

Javadoc tool

Chonnam National University
School of Electronics and
Computer Engineering

Kyungbaek Kim

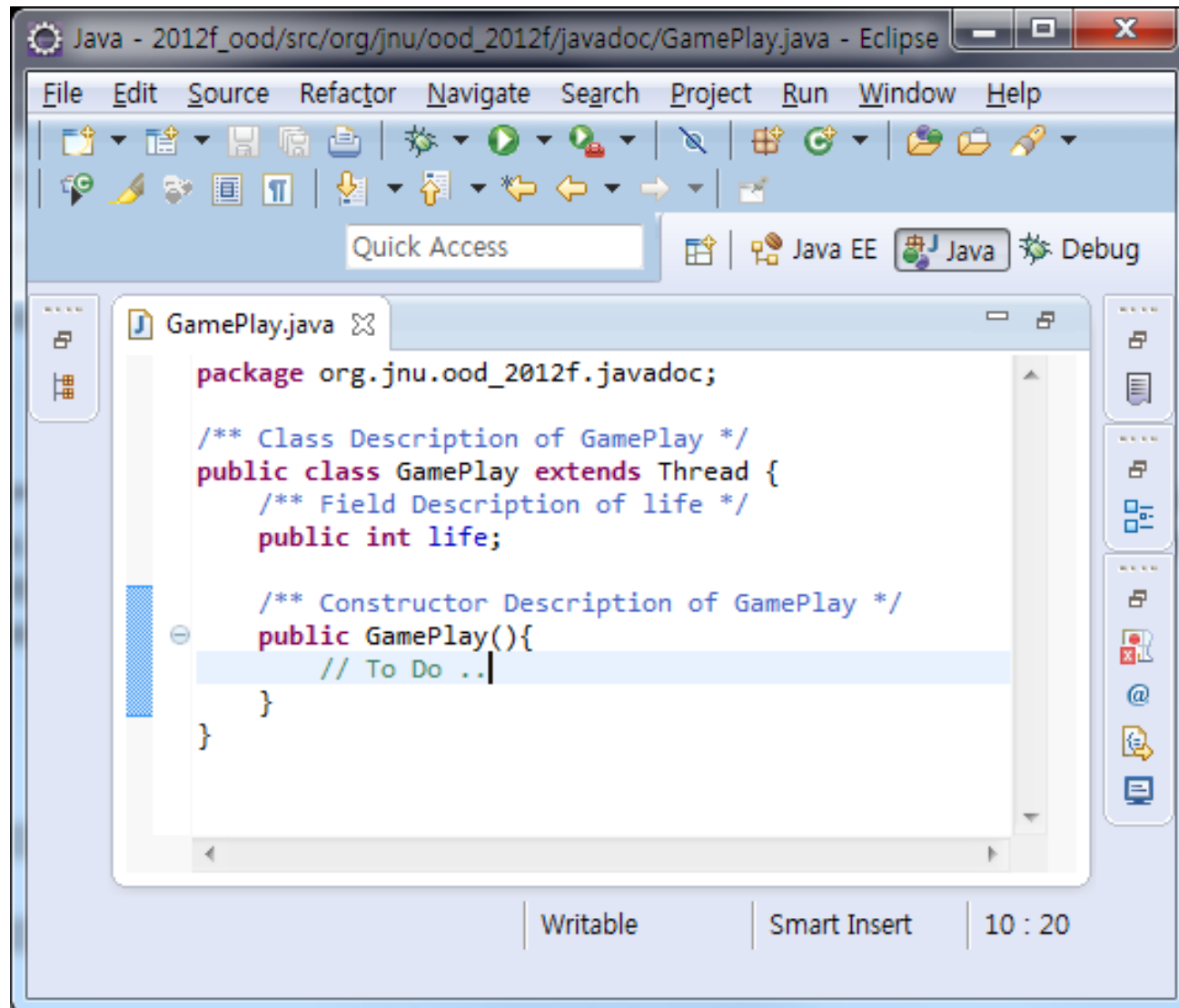
What is Javadoc?

- Javadoc
 - A tool for generating API documentation in HTML format from doc comments in source code
 - API should describe “all aspects of the behavior of each method on which a caller can rely.”
 - As part of the Java 2 SDK

Javadoc Comments

- Javadoc recognize special comments with `"/** */"`
 - Highlighted blue by default in Eclipse
 - c.f., regular comments (`"/* ... */"`, `"//"`) are highlighted green
- Comments are the descriptions to classes, constructors, fields, interfaces and methods

Example of Javadoc comment



General Format of Comments

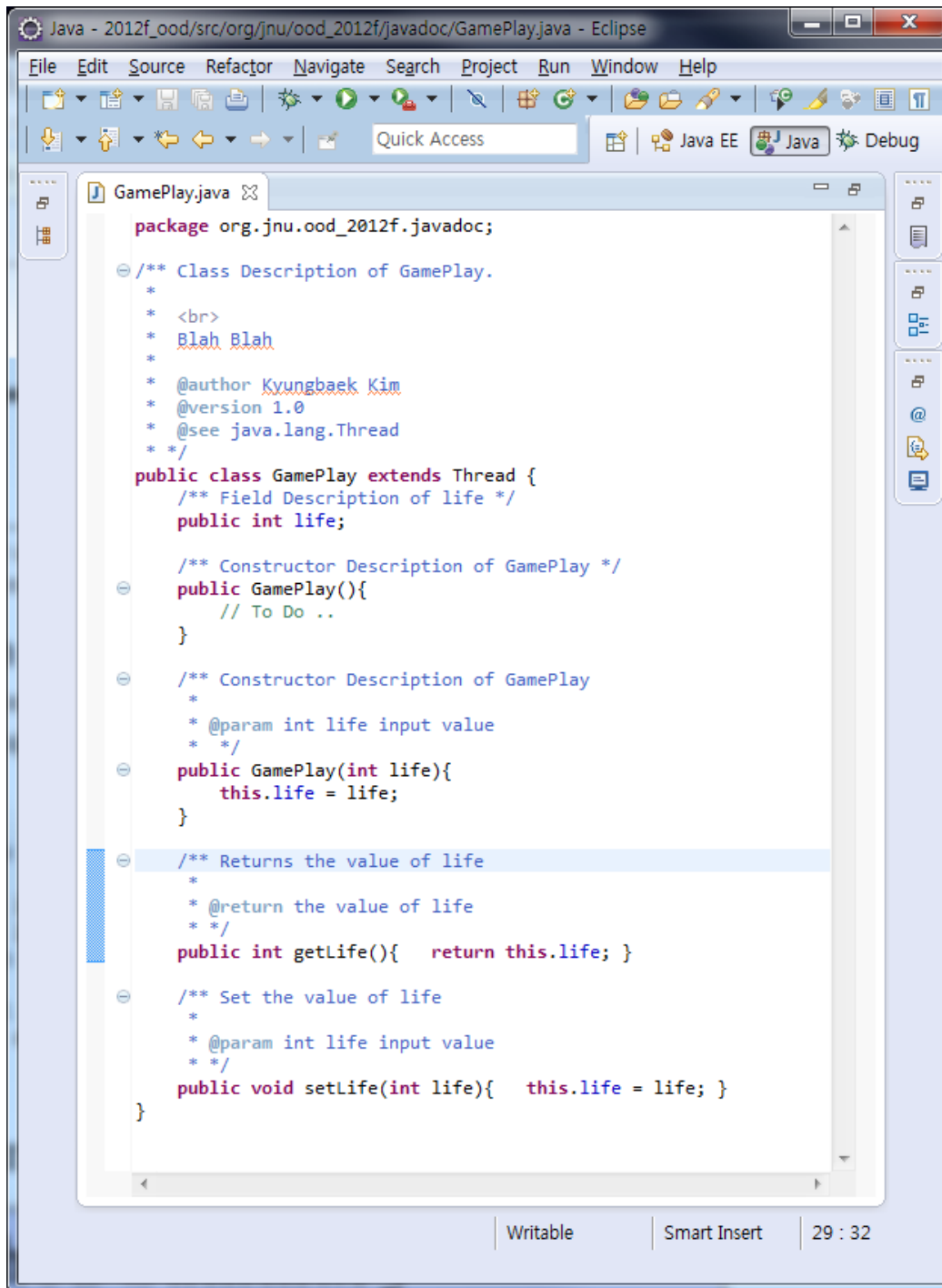
```
/**  
 * This is the description part of the doc comment.  
 * <br>  
 * Additional details  
 * <p> More Details </p>  
 *  
 * @tag1 Tag Content  
 * @tag2 Tag Content  
 *  
 */
```

- Tags are keywords recognized by javadoc
 - Define the type of information that follows
- Tags come after the description
 - Separated by a new line

Predefined tags

- *@author [author name]* - identifies author(s) of a class or interface.
- *@version [version]* - version info of a class or interface.
- *@param [argument name] [argument description]* - describes an argument of method or constructor.
- *@return [description of return]* - describes data returned by method (unnecessary for constructors and void methods).
- *@exception [exception thrown] [exception description]* - describes exception thrown by method.
- *@throws [exception thrown] [exception description]* - same as *@exception*.
- *@see [classname]* – add a hyperlink to a section
- *@since [since-text]* – Specify from which version of java this class, method or interface has been available
- *@deprecated* – indicate that this code is deprecated

Example of Tags



```
Java - 2012f_ood/src/org/jnu/ood_2012f/javadoc/GamePlay.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Java EE Java Debug

GamePlay.java
package org.jnu.ood_2012f.javadoc;

/** Class Description of Gameplay.
 *
 * <br>
 * Blah Blah
 *
 * @author Kyungbaek Kim
 * @version 1.0
 * @see java.lang.Thread
 */
public class Gameplay extends Thread {
    /** Field Description of life */
    public int life;

    /** Constructor Description of Gameplay */
    public Gameplay(){
        // To Do ..
    }

    /** Constructor Description of Gameplay
     *
     * @param int life input value
     */
    public Gameplay(int life){
        this.life = life;
    }

    /** Returns the value of life
     *
     * @return the value of life
     */
    public int getLife(){ return this.life; }

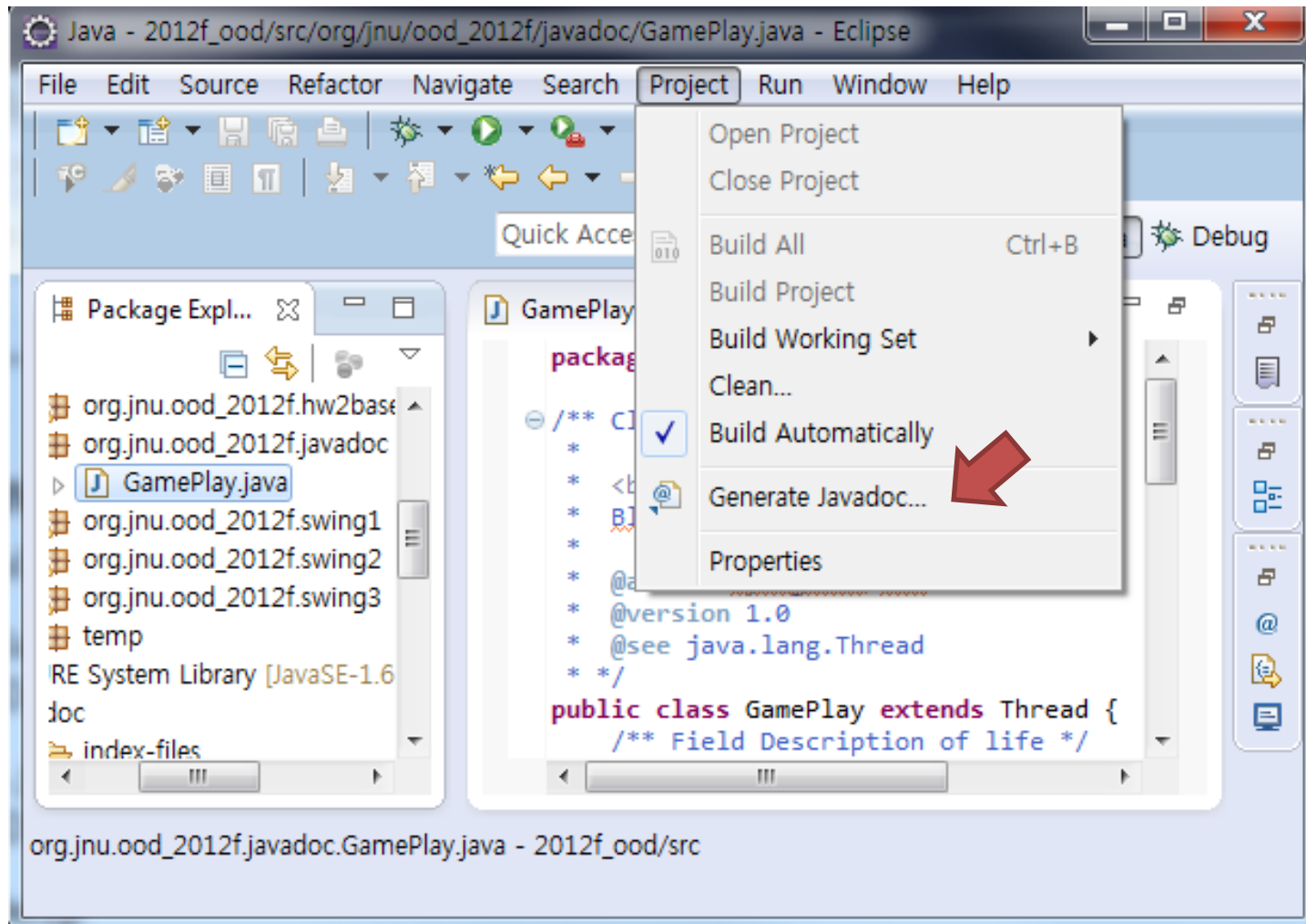
    /** Set the value of life
     *
     * @param int life input value
     */
    public void setLife(int life){ this.life = life; }
}
```

Writable | Smart Insert | 29 : 32

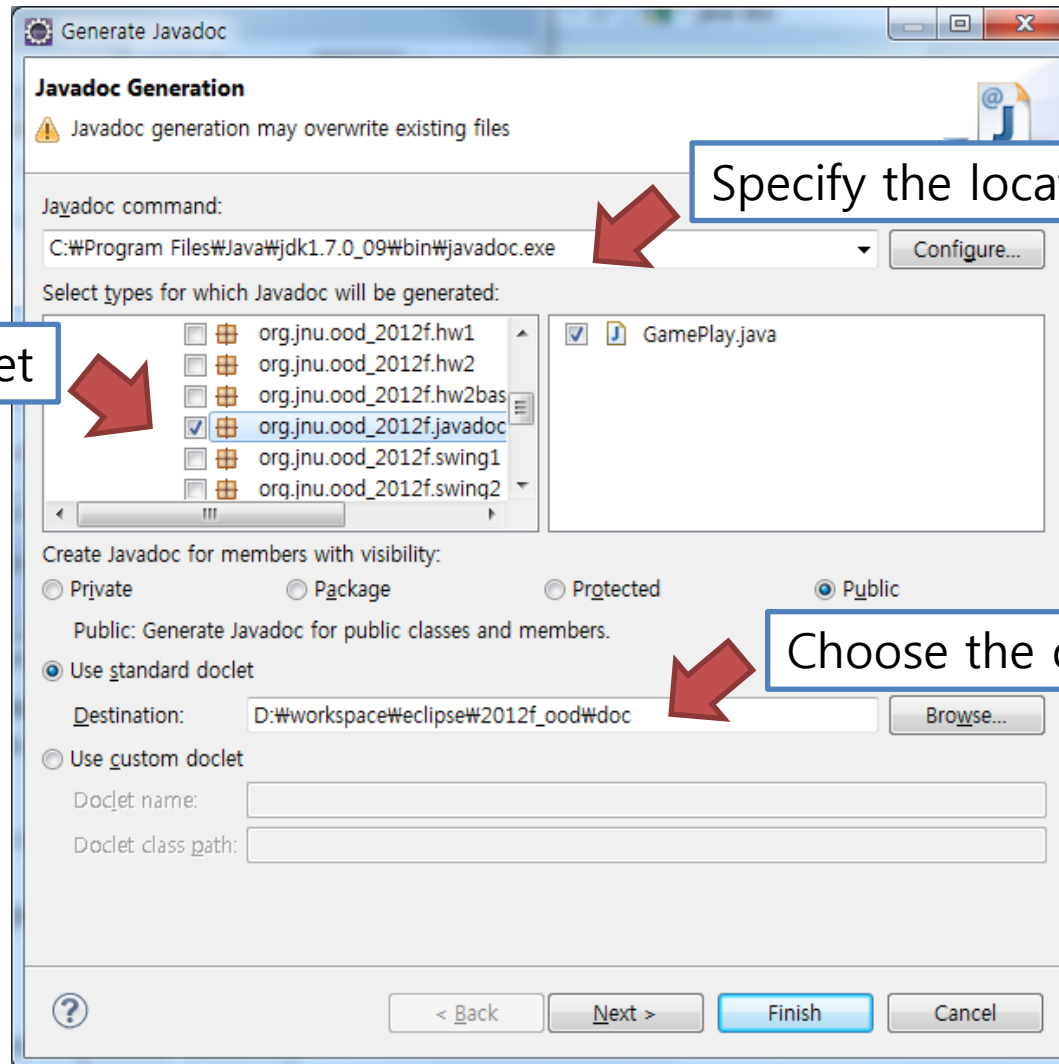
Producing the output

- `javadoc Gameplay.java`
 - Creates JavaDoc for the file `GamePlay.java`
- `javadoc *.java`
 - Create JavaDoc for all java files in the current directory
- Option
 - `-help` : displays the command line options
 - `-d` : direct generated JavaDoc to a specific directory

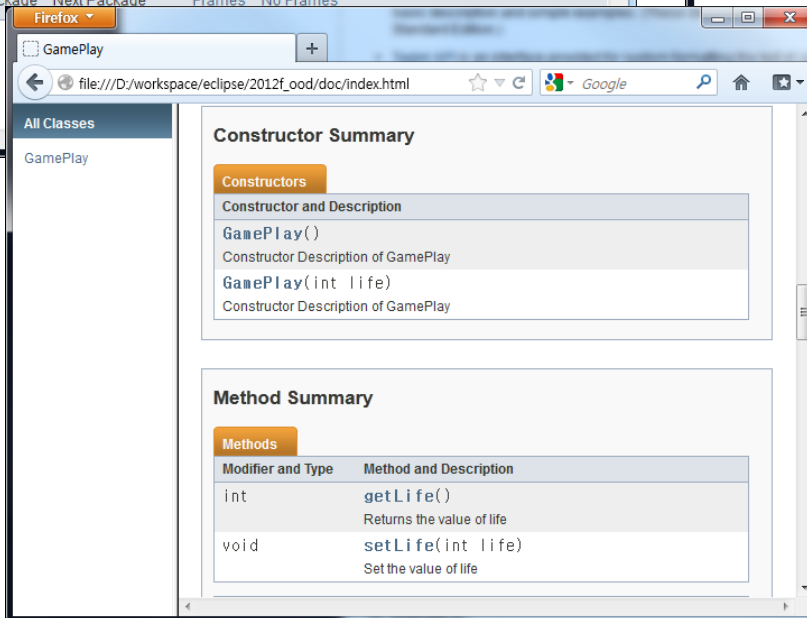
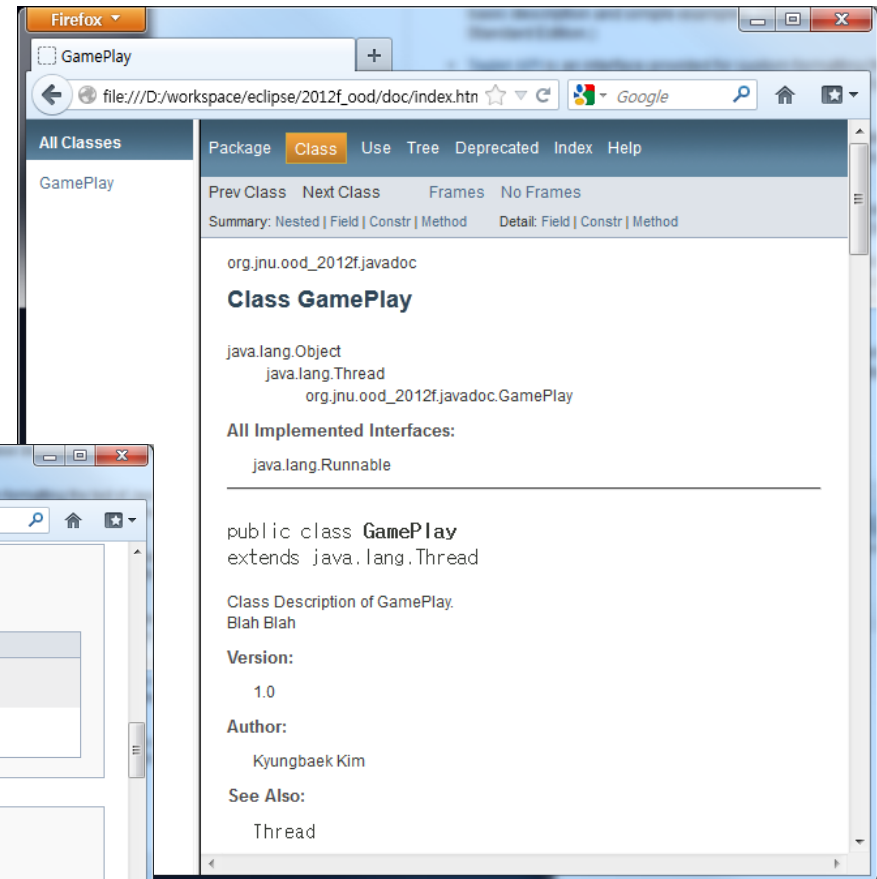
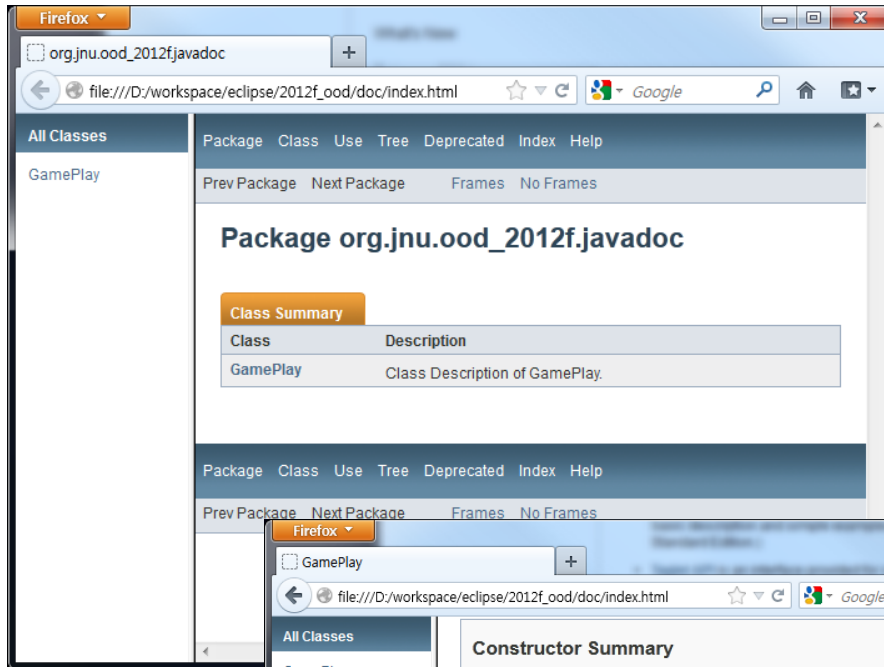
JavaDoc in Eclipse



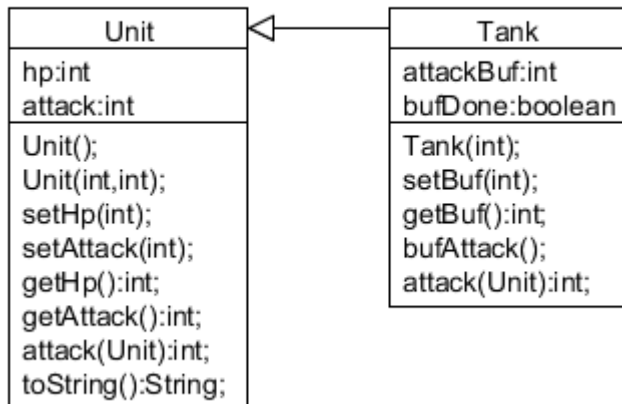
JavaDoc in Eclipse



Output of Javadoc



Test



Write codes related to the class diagram.



Then, write comments for classes, methods, and parameters.



Finally, generate the API documentation by using javadoc