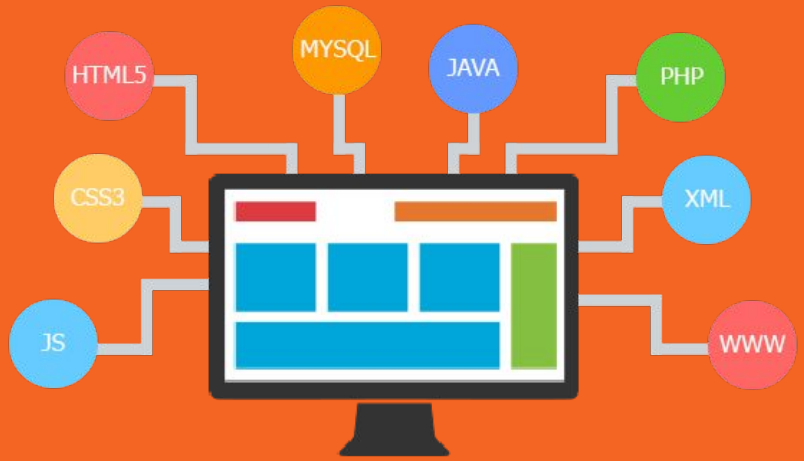
Développement web FULLSTACK JS



INTRODUCTION

Site web

Un site web est l'**ensemble des pages web** et des ressources inter reliées entre elles par des liens hypertextes, auxquelles l'internaute peut accéder par une adresse web appelée Url, le tout enregistré sous le même nom de domaine.

Les types de sites web :

- Les sites e-commerce
- Les sites vitrines
- Les sites institutionnels
- Les sites portails
- Les sites personnels

Application web

Une application web désigne un **logiciel applicatif hébergé** sur un serveur et accessible via un navigateur web.

Quelques exemples:

- Une application de gestion de réservation pour un hôtel
- Une application de facturation en ligne pour une entreprise ou un commerçant
- Une application de gestion de dossiers patients pour un médecin

FRONT-END (côté client)

Le terme « frontend » désigne les éléments d’un site que l’on voit à l’écran et avec lesquels on peut interagir depuis un navigateur. Il s’agit notamment de polices, de menus déroulants, de boutons, de transitions, de curseurs, de formulaires de contact, etc.

BACK-END (côté serveur)

Le Back-End, c’est la partie du code qui est exécutée par le serveur, il s'agit du travail qu’il réalise sur les pages Web des sites dynamiques avant de les envoyer au client.

Le Backend se compose généralement de trois éléments:

- Un serveur
- Une application web
- Une base de données

FRAMEWORK

Un ensemble de composants bien structurés qui sert à créer les bases et organiser le code.

Facilite le travail des développeurs

Deux types de framework :

- Côté client : Frontend

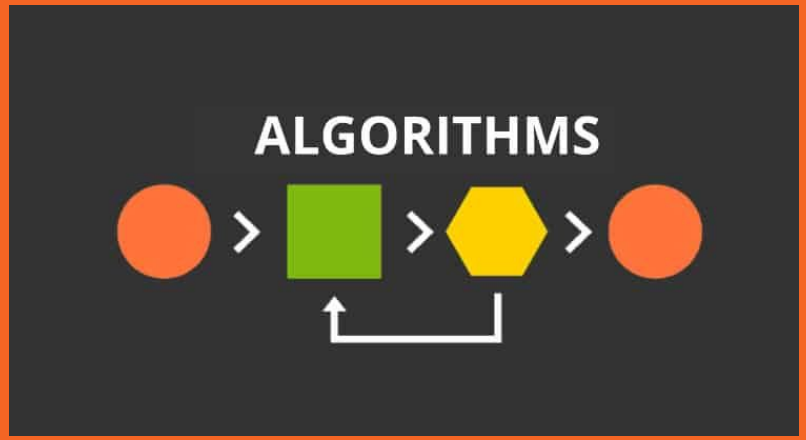
- Côté serveur : Backend

BIBLIOTHÈQUE

est un ensemble de **fonctions utilitaires**, regroupées et mises à disposition afin de pouvoir être utilisées sans avoir à les réécrire.

Langages de programmation	Framework Front-End	Framework Back-End
JavaScript	**Angular** React JS Vue JS Ember Meteor Backbone	**Node JS** **Express JS** Nest JS
Ruby		Ruby On Rails
C#		.Net / Core 3
Java		Spring Boot
PHP		Laravel Symfony

ALGORITHME



DÉFINITION ALGORITHME

Un algorithme est composé d'instructions et d'opérations réalisées, dans un ordre précis, sur des données afin de produire un résultat, et souvent résoudre un problème plus ou moins complexe.

Un algorithme prend des **données en entrée**, exprime un traitement particulier et fournit des **données en sortie**.

DÉCLARATION DES VARIABLES

Variable <nom de donnée> : **type**

• Instruction permettant de réserver de l'espace mémoire pour stocker des données.

• Type des données :

- entier (x=24)
- réel (x=2.15)
- chaîne de caractères (x="CrocoCoder")
- booléen (x=vrai)
- Tableau (T=[12,"Croco",2.14,vrai])

DÉCLARATION DES CONSTANTES

Constante <nom de donnée> : **type ← valeur ou expression**

• Instruction permettant de réserver de l'espace mémoire pour stocker une constante dont la valeur ne varie pas.

• **Exemples :**

• Constante MAX : entier ← 10 DEUXFOISMAX : entier ← MAX x 2

EXERCICE 1 (Enoncé)

Ecrire un programme qui lit le **prix HT d'un article**, le **nombre d'articles** et le **taux de TVA**, et qui fournit le **prix total TTC** correspondant.

EXERCICE 1 (Corrigé)

Variables prix HT, TVA, prix TTC numériques NbAr

Début

Ecrire (“entrez le prix HT de l’article”)

Lire (HT)

Ecrire (“entrez le TVA”)

Lire (TVA)

Ecrire (“donnez le nombre de l’article”)

Lire (NbAr)

Prix TTC <-- NbAr * prix HT * (1+TVA)

Ecrire (“le prix TTC de l’article est : * prix TTC”)

Fin

EXERCICE 2 (Énoncé)

Ecrire un programme qui demande un **nombre** à l'utilisateur, puis qui calcule et affiche le **carré de ce nombre**

EXERCICE 2 (Corrigé)

Variable nb, carré numérique

Début

Ecrire (“entrez un nombre lire nb”)

Carré ← nb * nb

Ecrire (“le carré de nb est :”, carré)

EXERCICE 3 (Énoncé)

Ecrire un programme qui demande le **rayon** d'un cercle, puis qui calcule et affiche le **périmètre.**

EXERCICE 3 (Corrigé)

```
Algorithme périmètre_cercle
Constante PI = 3.1415
Variable rayon: Réel
         périmètre: Réel
Début
Écrire("Rayon = ")
 Lire(rayon)
périmètre <- 2 * PI * rayon
Écrire("Le périmètre est : ", périmètre)
Fin
```

EXERCICE 4 (Énoncé)

Ecrire un programme qui permet d’échanger deux variables données (permutation)

EXERCICE 4 (Corrigé)

Algorithme permutation

Variable A: entier

B: entier

C : entier

Début

Écrire("Donnez deux entiers ")

Lire(A)

Lire (B)

C ← A

A ← B

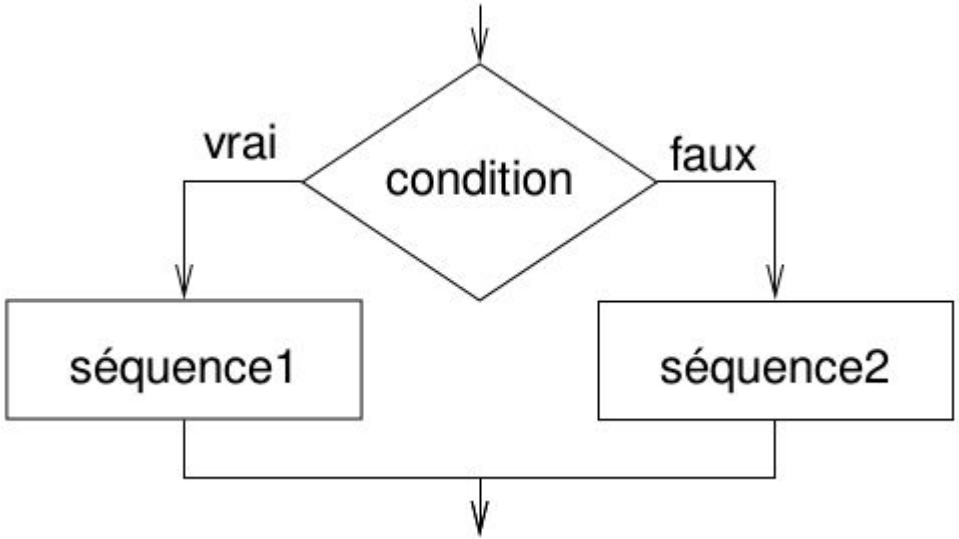
B ← C

Écrire(A,B)

Fin

STRUCTURES CONDITIONNELLES

On appelle structure conditionnelle les instructions qui permettent de tester si une condition est vraie ou non.



STRUCTURE ALTERNATIVE Si … Sinon

```
Si condition alors action1
Sinon action2
```

EXERCICE 5 (Énoncé)

Ecrire un algorithme qui demande un nombre à l'utilisateur, et l'informe ensuite si ce nombre est **positif** ou **négatif**

EXERCICE 5 (Corrigé)

Algorithme signe

Variable NB: entier

Début

Écrire("Donnez un entier")

Lire(NB)

Si NB >=0 alors

Ecrire (“le nombre que vous avez entré est positif”)

Sinon

Ecrire (“le nombre que vous avez entré est négatif”)

Fin si

Fin

EXERCICE 6 (Énoncé)

Ecrire un algorithme qui demande deux nombres à l'utilisateur, et l'informe ensuite si leur produit est négatif ou positif (Attention toutefois : on ne doit pas calculer le produit des deux nombres)

EXERCICE 6 (Corrigé)

```
Algorithme signe_produit
Variable x,y: entier
Début
Écrire("Donnez 2 entiers")
 Lire(x,y)
Si (x>0 et y>0) ou (x<0 et y<0)Alors
écrire (“le produit de x et y est positif”)
Sinon
écrire (“le produit de x et y est négatif”)
Finsi
Fin
```

EXERCICE 7 (Énoncé)

Ecrire un algorithme qui demande une valeur entière et affiche son double si cette donnée est inférieure à un seuil donné

EXERCICE 7 (Corrigé)

```
Algorithme simpleOudouble

constante (SEUIL : entier) ←10

Variable val : entier

Début

Ecrire("Donnez un entier : ")

Lire(val)

si (val < SEUIL) alors

Écrire ("Voici son double :" , val ×2)

sinon

Ecrire("Voici la valeur inchangée :" , val)

Fin si

Fin
```

EXERCICE 8 (Énoncé)

Ecrire un algorithme qui demande une année à l'utilisateur, et l'informe ensuite si cette année est bissextile ou non

1. L'année doit être divisible par 4 et non divisible par 100

2. L'année doit être divisible par 400

EXERCICE 8 (Corrigé)

Algorithme Année_Bixesstile

Variable A: entier

Début

Écrire("Donnez une année")

Lire(A)

si (A mod 4=0) et ((A mod 100>0) ou (A mod 400=0)) alors

écrire ("L'année ", A, "est bissextile.")

sinon

écrire ("L'année ", A, "n'est pas bissextile.")

Finsi

Fin

STRUCTURE ALTERNATIVE IMBRIQUÉE

Si condition1 alors action1
Sinon si condition2 alors action2

Si condition1 alors action1
Sinon si condition2 alors action2
Sinon action3

EXERCICE 10 (Énoncé)

Ecrire un algorithme qui demande une note à l'utilisateur.

Ensuite, il l'informe de sa mention:

"Insuffisant" en dessous de 10

"Passable" de 10 à 11

"Assez bien" de 12 à 13

"Bien" de 14 à 15

"Très bien" de 16 à 20

EXERCICE 10 (Corrigé)

```
Algorithme Mention
Var : note: réél
Début
 Ecrire (“Entrez votre note”)
Lire(note)
Si (note < 0 ou note > 20) Alors
 Ecrire (“Hors intervalle”)
 Sinon Si (age < 10) Alors
Ecrire (“Insuffisant”)
Sinon Si (age < 12) Alors
Ecrire (“Passable”)
Sinon Si (age < 14) Alors
Ecrire (“Assez bien”)
Sinon Si (age < 16) Alors
Ecrire (“Bien”)
Sinon
Ecrire (“Très bien”)
Fin Si
Fin
```

STRUCTURE SELON

```
Selon Cas <variable>
    Cas <expression>
            <action1>
    Cas <expression>
            <action2>
        ...
    Cas Sinon
            <action3>
Fin selon
```

EXERCICE 11 (Énoncé)

Ecrire un algorithme permettant d'afficher la saison en introduisant le numéro du mois.

EXERCICE 11 (Corrigé)

```
Algorithme Saison
 Var Mois :entier ;
Début
Ecrire(‘Donner un numéro de mois 1--12’)
Lire(Mois)
Selon Mois faire
3,4,5 : Ecrire(‘La saison est : PRINTEMPS’)
 6,7,8 : Ecrire(‘La saison est : ETE’)
9,10,11 : Ecrire(‘La saison est : AUTOMNE’)
12,1,2 : Ecrire(‘La saison est : HIVER’) ;
Fin selon
Fin
```