

5 - Documentation avec JavaDoc

Les commentaires en Java

- Le commentaire dans un code Java est de deux types :
- Le commentaire qui explique un détail d'implémentation

```
// ces lignes sont mises en commentaire
```

```
// et ne seront pas compilées
```

```
/* ces lignes sont mises en commentaire
```

```
et ne seront pas compilées non plus
```

```
etc... etc... etc... etc... etc... etc...
```

```
etc... etc... etc... etc... etc... etc... */
```

- Les commentaires de documentation embarquée JavaDoc. Commentaire qui sera extrait pour être transformé en page HTML.

```
/** ces lignes sont destinées à compléter le code d'une
```

```
* d'une documentation qui explique à quoi sert la classe,
```

```
* la méthode etc...
```

```
*/
```

Que documenter ? (3 niveaux)

- pour commenter le rôle général de la classe

```
/**
 * Le type Duree permet la manipulation ...
 * etc.
 */
public class Duree {
    ...
}
```

- pour commenter le rôle d'un attribut de la classe

```
public class Duree {

    /* La durée s'exprime en millisecondes
    */
    private long leTemps;

    ...
}
```

Que documenter ? (suite)

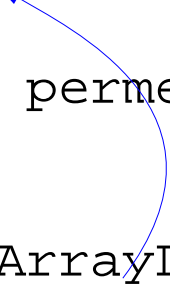
- pour commenter le rôle d'une méthode de la classe

```
public class Duree {  
    ...  
    /**  
     * Effectue une comparaison entre la  
     * durée courante et une autre durée.  
     * ...  
     */  
    public int compareA (Duree uneDuree) {  
        ...  
    }  
}
```

Comment documenter ?

!! Le commentaire JavaDoc doit se placer **juste avant** la classe, l'attribut ou la méthode.

```
/**  
 * Le type Duree permet la manipulation ...  
 * etc.  
 */  
import java.util.ArrayList;  
  
public class Machin {  
    . . .  
}
```



Erreur de placement de l'import

Documenter le rôle d'une classe

Pour documenter le rôle d'une classe, javaDoc reconnaît un certain nombre de balises (tags) qui lui sont propres.

Nous utiliserons au moins :

`@author` « l'auteur de la classe »

`@version` « numéro de version de la classe »

```
/**
 * Le type Duree permet la manipulation etc.
 *
 * @author Jacques Dupond
 * @version 1.1.0
 */
public class Duree {
    ...
}
```

Documenter le rôle d'une méthode

Les balises JavaDoc spécifiques aux méthodes sont :

`@param «nomParamètre» : «description du paramètre»`

`@return «description du type retourné»`

`@throws «type d'exception» «description du cas»`

Exemple de documentation d'une méthode

```
/**
```

```
 * Effectue une comparaison entre la durée courante  
 * et une autre durée.
```

```
 * @param autreDuree : durée à comparer à la  
   durée courante.
```

```
 * @return une valeur entière avec la signification  
   suivante
```

```
*/
```

```
public int compareA ( Duree autreDuree ) { ... }
```


Production des pages HTML

Pour générer les pages HTML de documentation d'une classe, taper la commande :

```
javadoc [options] chemin/MyClass.java
```

[options] :

- d cheminRépertoireHTML
- version
- author
- private
- ...

Exemple à partir d'un répertoire ww:

```
javadoc -private -author -d ../doc ../source/*.java
```

Format des pages HTML produites

1. La description de la classe

[Class](#) [Tree](#) [Deprecated](#) [Index](#) [Help](#)

PREV CLASS NEXT CLASS

SUMMARY: INNER | [FIELD](#) | [CONSTR](#) | [METHOD](#)

[FRAMES](#) [NO FRAMES](#)

DETAIL: [FIELD](#) | [CONSTR](#) | [METHOD](#)

Class Duree

java.lang.Object

|

+-**Duree**

```
public class Duree
extends java.lang.Object
```

Description

Le type "Duree" permet la manipulation d'intervalles de temps : conversion en JHMS, addition de durées, comparaison entre durées.

Format des pages HTML produites

2. La description résumée des attributs et méthodes

Field Summary

private	<u>leTemps</u>	
long		la durée s'exprime en millisecondes

Constructor Summary

[Duree](#) ([Duree](#) autreDuree)
Constructeur qui clone une Duree existante.

[Duree](#) (int heures, int minutes, int secondes)
Constructeur de Duree avec initialisation.

[Duree](#) (long millisec)
Constructeur de Duree avec initialisation.

Method Summary

void	<u>ajoute</u> (<u>Duree</u> uneDuree)	
		Modificateur qui ajoute une durée à la durée courante.

int	<u>compareA</u> (<u>Duree</u> autreDuree)	
		Effectue une comparaison entre la durée courante et une autre durée.

private	int[]	<u>enJHMS</u> ()	
			Effectue un découpage de la durée en intervalles (jours, heures, minutes, secondes, millisecondes).

java.lang.String	<u>enTexte</u> (char mode)		
			Renvoie une image textuelle de la durée courante.

long	<u>getLeTemps</u> ()		
			Accesseur qui retourne la valeur de la durée en millisecondes.

Format des pages HTML produites

3. La description détaillée des attributs et méthodes

Field Detail

leTemps

```
private long leTemps
```

la durée s'exprime en millisecondes

Constructor Detail

Duree

```
public Duree(Duree autreDuree)
```

Constructeur qui clone une Duree existante.

Parameters:

autreDuree - autre durée à copier

Duree

```
public Duree(int heures,  
             int minutes,  
             int secondes)
```

Constructeur de Duree avec initialisation.

Parameters:

heures - nombre d'heures

minutes - nombre de minutes

secondes - nombre de secondes

Initialisation d'une durée par un nombre d'heures, minutes et secondes.

Format des pages HTML produites

Method Detail

getLeTemps

```
public long getLeTemps()
```

Accesseur qui retourne la valeur de la durée en millisecondes.

Returns:

la valeur actuelle de la durée en millisecondes

compareA

```
public int compareA(Duree autreDuree)
```

Effectue une comparaison entre la durée courante et une autre durée.

Parameters:

autreDuree - durée à comparer à la durée courante

Returns:

Une valeur entière avec la signification suivante :

- -1 : si la durée courante est plus petite que l'argument
- 0 : si les deux durées sont égales
- +1 : si la durée courante est plus grande que l'argument