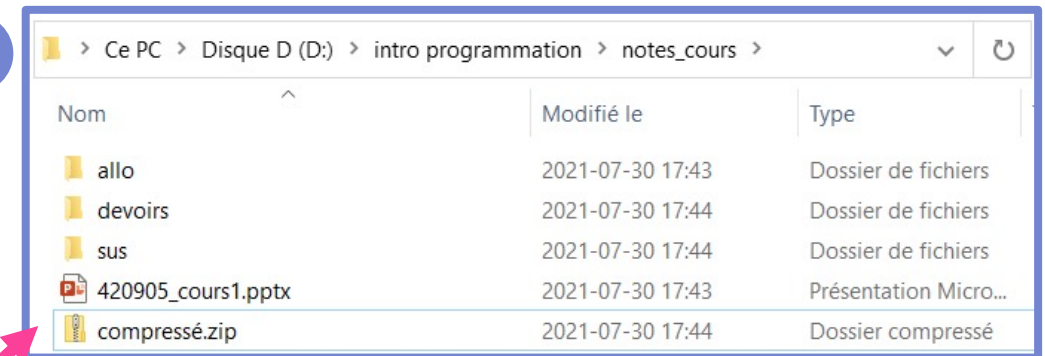
1



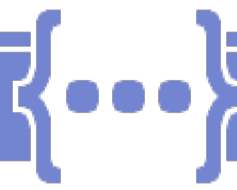
2



3



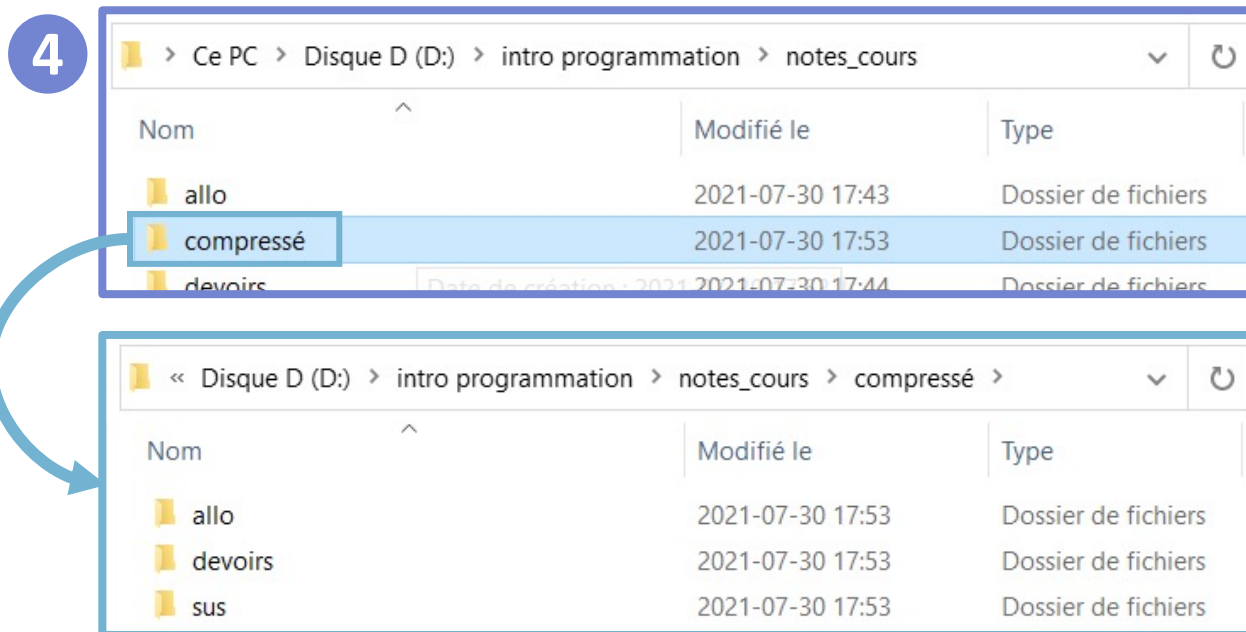
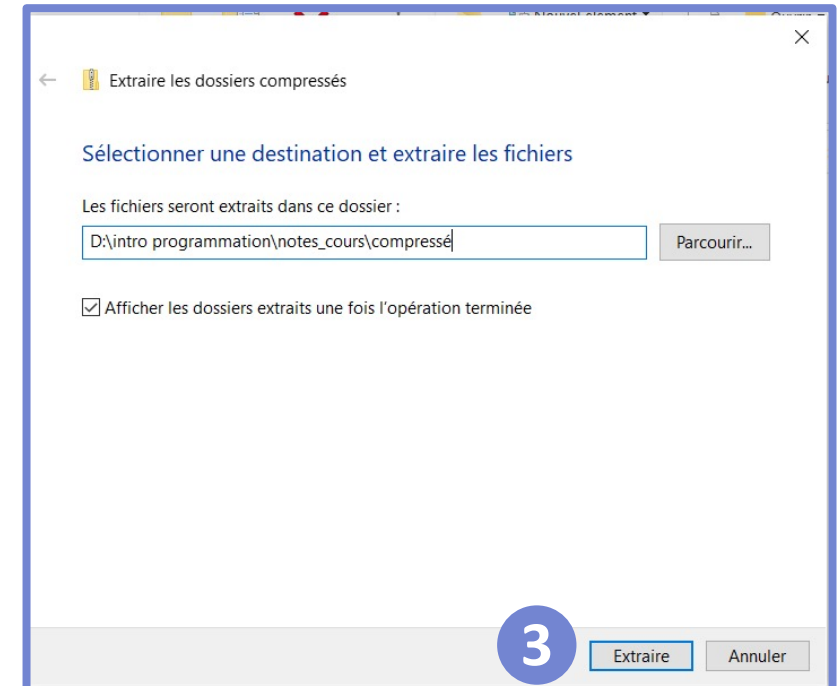
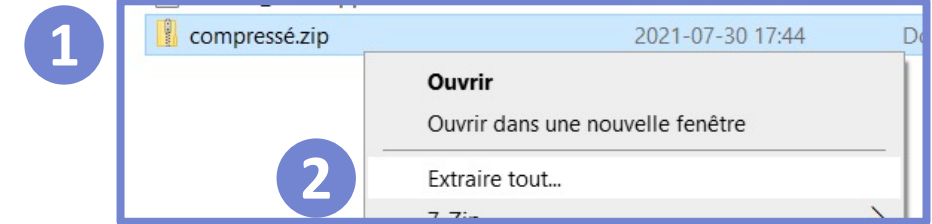
- ❖ On obtient donc un **dossier compressé** qui se termine avec l'extension **.zip**
 - ◆ On peut le renommer, tant que l'extension **.zip** est conservée
 - ◆ Les dossiers/fichiers initiaux ne sont pas supprimés!

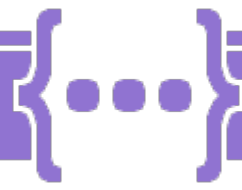


❖ Décompresser un fichier

◆ Permettra de le rendre utilisable à nouveau

- 1) **Clic-droit** sur le **fichier compressé**
- 2) **Extraire tout...**
- 3) Appuyer sur « **Extraire** » dans la nouvelle fenêtre
- 4) On obtient une **copie décompressée**





❖ Qu'est-ce que JavaScript?

◆ Langage de programmation né en 1996

- Les fichiers de code JavaScript ont l'extension **.js**  script.js
 - (et ce sont de simples fichiers texte)

◆ Très utilisé pour la programmation **Web**

- La très très grande majorité des sites Web l'utilisent
 - Et c'est une des raisons qui en font un langage de choix pour apprendre à coder!
- On peut utiliser ce langage dans les navigateurs **Web**! (Firefox, Chrome, Edge, etc.)
 - Nous allons le faire dans ce cours!

◆ Exemples de produits qui utilisent JavaScript

NETFLIX



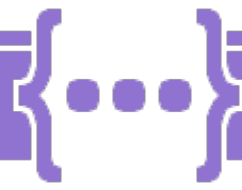
LinkedIn

Uber



2048





❖ Qu'est-ce que JavaScript?

- ◆ Exemple de morceau de code JavaScript
 - Pas très intuitif pour le moment...!

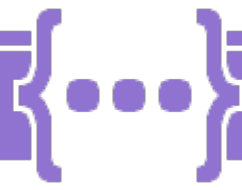
```
>
class Vehicle {
  constructor(make, model, color) {
    this.make = make;
    this.model = model;
    this.color = color;
  }

  getName() {
    return this.make + " " + this.model;
  }
}

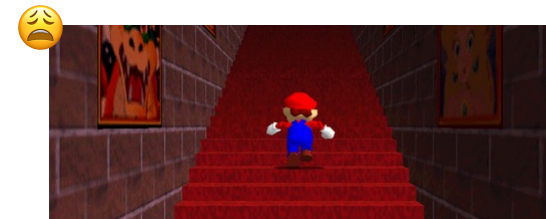
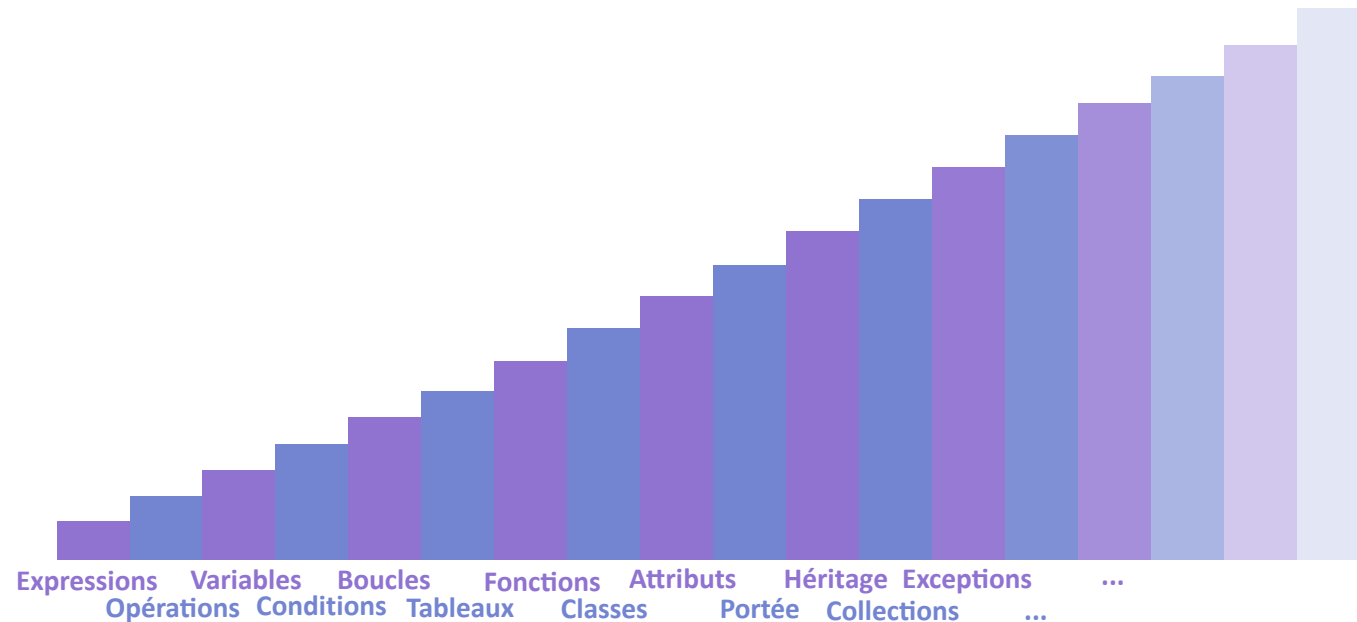
class Car extends Vehicle{
  getName(){
    return super.getName() + " - called base class function from child class.";
  }
}

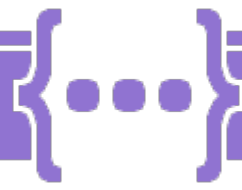
let car = new Car("Honda", "Accord", "Purple");

car.getName();
< "Honda Accord - called base class function from child class."
```



- ❖ Apprendre un langage de programmation
 - ◆ Il y a une **longue route** avant de pouvoir « coder des choses concrètes et complexes » comme des jeux, des sites Web et des applications
 - Cette longue route est différente pour chaque type de projet, et nécessite parfois d'apprendre d'autres **langages de programmation** ou **technologies**





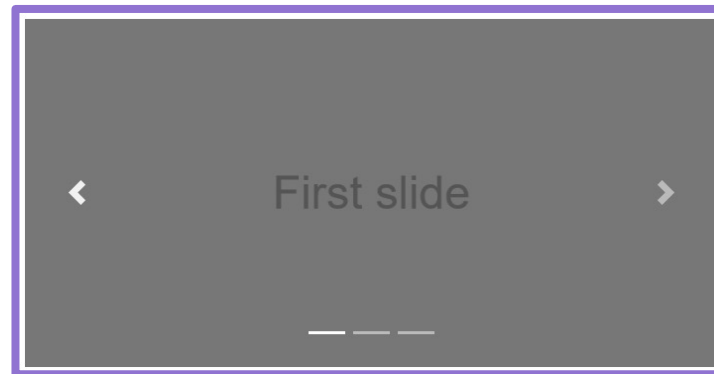
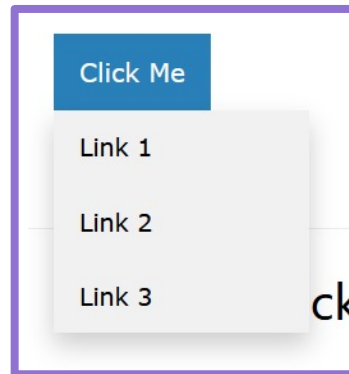
❖ Apprendre un langage de programmation

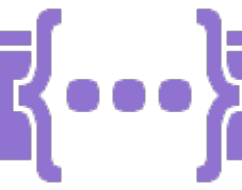
◆ Dans le cadre du cours, **JavaScript** va nous permettre de modifier/interagir avec les éléments d'une **page Web** pour la rendre **interactive**

○ Exemples :

- Un bouton change la couleur du texte
- Survoler un élément fait dérouler un menu avec plusieurs options
- Une galerie d'images qui alternent automatiquement

◆ Mais avant, nous avons beaucoup de notions à aborder pour pouvoir faire cela!





❖ Introduction à **JavaScript**

- ◆ JavaScript avec un navigateur Web
- ◆ Opérateurs arithmétiques de base
- ◆ Variables
 - Conventions de nommage, déclaration, affectation
- ◆ Types de données
- ◆ Autres opérateurs arithmétiques
 - Opérateurs d'affectation
 - Priorité des opérateurs
- ◆ Usage de variables
- ◆ Concaténation



❖ JavaScript avec un navigateur Web

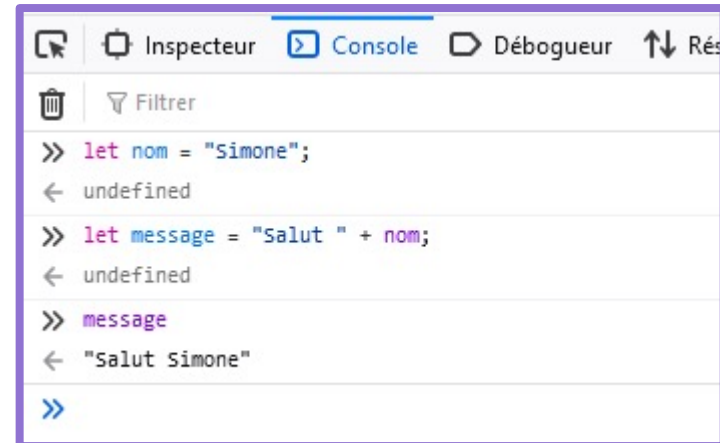
- ◆ Pour le moment, nous utiliserons la « Console du navigateur Web » de **Google Chrome** ou **Mozilla Firefox** pour pratiquer avec JavaScript
- ◆ Ouvrez un **navigateur Web** et appuyez sur **F12** (ou faites clic-droit -> Inspecter -> Console)

Vous pouvez
écrire du code
dans la **console**!



```
>> let a = 1;
← undefined
>> let b = a + 1;
← undefined
>> a + b
← 3
>>
```

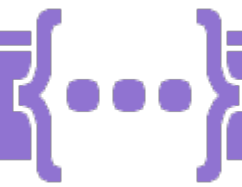
Exemple avec **Google Chrome**



```
>> let nom = "Simone";
← undefined
>> let message = "Salut " + nom;
← undefined
>> message
← "Salut Simone"
>>
```

Exemple avec **Mozilla Firefox**





❖ Opérateurs arithmétiques

◆ Les programmes nécessitent souvent de faire des calculs mathématiques

◆ Opérateurs simples :

○ **Addition +**

5.5 + 3.5 (Vaut 9)

7 + -3 (Vaut 4)

○ **Soustraction −**

3 - 4 (Vaut -1)

5 - 1.5 (Vaut 3.5)

○ **Multiplication ***

2 * 3 (Vaut 6)

-2 * 1.5 (Vaut -3)

○ **Division /**

3 / 2 (Vaut 1.5)

30 / -5 (Vaut -6)

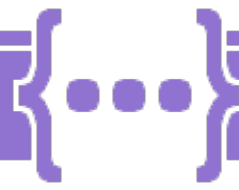
○ **Modulo %**

5 % 2 (Vaut 1)

8 % 3 (Vaut 2)

- Restant d'une division entière

Exemple : 5 divisé par 2... 2 rentre deux fois dans 5, puis il reste 1. Donc 1 est le résultat.



❖ Opérateurs arithmétiques

◆ Usage de **nombre**s décimaux

- Toujours utiliser un **point** (et non une virgule) pour séparer la partie entière de la partie décimale!

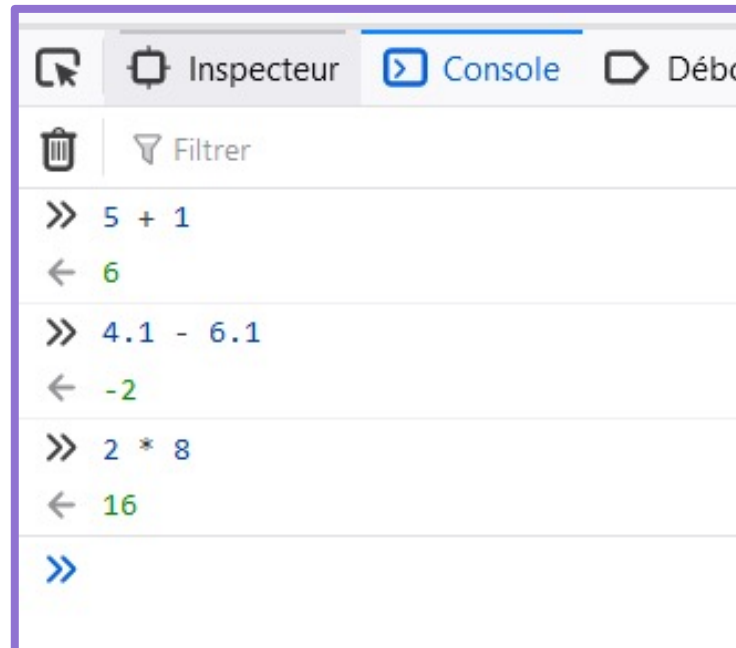
5.5

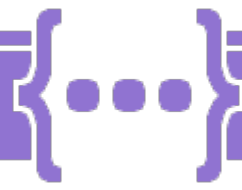
10.97

31.335

-52.5

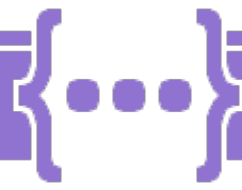
- On peut faire des calculs avec la console d'un navigateur!





❖ Variables

- ◆ Espace dans la mémoire permettant de stocker une **donnée**
 - Dans un programme (ou logiciel, ou jeu, ou application, ...), on a parfois besoin de **stocker des informations** pour assurer son bon fonctionnement
 - **Exemples**
 - Stocker les points de vie d'un personnage dans un jeu : « **78** » (points de vie) ❤️
 - Stocker la taille du pinceau utilisé dans Photostop : « **5.5** » (pixels) 🎨
 - Stocker le nom d'un item dans un jeu mobile : « **Emerald** » 💎
 - Stocker votre niveau de concentration en classe : « **10** » (%) 🙄
 - Les **variables** servent exactement à stocker ce genre de données!



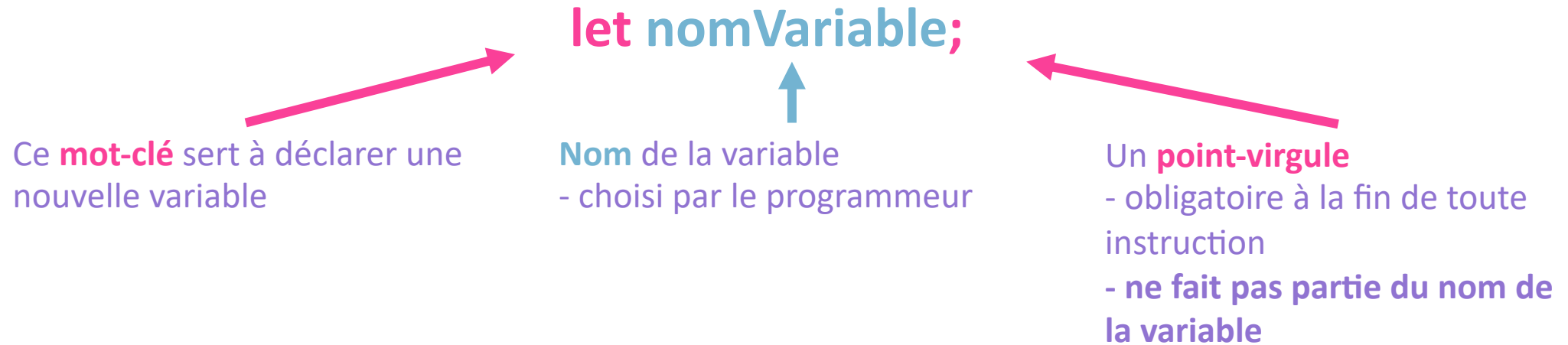
❖ Variables

- ◆ Chaque variable possède un nom qui permet de la distinguer
- ◆ **Règles** de nommage (obligatoires)
 - Doit commencer par une **lettre**
 - Peut contenir des **lettres**, des **chiffres** et des **traits de soulignement** _
 - **Ne** peut **pas** contenir d'**espace** ni d'autres **caractères spéciaux** (?!\$#/%&*~\)
- ◆ **Conventions** de nommage (fortement suggérées)
 - Le nom d'une variable doit être **significatif** (nomDragon, prix, age, abc, lmao, hm, ...)
 - Si le nom est composé de plusieurs mots, **le premier commence par une minuscule** et les suivants par des **majuscules** (birthDate, numberOfStudents, pointsDeVie, ...)



❖ **Déclaration** d'une variable

- ◆ Déclarer une variable permet de la créer et de pouvoir l'utiliser par la suite dans le code
- ◆ Il faut utiliser la forme suivante pour déclarer une variable :



- ◆ Exemple

let prixRubis;



❖ Affecter une valeur à une variable

- ◆ Permet de stocker une information dans une variable
- ◆ L'affectation utilise l'opérateur =

prixRubis = 800;

Nom de la variable affectée
-> placé à gauche du =

Valeur affectée
-> placée à droite du =

- ◆ Exemple : on **déclare** 2 variables, puis on les **affecte**

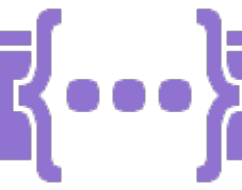
let prixRubis;

let prixEmeraude ;

prixRubis = 800;

prixEmeraude = 500 ;

Les **points-virgules** peuvent être collés ou espacés à la fin d'une instruction



❖ Déclarer et affecter autrement

- ◆ On peut **déclarer** plusieurs **variables** d'un coup (séparées par des **virgules**)

let variable1, variable2, variable3 ;

- ◆ On peut **déclarer** et **affecter** immédiatement

let ageRoland = 102 ;

- ◆ On peut **déclarer** et **affecter** plusieurs **variables** d'un coup

let triangle = 3, carre = 4, pentagone = 5 ;

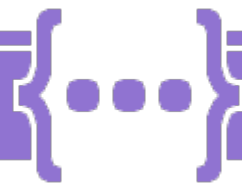
- ◆ On peut **affecter** une valeur pour **écraser** / **effacer** une précédente valeur

let prixRubis = 700 ;

prixRubis = 800;



prixRubis contient donc la valeur 800 à partir de maintenant plutôt que 700!



❖ Types de données

- ◆ Lorsqu'on **affecte** une **valeur** à une **variable**, on associe la variable à un **type de données**
- ◆ Exemples de types de données
 - **Nombre entier** (1, -5, 351, ...)
 - **Nombre décimal** (-8.54, 57.2241, ...)
 - Toujours utiliser un **point** (et non une virgule) pour séparer la partie entière de la partie décimale!
 - **Chaîne de caractères** ("Le cours est plate", "69420", ...)
 - Les **chaînes de caractères** sont TOUJOURS encadrées par des **guillemets** doubles (" ... ") ou simples (' ... '). Cela permet de les différencier des **noms de variables** ou des **nombres**

let nbAmis = 0 ;



Nombre entier

let prixTicket = 34.99 ;



Nombre décimal

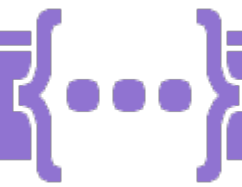
let nomRoi = "Arthur" ;



Chaîne de caractères

let noPorte = "121" ;





❖ Types de données

- ◆ Le **type de donnée** d'une variable peut **changer** sans problème avec une affectation

```
let prixJouet = 40;
```



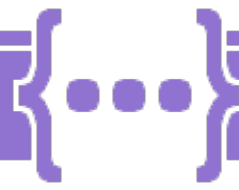
Nombre entier

```
prixJouet = 39.99;
```



Nombre décimal (le type de donnée a changé!)

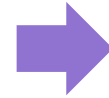
- ◆ Le **type de donnée** détermine l'espace nécessaire dans la mémoire pour une variable
 - Les **nombres décimaux** prennent plus de place (plus de bits) que les **nombres entiers** dans la mémoire
 - Une **chaîne de caractères** prend encore plus de place!
 - Ne vous inquiétez pas : on peut déclarer énormément de variables sans problème



❖ Opérateurs d'affectation

◆ Affectation simple =

- Affecte simplement la valeur à droite



let a = 4 * 3; (a contient 12)

◆ Incrémentation ++

- Affecte la valeur actuelle + 1



let a = 4;
a++; (a contient 5)

◆ Décrémentation --

- Affecte la valeur actuelle - 1



let a = 4;
a--; (a contient 3)

◆ Autres affectations +=, -=, *=, /=, % =

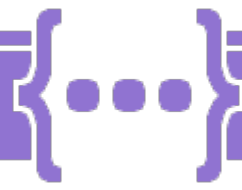
- Affecte la valeur actuelle, + une autre valeur (ou - * / %)



let a = 2;
a += 3; (a contient 5)
Équivalent à **a = a + 3;**



let a = 5;
a *= 2; (a contient 10)
Équivalent à **a = a * 2;**



❖ Priorité des opérateurs

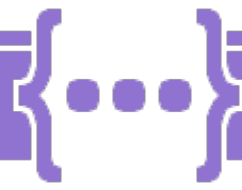
◆ Ordre de priorité (du premier au dernier)

- Parenthèses **()**
- Multiplication, division et modulo *** / %**
- Addition et soustraction **+ -**
- Affectation **=**

let **a** = **4 + 6 / 2 + 3**; (**a** contient **10**)

let **b** = **(4 + 6) / 2 + 3**; (**b** contient **8**)

◆ Les **parenthèses** permettent donc de modifier la **priorité des opérations**



❖ Concaténation

- ◆ L'opérateur **+** fonctionne différemment dès qu'une donnée de type **chaîne de caractères** fait partie de l'opération

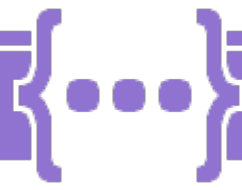
5 + 5 (vaut **10**)

5 + "5" (vaut **"55"** et devient une **chaîne de caractères**)

"Salut" + " " + "Simone" (vaut **"Salut Simone"**)



Ceci est une **chaîne de caractères** qui ne contient qu'une **espace** (une espace est un caractère comme un autre)



❖ Usage de variables dans les opérations

- ◆ Une variable qui contient une valeur peut être utilisée dans des opérations

```
let nbGemmes = 5 ;
```

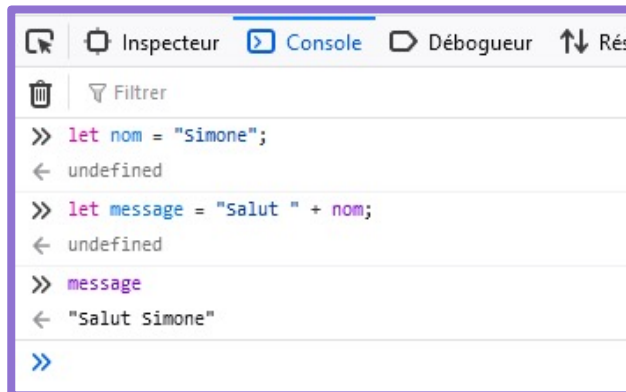
```
let prixGemme = 80 ;
```

```
let prixTotal = prixGemme * nbGemmes; (vaut 400)
```

```
let nom = "Roland" ;
```

```
let qualificatif = "lâche" ;
```

```
let message = nom + " est " + qualificatif; (vaut "Roland est lâche")
```



Dans la **console** Chrome/Firefox, on peut vérifier le contenu d'une variable en tapant son nom

Ici, on a écrit « **message** » et la **console** nous a retourné sa valeur : « **Salut Simone** »