WW

Java Doc Expliquée

Table des matières

WW

I) Généralités

Pour faire un commentaire sur une celle ligne en java, on utilise le // :

```
// Ceci est un commentaire
```

Pour faire un commentaire, sur plusieurs lignes, on utilise /*……*/ :

```
/*
* Ceci est
* un commentaire
* sur plusieurs lignes
*/
```

Pour faire un commentaire de javadoc, on utilise le symbole /**…..*/ :

```
/**
* Ceci est
* un exemple de javadoc
*/
```

On utilise cette javadoc(en réalité un super commentaire) à plusieurs endroits :

- Avant l'instanciation d'une classe, pour expliquer le rôle de la classe
- Avant la déclaration d'attributs, décrivant le but de cet attribut
- Avant la déclaration des méthodes, expliquant les paramètres, les sorties ainsi que le rôle de la classe

Mais nous reverrons chaque passage plus en détail plus tard.

II) L'utilisation de balise

Comme nous l'avons compris juste avant, l'intérêt de la java doc est de documenter notre classe et expliquer son utilité. Pour faciliter la description et la normalisation de cette documentation, nous pouvons (et DEVONS) utiliser les balises à nos dispositions. Il existe deux types de balises :

- « Block » placées à la fin du commentaire (une par ligne)
- « Inline » à mettre dans le texte du commentaire

Dans la partie suivante, nous allons donc étudier ces exemples.

WW

a) Balise Block

Comme nous avons vu précédemment, les balises de type block sont à placées de la manière suivante :

```
/**
 * @author William
 * @version 1.2
 * @since 1.0
 */
public double moyenne(int a, int b) {
    if (a < 0 || b < 0) throw new IllegalArgumentException();
    return (a + b) / 2.0;
}
```

Voici une liste non exhaustive, avec leur description, et si on les utilisera ou pas :

Nom	Description	Utilisée
@author	Auteur de la classe	✅
@version	Version du fichier / document	〰
@param	Indique le paramètre d'une méthode	✅
@return	Décrit ce que retourne la méthode	✅
@throws ou @exception	Documente les exceptions que la méthode peut lever	✅
@see	Lien vers une autre classe	〰
@since	Indique depuis quelle version cette fonctionnalité existe	〰
@deprecated	Marque comme obsolète	✅
@serial	Décrit une variable utilisée lors de la sérialisation	❌
@serialField	Utilisé avec des champs Serializable	❌
@serialData	Donne des infos sur les données sérialisées	❌
@inheritDoc	Hérite la documentation de la classe ou méthode parente	❌
@link	Insère un lien vers une autre classe ou méthode, en ligne	❌

WW

b) Balise Inline

Ces balises là se situe à l'intérieur des lignes de commentaires. Les plus courantes sont les suivantes :

Balise	Description	Utilisée
{@link …}	Crée un lien cliquable vers une classe/méthode	✅
{@code …}	Affiche du texte en monospace	✅
{@literal...}	Affiche sans interprétation HTML	❌
{@value...}	Affiche la valeur d'une constante	~~

III) Exemple

```java
/**
 * Représente un compte bancaire simple avec dépôt, retrait et
consultation du solde.
 * <p>
 * Exemple d'utilisation :
 * {@code CompteBancaire compte = new CompteBancaire("Jean", 1000);}
 * </p>
 *
 * @author William
 * @version 1.0
 * @since 2025
 */
public class CompteBancaire {

    /**
     * Le nom du titulaire du compte.
     */
    private String titulaire;

    /**
     * Le solde actuel du compte.
     */
    private double solde;

    /**
     * Constructeur d'un compte bancaire.
     *
     * @param titulaire le nom du titulaire du compte
     * @param soldeInitial le solde de départ
     */
    public CompteBancaire(String titulaire, double soldeInitial) {
        this.titulaire = titulaire;
        this.solde = soldeInitial;
    }
```

WW

```
        /**
         * Dépose un montant sur le compte.
         *
         * @param montant le montant à déposer (doit être positif)
         */
        public void deposer(double montant) {
            if (montant > 0) {
                solde += montant;
            }
        }

        /**
         * Retire un montant du compte.
         *
         * @param montant le montant à retirer (doit être positif et ≤
    solde)
         * @throws IllegalArgumentException si le montant est supérieur au
    solde
         */
        public void retirer(double montant) {
            if (montant > solde) {
                throw new IllegalArgumentException("Fonds insuffisants");
            }
            solde -= montant;
        }

        /**
         * Retourne le solde actuel du compte.
         *
         * @return le solde
         */
        public double getSolde() {
            return solde;
        }

        /**
         * Retourne le nom du titulaire du compte.
         *
         * @return le nom du titulaire
         */
        public String getTitulaire() {
            return titulaire;
        }
    }
```