JavaScript

Premiers pas dans le monde de la programmation

Antoine Meresse

L’appel de la tortue

Comprendre la syntaxe,
l’ordre des instructions et
comment lier le code
JavaScript à son HTML

Les instructions de base :

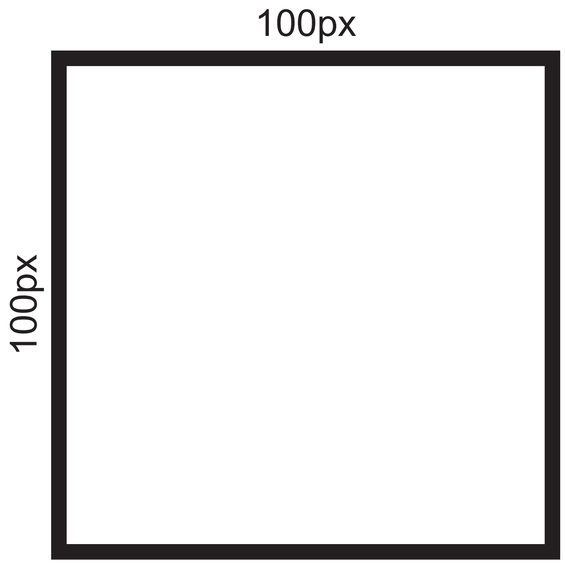
- forward(X) : fait avancer de X pas la tortue.

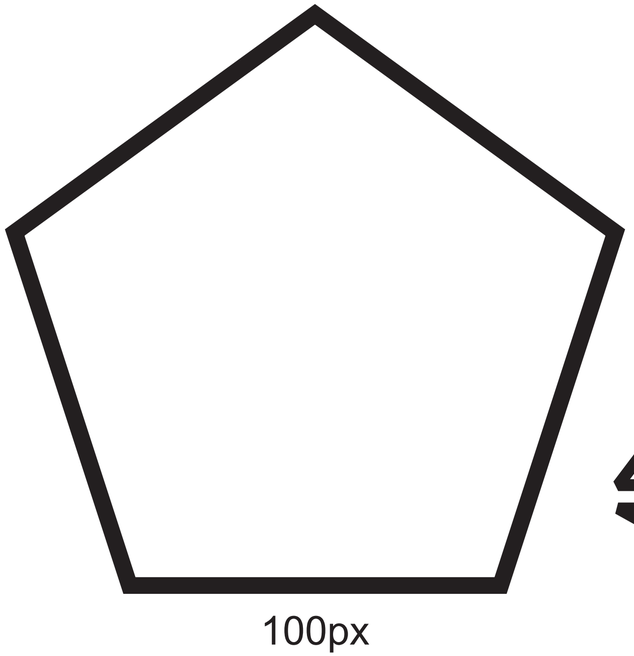
- left(angle) : effectue une rotation vers la gauche de « angle » degrés

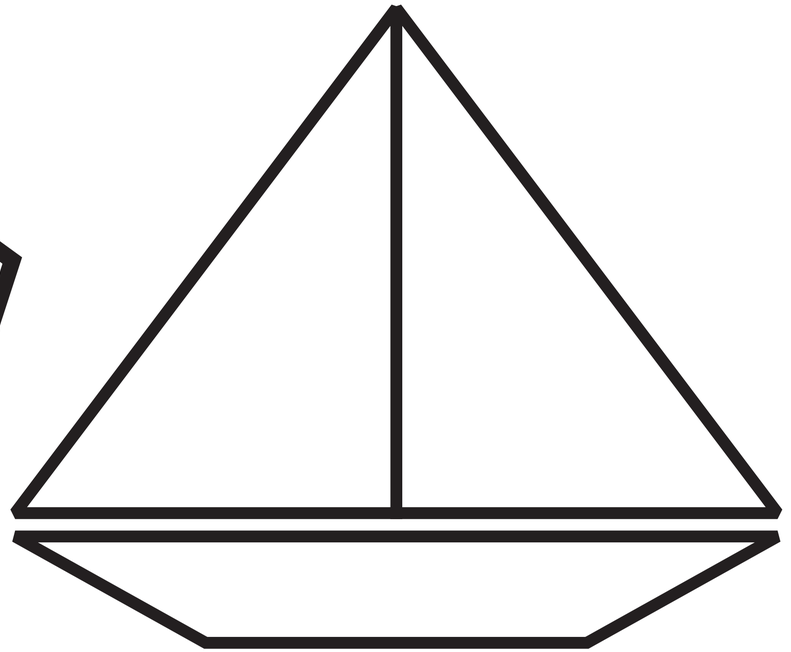
- right(angle) : pareil que left, mais vers la droite

Exercices

Reproduisez avec la tortue les dessins suivants :







On utilisera penup() pour se déplacer sans dessiner
et pendown() pour reprendre le dessin

La programmation modulaire

Découper un problème en plus petits problèmes.

À la découverte des fonctions !

Les fonctions : principe de base

- C'est un morceau de code qui sert à faire quelque chose de précis.

Pour créer une fonction, on écrit :

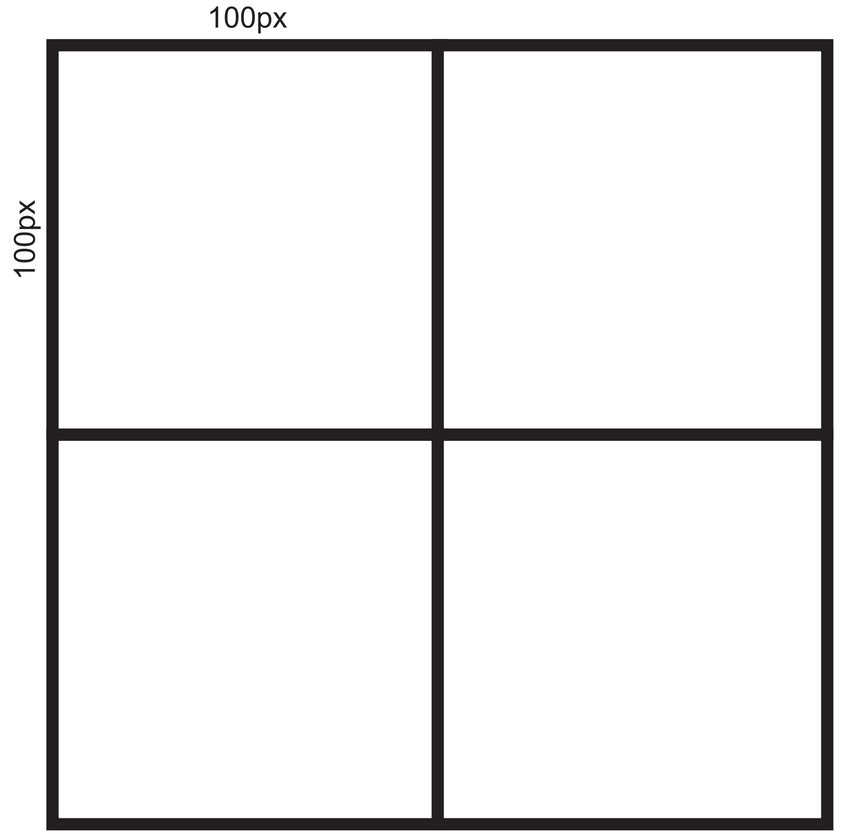
```
function nomDeMaFonction() {
  Code de ma fonction;
}
```

Pour utiliser la fonction, on utilise la syntaxe :

```
nomDeMaFonction();
```

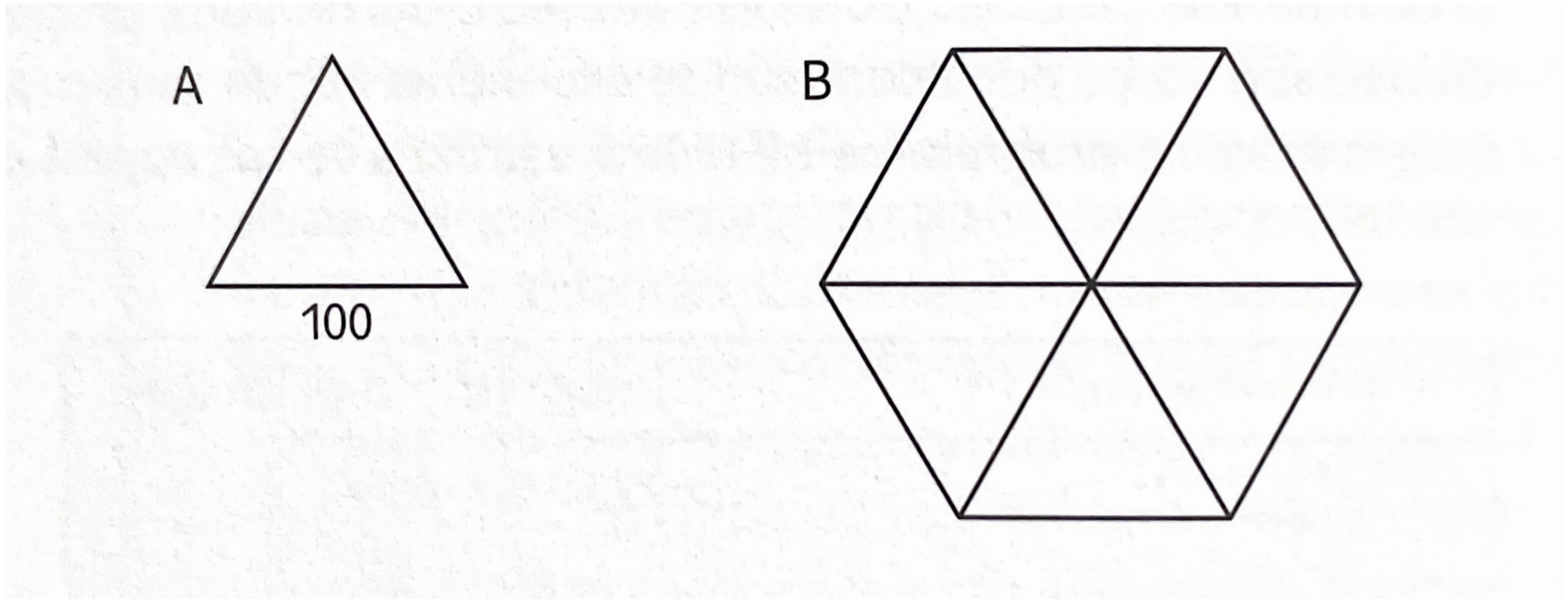
Exercice 1:

Créez une fonction permettant de créer un carré de 100 de coté puis reproduisez la figure suivante :



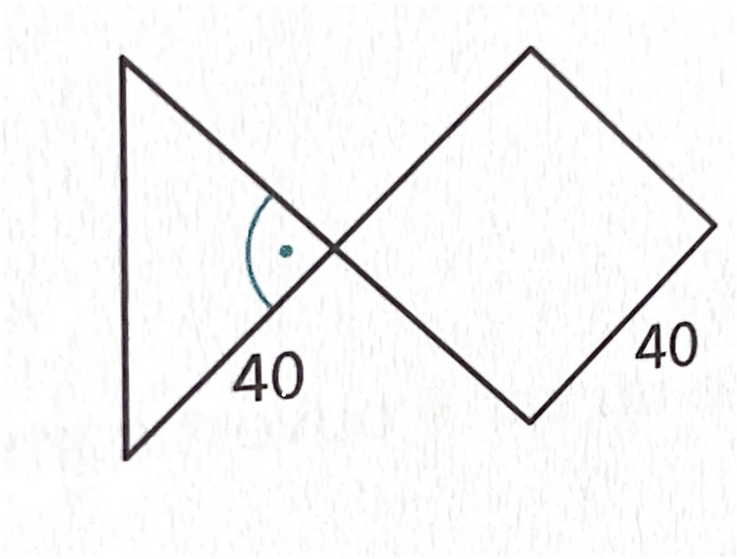
Exercice 2:

Créez une fonction permettant de créer un triangle équilatéral de 100 de coté puis reproduisez la figure suivante :



Exercice 3:

- Créez une fonction permettant de créer un triangle rectangle isocèle de 40 de coté puis une fonction permettant de créer un carré de 40 de coté.
- À l'aide de ces deux fonctions, créez une qui permet de de dessiner un poisson comme ci-dessous :



La programmation modulaire : suite !

Précédemment nous avons vu comment ajouter des commandes en JS.

Rendons cela plus paramétrable!

Les fonctions : Les paramètres

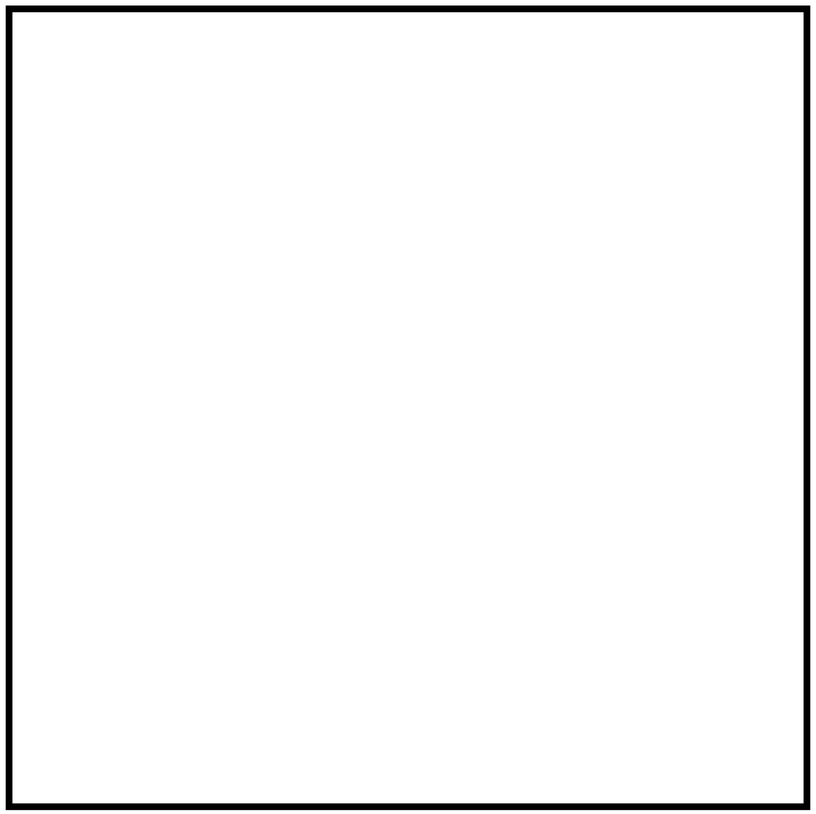
```
function nomDeMaFonction(paramètre) {
  forward(paramètre);
}
```

Lors d'un appel à une fonction, on peut lui passer un paramètre afin de changer le comportement de la fonction. (exemple avec forward(x) où x est un paramètre qui doit contenir un nombre qui donne le nombre de pixel que doit parcourir la tortue.)
Lors de l'exécution de la fonction, paramètre est remplacé par sa valeur.

On peut donner plusieurs paramètres à une fonction. Pour cela, on les sépare par une virgule.

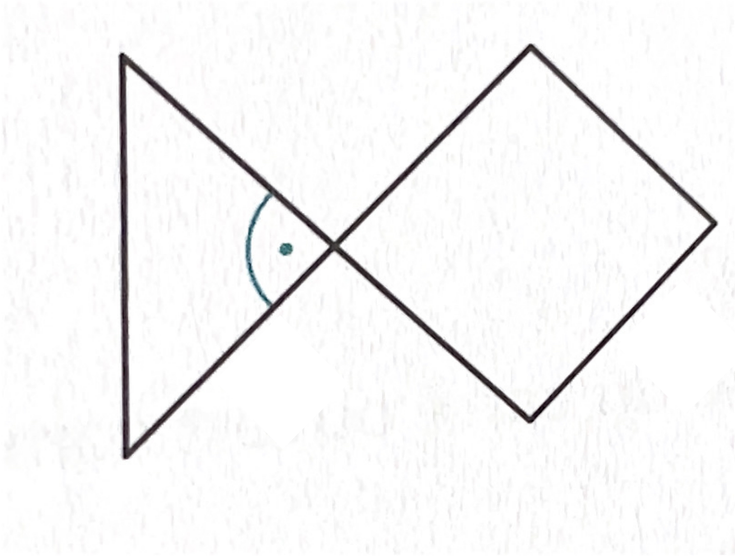
Exercice 1:

Créez une fonction permettant de créer un carré ayant une taille fournie en paramètre.



Exercice 2:

- Créez une fonction qui prend deux paramètres (La taille de la queue et la taille du carré) et qui permet de de dessiner un poisson comme ci-dessous.



Les variables

Où comment stocker des données et les manipuler pendant l’exécution de notre code.

Les variables : le grand principe.

Les variables sont des éléments qui associent un nom à une valeur, qui sera implantée dans la mémoire du système programmé. Une variable contient une valeur qui peut varier au cours de l'exécution du programme.

On déclare une variable ainsi :

```
let nomDeMaVariable;
```

Pour assigner une valeur à cette variable, on utilise le symbole « = »

```
nomDeMaVariable = 5;
```

Pour en savoir plus : cliquez ici

Les variables : Les types de données

Les variables peuvent contenir différent types de données. Les principales données que peuvent contenir une variable sont :

- les nombres
- les chaines de caractères
- les booléens
- les objets (ex : tableaux)

L'utilisation de ces différents types de données peut-être vu ici (pour les types de bases) et ici (pour les tableaux et autres objets).

Exécuter du code sous certaines conditions

Choisissez la condition appropriée pour contrôler le déroulement de votre programme (if, else, switch)

Répéter des opérations : les boucles !

En programmation, il y a des ensembles d'instructions à répéter plusieurs fois. Parfois, vous connaîtrez à l'avance le nombre de répétitions, d'autres fois non. Pour en savoir plus, ça se passe ici

Exercice 1:

Créez une fonction qui prend comme argument un nombre « n » et qui affiche un triangle de base « n » dans la console.

Exemple avec n = 7

Exercice 2:

Créez un programme qui affiche dans la console tous les nombres de 1 à 100 avec deux exceptions : les nombres divisibles par 3, doivent être remplacés par « Fizz » et ceux divisibles par 5 doivent être remplacés par « Fuzz ».

La programmation modulaire : suite n°2

Maintenant que nous avons vu les variables : voyons de nouvelles fonctionnalités des fonctions.

Les fonctions : la valeur de retour

```
function addition(nb_a, nb_b) {
  return nb_a + nb_b;
}

console.log(addition(addition(5, 10), 15));
//Affiche 30
```

Exercice 1:

Créez une fonction qui prend pour paramètres deux nombres (nb_a et nb_b) et qui renvoie *true* si nb_a > nb_b et *false* sinon.

Exercice 2:

Créez une fonction qui prend comme paramètre un tableau de nombre, et renvoie la somme de tous les nombres de celui-ci.