Utiliser Javadoc

Ce document ne présente pas de manière exhaustive le fonctionnement de `javadoc`. Il présente néanmoins ce qu'il vous est demandé d'utiliser au minimum pour réaliser le projet. Pour plus d'informations, vous pouvez vous reporter à la documentation officielle sur le site de sun.

Javadoc est un outil de génération automatique de documentation. Pour utiliser cet outil, il faut :

- Utiliser un format spécial pour écrire les commentaires dans les fichiers sources.
- Utiliser la commande `javadoc` avec les fichiers sources.

Javadoc utilise un format de documentation par défaut en anglais sur les machines de l'UFR. Dans un soucis d'homogénéité il serait donc plus adéquat de rédiger des commentaires en anglais. Toutefois, nous préférons avoir un code bien commenté en français que mal commenté dans une langue qui vous est moins familière.

1 Format des commentaires

Les commentaires qui servent à générer la documentation doivent avoir la forme suivante :

```
/** texte en commentaire qui apparaitra dans la documentation */
```

C'est la double étoile qui permet de faire la différence avec des commentaires classiques. Un certain nombre de "tags" prédéfinis vous aideront à ajouter des informations à la documentation. Un tag est un mot-clé précédé du symbole `@`. Nous préconisons pour le projet l'utilisation des tags `@author`, `@return` et `@param` dont nous allons voir l'utilisation dans cette partie. Il n'est pas interdit d'utiliser d'autres tags afin de rendre votre documentation plus claire.

1.1 Documenter une classe

Pour documenter une classe publique, il faut faire précéder sa déclaration de commentaires au format Javadoc. Vous utiliserez le tag `@author` pour indiquer le nom du ou des auteurs de la classe, selon la manière dont vous avez organisé votre travail en binôme.

Exemple :

```
/**
* Cette classe decrit la figure geometrique carre definie
* par une abcisse, une ordonnee, une couleur, et la longueur
* de son cote.
*
* @author Jimmy Jazz
*/
public class Carre extends Figure{
```

1.2 Documenter un attribut

1. **Attributs *private* :** ceux-ci ne seront pas documentés à la manière javadoc, mais en commentaires standard (et c'est obligatoire).

   ```
   // longueur du cote d'un carre
   private int cote ;
   ```

2. **Attributs *public* et *protected* :** ils seront commentés au format javadoc. On rappelle au passage qu'attribuer la visibilité *public* à un attribut doit être justifié.

```
/**
* longueur du cote d'un carre
*/
public int cote ;
```

On rappelle qu'il est déconseillé d'utiliser la visibilité par défaut.

1.3 Documenter les constructeurs et les méthodes

Toutes les méthodes et les constructeurs seront documentés, quelle que soit leur visibilité.

Vous utiliserez le tag `@param` pour documenter chaque paramètre de la méthode ou du constructeur.

Vous utiliserez le tag `@return` pour documenter la valeur retournée par une méthode, si cette méthode ne retourne pas `void`.

Exemple : documenter un constructeur

```
/**
* Constructeur d'un carré étant donnés une abcisse et une ordonnée,
* une couleur, et une longueur de côté.
* @param abcisse coordonnée abcisse du coin superieur gauche.
* @param ordonnee coordonnée ordonnée du coin superieur gauche.
* @param couleur couleur du carre lorsqu'on le dessine.
* @param cote longueur du cote du carre.
*/
public Carre( int abcisse , int ordonnee , Color couleur , int cote ){
    super( abcisse , ordonnee , couleur ) ;
    this.cote = cote ;
}
```

Exemple : documenter une méthode

```
/**
* Retourne la description d'une instance de la classe Carre
* sous forme de chaine de caracteres.
* @return la description de l'instance.
*/
public String toString(){
    return (super.toString() + " " + cote) ;
}
```

2 La commande javadoc

C'est cette commande qui permet de générer la documentation sous format html de vos fichiers sources correctement documentées.

La commande peut être invoquée avec un nom de package, ou avec des noms de fichier sources.

Synopsis :

```
javadoc [ options ] packagename
javadoc [ options ] filenames
```

2.1 Options

Voici quelques options qui peuvent vous être utiles. Vous êtes invités à regarder dans la documentation officielle pour les autres options.

`-author` Indique que les commentaires tagés par `@author` devront être utilisées pour générer la documentation (par défaut, ces informations ne sont pas utilisées).

`-d repertoire` Permet de préciser le repertoire ou javadoc placera les fichiers HTML générés. Par défaut, la commande place tous les fichiers HTML dans le repertoire courant.

`-public` Ne documente que les membres *publics*.

`-private` Documente tous les membres, quelles que soient leur visibilité.

Par défaut, la documentation s'effectue sur les membres *publics* et *protected*.

2.2 Exemples d'utilisation

On suppose que vous disposez d'un répertoire **huffman** dans lequel vous avez créé trois répertoires :

- le répertoire **sources** contient les sources du projet.
- Le répertoire **docUtilisateur** contiendra la documentation de tous les membres publics.
- Le répertoire **docProjet** contiendra la documentation de tout votre projet (y compris celle des méthodes privées).

Une fois que vos sources sont correctement documentées, placez vous dans le répertoire huffman et exécutez les commandes suivantes :

```
javadoc -d docProjet -author -private sources/*.java
javadoc -d docUtilisateur -author sources/*.java
```

Ce qui créera les deux documentations du projet.

Il est également possible de créer la documentation javadoc à l'aide de Eclipse.