Documentation & **`Javadoc`**

Automated Documentation Generation

Java API

- Check out the Java API
  - Application Program Interface

http://docs.oracle.com/javase/8/docs/api/

- Its like an instruction manual for different Java commands

is a representable number, the exact result should be returned as the computed result; otherwise, either of the two floating-point values which bracket the exact result may be returned. For exact results large in magnitude, one of the endpoints of the bracket may be infinite. Besides accuracy at individual arguments, maintaining proper relations between the method at different arguments is also important. Therefore, most methods with more than 0.5 ulp errors are required to be *semi-monotonic*: whenever the mathematical function is non-decreasing, so is the floating-point approximation, likewise, whenever the mathematical function is non-increasing, so is the floating-point approximation. Not all approximations that have 1 ulp accuracy will automatically meet the monotonicity requirements.

Since:

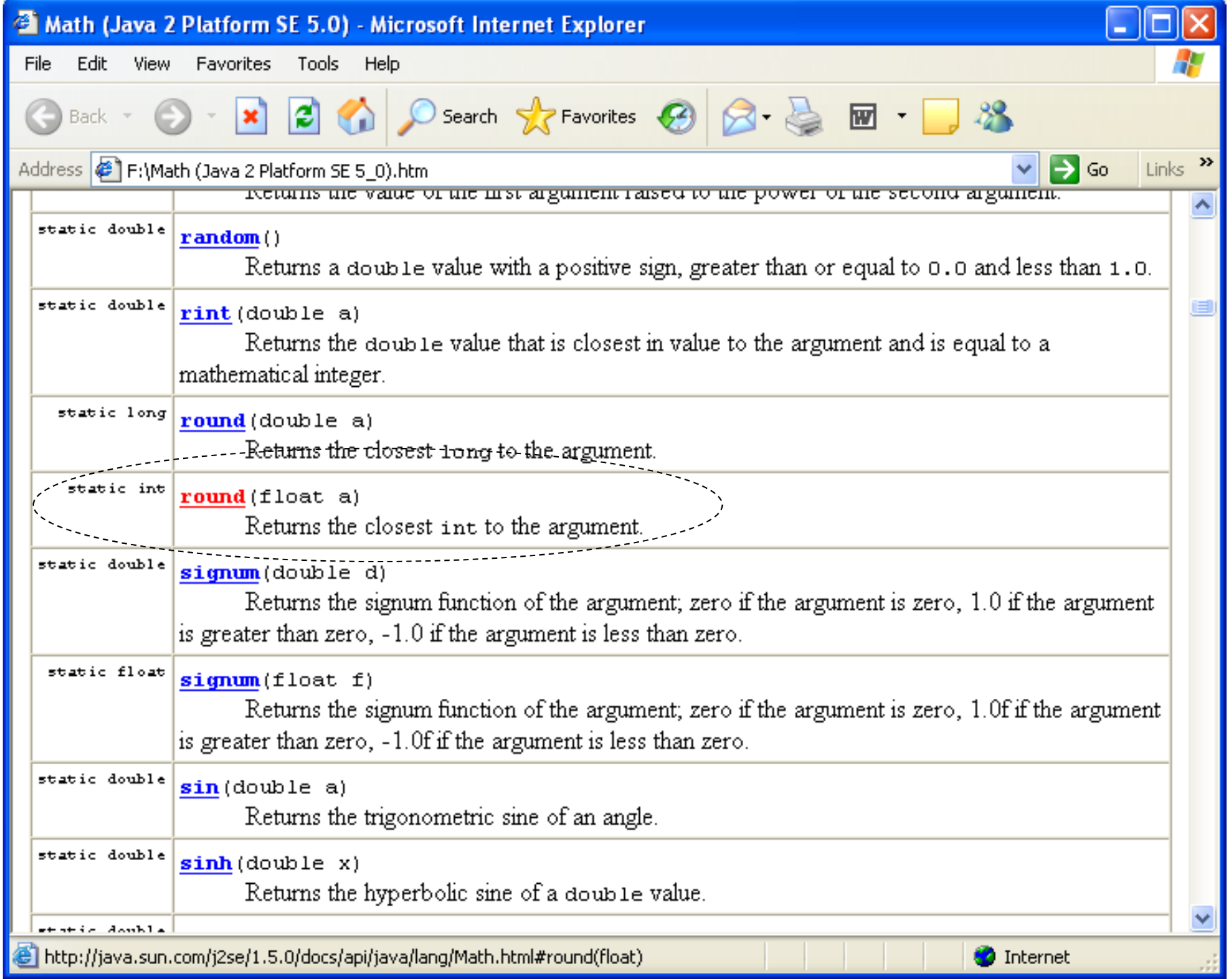
JDK1.0

Field Summary

static double	E The double value that is closer than any other to *e*, the base of the natural logarithms.
static double	PI The double value that is closer than any other to *pi*, the ratio of the circumference of a circle to its diameter.

Method Summary

static double	abs(double a) Returns the absolute value of a double value.
static float	abs(float a) Returns the absolute value of a float value.
static int	abs(int a) Returns the absolute value of an int value.
static long	abs(long a) Returns the absolute value of a long value.
static double	acos(double a) Returns the arc cosine of a value; the returned angle is in the range 0.0 through *pi*.
static double	asin(double a)


Math (Java 2 Platform SE 5.0) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address F:\Math (Java 2 Platform SE 5_0).htm
Go Links
Returns the value of the first argument raised to the power of the second argument.
static double random()
Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0.
static double rint(double a)
Returns the double value that is closest in value to the argument and is equal to a mathematical integer.
static long round(double a)
Returns the closest long to the argument.
static int round(float a)
Returns the closest int to the argument.
static double signum(double d)
Returns the signum function of the argument; zero if the argument is zero, 1.0 if the argument is greater than zero, -1.0 if the argument is less than zero.
static float signum(float f)
Returns the signum function of the argument; zero if the argument is zero, 1.0f if the argument is greater than zero, -1.0f if the argument is less than zero.
static double sin(double a)
Returns the trigonometric sine of an angle.
static double sinh(double x)
Returns the hyperbolic sine of a double value.
http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Math.html#round(float)
Internet

Self Documenting

- Programs should be self documenting
- Self documenting requires two things

1. Use meaningful names for variables, classes etc
   - Indentation
   - **Classes** begin with a capital letter and identifier names begin with a small letter.

2. Include comments where appropriate

Javadoc

- Programs are generally built using some pre-existing code
  - after all, why re-invent the wheel?
- Its important to describe your code for future users and for your future self
- `Javadoc` helps you to document your code properly and professionally

Javadoc on the command line

Future: Download the files from Day12 “Examples of Object Oriented inheritance” and store them in a directory on the **x: drive**. Then go to this directory in a DOS command shell and type the command:

```
javadoc –d  mydocuments TwoDShape.java
```

This command will create java documentation for the TwoDShape class in a newly created directory **mydocuments**.

Next open up the **index.html** file in the **mydocuments** directory in a browser to view the documentation.

JavaDoc Comments

- Comments should be included before the start of every class

- Comments should be included before the start of every method

BlueJ and Javadoc

- All Java programs can be documented using Javadoc
  - Tools/Project Documentation
- A new directory (`/docs`) is added inside your current project
- Browse the documentation with a web-browser (double-click index.html)

Documentation Example

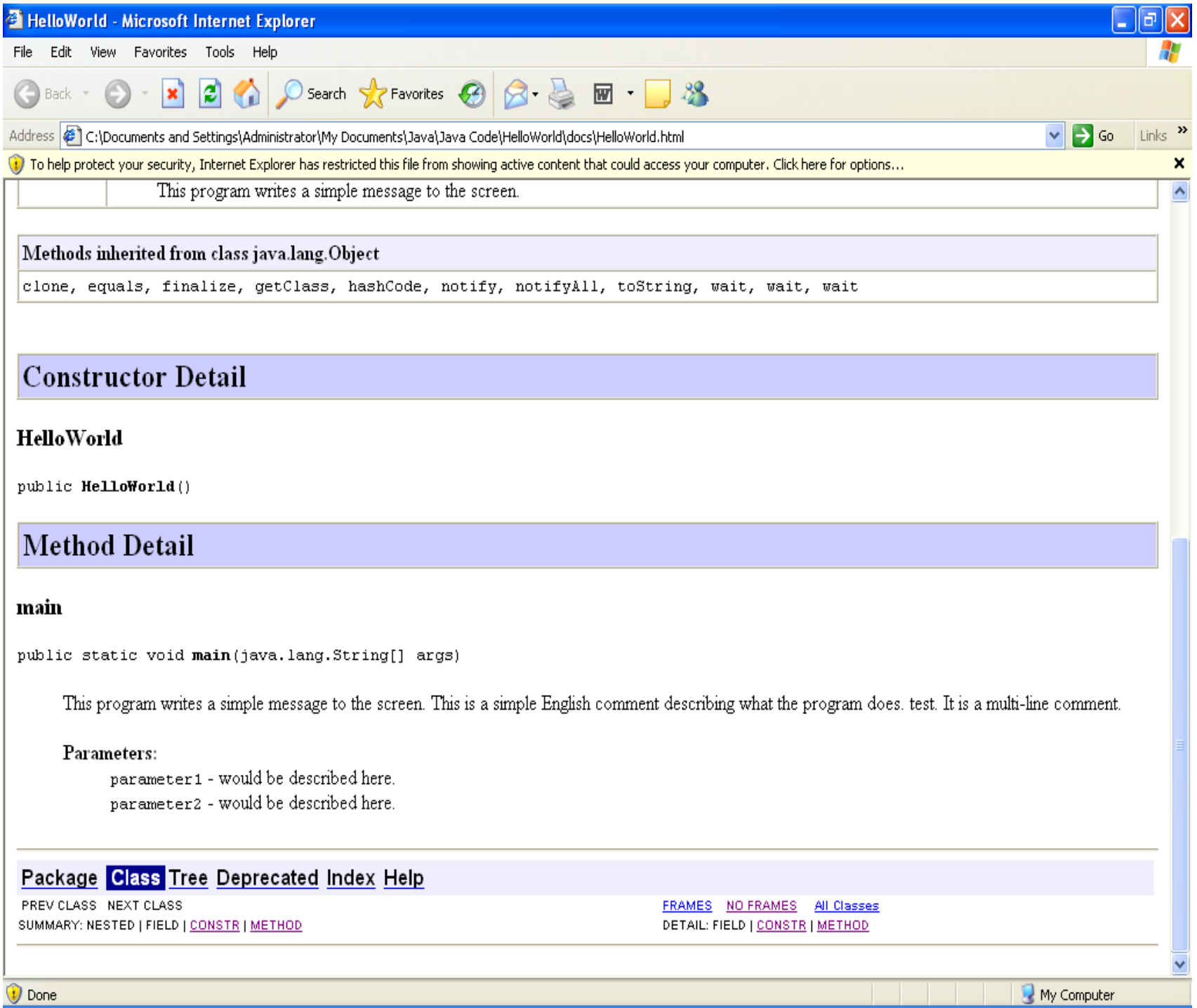
```
/** This class writes a simple message to the screen.
  program does. That message is contained within the
   class.
  @param no parameters for this class.
  @return  nothing returned by this program.
  @version v1.0
  @author Diarmuid O'Donoghue.
  @author 65109877. */
public class HelloWorld
{

    public static void main(String[] args)
    {
        System.out.println("Diarmuid really”
                                    + “ is great");
    }       // not very exciting I know

}
```

Javadoc for HelloWorld

Javadoc for HelloWorld



3 types of Comments

- There are 3 types of comments in Java

```
1. //  for a few in-line words
2. /*  */ for a paragraph
3. /**  */ used with Javadoc
```

- Comments are ignored by the compiler
- Comments are very **useful to people** who **read, use** and **update** your code

/ / comment

- Typically occurs on same line as a piece of code
- Comment finishes at the end of the line
- Useful to programmer who is looking at the lowest level of detail in the code

```
int radius ; // in  inches
prc = (pra/2) * 0.85 + (prb/2) * 0.85 + 0.15; // the PageRank formula
```

/* */ Comments

- The comment can extend over any number of lines
- Useful to programmers
- Can be used to explain the purpose of a couple of lines of code
- Ignored by Javadoc

/** */ Comments

- Used by `Javadoc` to automatically generate html
- Use one for each class and each method
- Place comment immediately *above* the relevant item

```
/** This is a simple class, that prints
    a message to the screen. */
public class HelloWorld
{
…
```

@param

- Explicitly identify parameters within the documentation
- 1 line given to each parameter
- Located within /** */

```
/** General method description ….
 @param deposit amount deposited by
customer
*/
```

@return

- Explicitly identify the retun value within the documentation
- 1 line given to each parameter
- Described within /** */

```
/**
 @param deposit amount deposited by customer
 @return newBalance new balance held by customer
*/
```

@author

- Identifies the author of a piece of code
- Use a separate line for each author

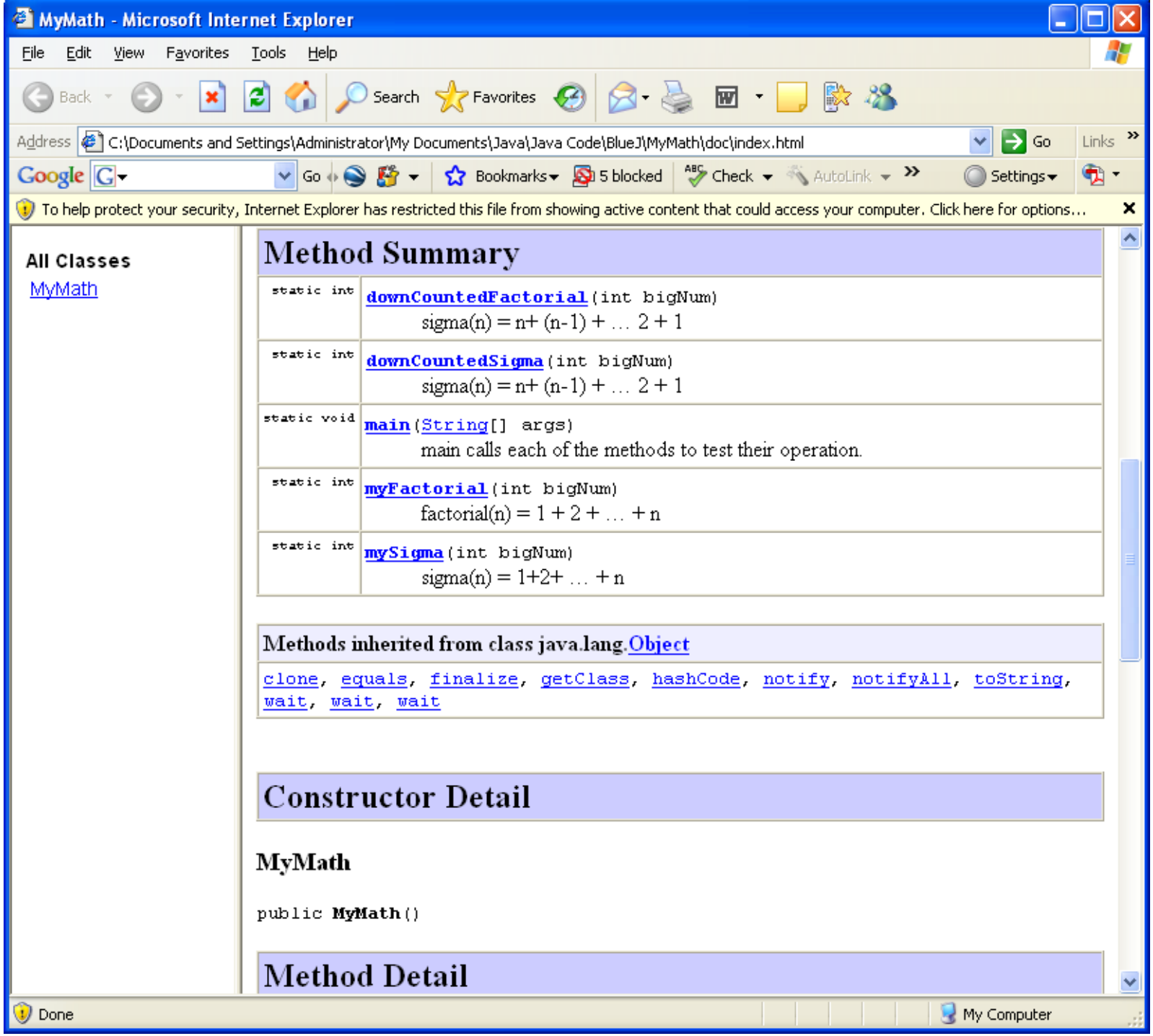
- Other Javadoc tags include:
- @version
- @see
- @deprecated
- @throws

/** */ in Course Work

- Include your name and ID as JavaDoc comments when submitting assignments
- As well as a description of your assignment

```
/**
 * This class is used for assignment 3.
 * @author Diarmuid O'Donoghue
 * @date 10 November 2005
 * @author 1st year CSSE
 * @author 65123456
*/
public class DODAssignment3
{
  public static void main (String[] args)
…
```


MyMath - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\Administrator\My Documents\Java\Java Code\BlueJ\MyMath\doc\index.html
Go Links
Google Go Bookmarks 5 blocked Check AutoLink Settings
To help protect your security, Internet Explorer has restricted this file from showing active content that could access your computer. Click here for options...
All Classes
MyMath
Package Class Tree Index Help
PREV CLASS NEXT CLASS
FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD
DETAIL: FIELD | CONSTR | METHOD
Class MyMath
java.lang.Object
MyMath
public class MyMath
extends Object
Various useful mathematical functions.
Version:
October 2006
Author:
D. O'Donoghue
Constructor Summary
MyMath()
Done
My Computer


MyMath - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address C:\Documents and Settings\Administrator\My Documents\Java\Java Code\BlueJ\MyMath\doc\index.html
Go Links
Google Go Bookmarks 5 blocked Check AutoLink Settings
To help protect your security, Internet Explorer has restricted this file from showing active content that could access your computer. Click here for options...
All Classes
MyMath
Method Summary
static int downCountedFactorial(int bigNum)
sigma(n) = n+ (n-1) + ... 2 + 1
static int downCountedSigma(int bigNum)
sigma(n) = n+ (n-1) + ... 2 + 1
static void main(String[] args)
main calls each of the methods to test their operation.
static int myFactorial(int bigNum)
factorial(n) = 1 + 2 + ... + n
static int mySigma(int bigNum)
sigma(n) = 1+2+ ... + n
Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait
Constructor Detail
MyMath
public MyMath()
Method Detail
Done
My Computer

Exercise

- Provide java documentation for the Circle Class and also for the application class you wrote to create an instance of the Circle Class.

- (Part of Lab 1 for Day 11) Provide java documentation for the Rectangle Class and also for the application class you wrote to create an instance of the Rectangle Class.

Conclusion

- JavaDoc Standard for documentation
- Javadoc comments allow English communication between collaborators
- Professional looking documentation