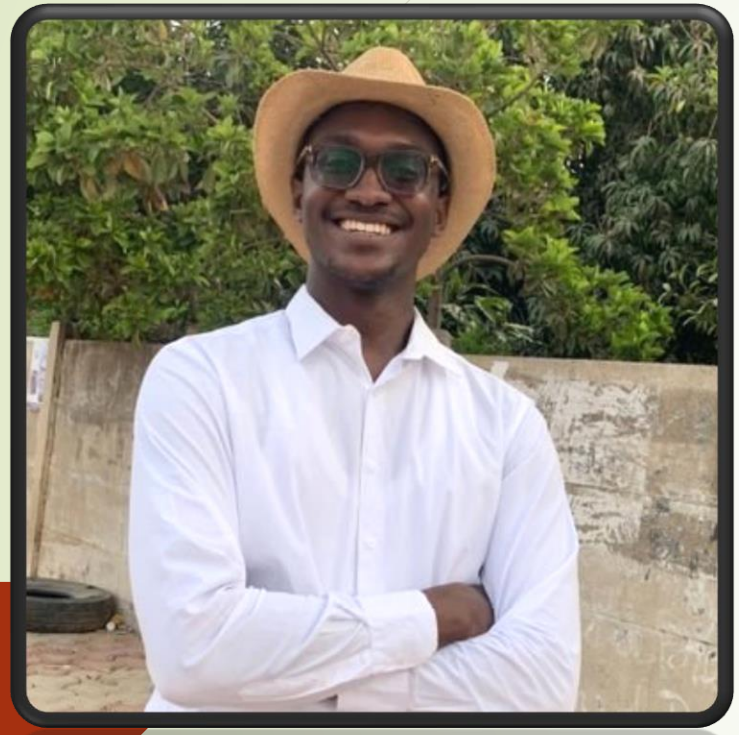
Présentation

Thème : Introduction à Java

Par : Médoune Siby Georges Baldé
Communauté GalsenDev



Présentateur

Médoune Siby Georges Baldé

Ingénieur Logiciel
Super Hero

Partner Manager
GalsenDEV

Questions ?! 

Syllabus :

1. Qu'allons nous voir dans cette présentation ?
2. Qu'est ce que Java ?
3. Qu'est ce qu'un langage de programmation orienté objet ?
4. Les Fondamentaux de la programmation en Java.
    a. Les Variables
    b. Opération arithmétiques
    c. Les structures de contrôle
5. Live time.
6. Questions.
7. Ressources Java
9. Conclusion.

Qu'allons nous voir dans cette présentation ?

Dans cette présentation nous allons parler du meilleur des langages de programmation :

« Java »

Qu’est ce que Java ?

Java est **un langage de programmation** utilisé pour le développement d'applications mobiles, de logiciels, de sites web et bien plus encore.

Il a été créé pour être portable, ce qui signifie qu'il peut fonctionner sur différentes plates-formes et systèmes d'exploitation.

Il est réputé pour sa fiabilité, sa sécurité et sa facilité d'utilisation. Java est un langage de programmation orienté objet.

Qu’est ce qu’un langage de programmation orienté objet ?

Un langage de programmation est un ensemble de règles et de symboles utilisés pour écrire des instructions comprises par un ordinateur.

L'orienté objet est une approche de programmation qui organise le code autour d'objets, qui regroupent des données et des fonctionnalités. Cela permet de créer des programmes modulaires et faciles à maintenir.

Les Fondamentaux de la programmation en Java.

Les fondamentaux de la programmation en Java sont les concepts de base que tout développeur Java devrait connaître. Ces concepts fournissent les bases nécessaires pour écrire du code Java fonctionnel et efficace.

a. Les Variables

LA DÉCLARATION D'UNE VARIABLE CONSISTE À CONSERVER DES DONNÉES LORS DE L'EXÉCUTION D'UN PROGRAMME EN SPÉCIFIANT LEUR TYPE ET À LEUR DONNER UN NOM QUI SERVIRA À LES IDENTIFIER.

a. Les Variables (suite)

Parmi les variables couramment utilisés nous pouvons citer :

int : UTILISÉ POUR STOCKER DES ENTIERS (NOMBRES ENTIERS).

double : UTILISÉ POUR STOCKER DES NOMBRES À VIRGULE FLOTTANTE (PRÉCISION ÉLEVÉE)

float : UTILISÉ POUR STOCKER DES NOMBRES À VIRGULE FLOTTANTE

boolean : UTILISÉ POUR STOCKER DES VALEURS BOOLÉENNES (VRAI OU FAUX).

String : UTILISÉ POUR STOCKER DES CHAÎNES DE CARACTÈRES.

char : UTILISÉ POUR STOCKER UN SEUL CARACTÈRE UNICODE SUR 16 BITS.

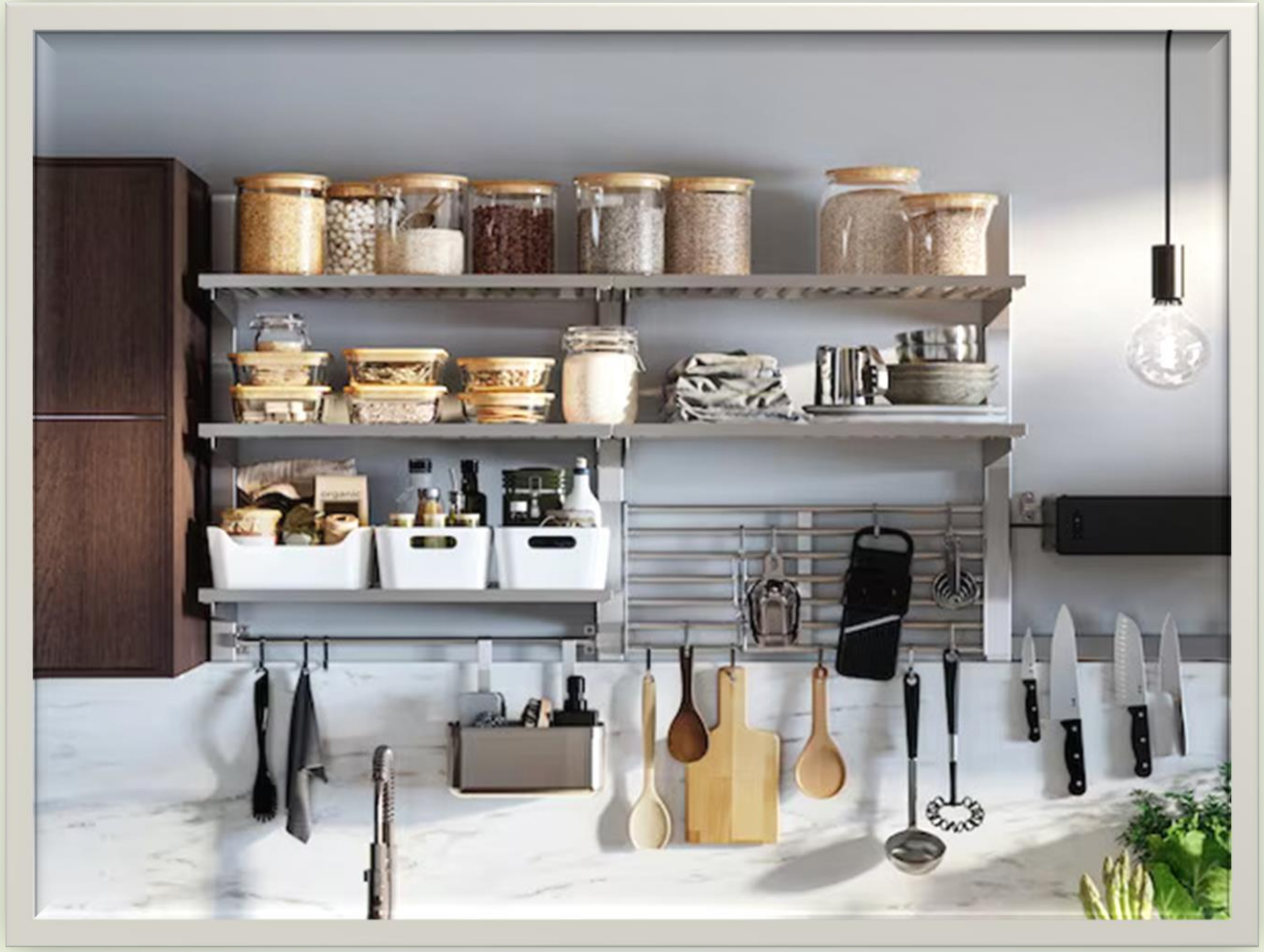
byte : UTILISÉ POUR STOCKER DE PETITS ENTIERS SIGNÉS SUR 8 BITS.

long : UTILISÉ POUR STOCKER DE GRANDS ENTIERS SIGNÉS SUR 64 BITS.

PAR EXEMPLE : VOUS POUVEZ DÉCLARER UNE VARIABLE ENTIÈRE

EN UTILISANT : int SCORE = 3;

Visualisation des variables



b. Opération arithmétiques

ADDITION, SOUSTRACTION, MULTIPLICATION ET DIVISION : JAVA NOUS PERMET D'EFFECTUER DES OPÉRATIONS ARITHMÉTIQUES DE BASE EN UTILISANT LES OPÉRATEURS +, -, *, / et %.

OPÉRATEURS DE COMPARAISON : JAVA PROPOSE DES OPÉRATEURS DE COMPARAISON TELS QUE ==, !=, <, >, <= ET >= POUR COMPARER DES VALEURS. CES OPÉRATEURS RENVOIENT UNE VALEUR booléenne : (true OU false).

OPÉRATEURS D'INCRÉMENTATION ET DE DÉCRÉMENTATION : LES OPÉRATEURS ++ ET -- NOUS PERMETTENT D'INCRÉMENTER ET DE DÉCRÉMENTER UNE VARIABLE RESPECTIVEMENT.

Visualisation des Opération arithmétiques

```java
int a = 5;
int b = 3;

// Addition
int sum = a + b;
System.out.println("Addition : " + sum);

// Soustraction
int difference = a - b;
System.out.println("Soustraction : " + difference);

// Multiplication
int product = a * b;
System.out.println("Multiplication : " + product);

// Division
int quotient = a / b;
System.out.println("Division : " + quotient);

// Modulo (reste de la division)
int remainder = a % b;
System.out.println("Modulo : " + remainder);
```

Visualisation des Opération arithmétiques (suite)

```java
int a = 5;
int b = 3;

// Opérateurs de comparaison
boolean isEqual = (a == b);
boolean isGreater = (a > b);
boolean isLess = (a < b);

System.out.println("a == b : " + isEqual); // Vérifie si a est égal à b
System.out.println("a > b : " + isGreater); // Vérifie si a est strictement supérieur à b
System.out.println("a < b : " + isLess); // Vérifie si a est strictement inférieur à b

// Opérateurs d'incrémentation et de décrémentation
int c = 10;

c++; // Incrémentation de c
System.out.println("Après l'incrémentation : " + c);

c--; // Décrémentation de c
System.out.println("Après la décrémentation : " + c);
```

c. Les structures de contrôle

LES STRUCTURES DE CONTRÔLE SONT DES ÉLÉMENTS ESSENTIELS DE LA PROGRAMMATION EN JAVA. ELLES PERMETTENT DE CONTRÔLER LE FLUX D'EXÉCUTION D'UN PROGRAMME ET DE PRENDRE DES DÉCISIONS CONDITIONNELLES.

LES BOUCLES : LES BOUCLES NOUS PERMETTENT DE RÉPÉTER L'EXÉCUTION D'UN BLOC DE CODE PLUSIEURS FOIS. EN JAVA, NOUS AVONS PLUSIEURS TYPES DE BOUCLES, TELS QUE :

FOR : EFFECTUE UN NOMBRE SPÉCIFIÉ D'ITÉRATIONS AVEC UNE SYNTAXE STRUCTURÉE.

WHILE : RÉPÈTE UN BLOC DE CODE TANT QU'UNE CONDITION DONNÉE EST VRAIE.

DO-WHILE : RÉPÈTE UN BLOC DE CODE TANT QU'UNE CONDITION DONNÉE EST VRAIE, AVEC UNE ÉVALUATION DE LA CONDITION À LA FIN DE CHAQUE ITÉRATION.

c. Les structures de contrôle (suite)

LES STRUCTURES DE CONTRÔLE CONDITIONNELLES:
ELLES PERMETTENT DE PRENDRE DES DÉCISIONS ET D'EXÉCUTER DIFFÉRENTS BLOCS DE CODE EN FONCTION DU RÉSULTAT DE CES CONDITIONS. PARMI CELLES-CI ON PEUT CITER :

FOR : EFFECTUE UN NOMBRE SPÉCIFIÉ D'ITÉRATIONS AVEC UNE SYNTAXE STRUCTURÉE.

IF : EXÉCUTE UN BLOC DE CODE SI UNE CONDITION EST VRAIE.

IF-ELSE : EXÉCUTE UN BLOC DE CODE EN FONCTION DE LA VÉRACITÉ D'UNE CONDITION.

SWITCH : SÉLECTIONNE UN BLOC DE CODE EN FONCTION DE DIFFÉRENTES VALEURS D'UNE EXPRESSION

Visualisation des boucles

```
for (initialisation; condition; incrémentation) {
    // Bloc de code à exécuter
}

while (condition) {
    // Bloc de code à exécuter
    // La condition est évaluée avant chaque itération
}

do {
    // Bloc de code à exécuter
    // La condition est évaluée à la fin de chaque itération
} while (condition);
```

Visualisation des Conditions

```java
int nombre = 10;

if (nombre > 0) {
    System.out.println("Le nombre est positif");
} else {
    System.out.println("Le nombre est négatif ou nul");
}

int age = 20;

if (age >= 18) {
    System.out.println("Vous êtes majeur");
}

char grade = 'B';

switch (grade) {
    case 'A':
        System.out.println("Excellent !");
        break;
    case 'B':
        System.out.println("Bien !");
        break;
    case 'C':
        System.out.println("Pas mal !");
        break;
    default:
        System.out.println("Échec !");
}
```

Live Time ! 

Questions ?! 

Ressources Java

FORMATION VIDÉO YOUTUBE

https://youtu.be/_l4pJ7HCrl4

Amigoscode

https://www.youtube.com/@amigoscode

https://dev.java/learn/

https://www.w3schools.com/java/

Galsen DEV

https://www.youtube.com/@GalsenDev221

Conclusion

J'ESPÈRE QUE VOUS AVEZ APPRÉCIÉ LA PRÉSENTATION ET JE VOUS REMERCIE POUR VOTRE PRÉSENCE.

VIVEMENT UNE PROCHAINE SESSION POUR PRÉSENTER UN SUJET ENCORE PLUS AVANCÉ.

A BIENTÔT !

RETROUVEZ LE SLIDE ET LES INFOS SUR GITHUB :

https://github.com/medounesgb/presentation-icagi