

## La commande javadoc

P21:  
La commande javadoc permet de générer la documentation d'un code source java en fichier HTML.

Exemple: `javadoc -private western -d Documentation`

La commande → pour montrer toutes les classes et membres (en fait tout, que ce soit public, private, protected et package)

→ le package que je veux documenter (¶ quand j'ai exécuté la commande, je me trouve dans le dossier src du projet)

→ l'option qui permet de dire dans quelle dossier je veux placer la documentation (sans cette option, il se place dans le dossier courant et c'est un peu le bordel)

On obtient la documentation de notre package en HTML. Si on veut être sur la page principale, il faut ouvrir le fichier: index.html.

Pour un commentaire de documentation en java: `/**  
 *  
 *  
 *  
 */`

Par défaut la commande javadoc utilise le docket standard pour générer la documentation en HTML. Mais on peut lui dire d'utiliser un docket à nous pour générer du PUML (avec l'option: `-docket`).

La commande pour exécuter notre docket.

`javadoc -private -sourcepath src -docket pumlFromJava.FirstDocket  
-docketpath out/production/P21.Projet western`

→ l'option pour indiquer les repertoires pour où trouver les packages

→ le chemin où trouver les fichiers .class (en absolu ou relatif)

→ l'option pour préciser que on utilise un docket à nous

→ le nom du dossier peut être différent (il faudra juste le changer par le bon si cela ne marche pas)

→ le package qui on veut analyser

→ où trouver le docket (celui où je me suis pas dans src mais juste au dessus, dans le dossier global)

(Pour l'instant le code de base fait juste différents System.out.println() il va falloir qui on fasse en sorte de créer un fichier PUML à un endroit donné)

Pour que dans l'IntelliJ, l'exécution de Java2Puml.java lance la commande javadoc et notre doctet avec ses bons arguments :

```
public class Java2Puml  
{  
    public static void main(String[] args)  
    {  
        CoolProvider toolProvider = CoolProvider.findFirst("javadoc").get();  
        System.out.println(toolProvider.name());  
        // Je me déclare un tableau de String avec des bons arguments.  
        String[] argument = new String[] {"-private", "-sourcepath", "src",  
            "-doctet", "pumlFromJava.FirstDoctet", "-doctetpath",  
            "out/production/P2Projet", "western"};  
        // Je donne à la méthode run mon tableau d'arguments à la place de  
        // toolProvider.run(System.out, System.err, argument);  
    }  
}
```