



**Java Code Geeks**  
JAVA 2 JAVA DEVELOPERS RESOURCE CENTER

Home » Java » Core Java » Java 7: Meet the Fork/Join Framework

### About Alexis Lopez



Java Lover, certified as Java Programmer, Mobile Application Developer and Web Component Developer.



## Java 7: Meet the Fork/Join Framework

Posted by: Alexis Lopez in Core Java  
October 17th, 2012

JSR-166(y) is the official name of this new feature which is included in Java 7. If you notice there is a 'y' in the name, this is because JSR-166 (Concurrency Utilities) is being added since Java 5, but it won't stop here as there are already plans to add new classes in Java 8 under the JSR-166(e). [Check this page](#) maintained by Doug Lea, the creator of JSR-166, for more information.

According to Wikipedia, [Parallelism](#) is the 'simultaneous execution of some combination of multiple instances of programmed instructions and data on multiple processors' and Java has classes and interfaces to achieve this (sort of...) since DAY 1. You may know them as: [java.lang.Thread](#), [java.lang.Runnable](#), etc... What Concurrency Utilities (

`java.util.concurrent` package) does is simplify the way we code concurrent tasks, so our code is much simpler and cleaner. As developers we haven't had to do anything when running our applications in machines with higher processing resources, obviously, the performance of our applications will improve, but are we really using the processing resources to the maximum? The answer is big NO.

This post will show you how the Fork/Join framework will help us in using the processing resources to the maximum when dealing with problems that can be divided into small problems and all the solutions to each one of those small problems produce the solution of the big problem (like recursion, divide and conquer).

### What you need

[NetBeans 7+](#) or any other IDE that supports Java 7 [JDK 7+](#)

[Blur on an image](#), example from Oracle

### The Basics

The Fork/Join framework focuses on using all the processing resources available in the machine to improve the performance of the applications. It was designed to simplify parallelism in [Divide and Conquer algorithms](#). The magic behind the Fork/Join framework is its work-stealing algorithm in which work threads that are free steal tasks from other busy threads, so all threads are working at all times. Following are the basics you should know in order to start using the framework:

- **Fork** means splitting the task into subtasks and work on them.
- **Join** means merging the solution of every subtask into one general solution.
- [java.lang.Runtime](#) use this class in order to obtain the number of processors available to the Java virtual machine. Use the method `+availableProcessors():int` in order to do so.
- [java.util.concurrent.ForkJoinPool](#) Main class of the framework, is the one that implements the work-stealing algorithm and is responsible for running the tasks.
- [java.util.concurrent.ForkJoinTask](#) Abstract class for the tasks that run in a `java.util.concurrent.ForkJoinPool`. Understand a task as a portion of the whole work, for example, if you need to do something on an array, one task can work on positions  $0$  to  $n/2$  and another task can work on positions  $(n/2) + 1$  to  $n-1$ ,

where  $n$  is the length of the array.

- **java.util.concurrent.RecursiveAction**  
Subclass of the abstract task class, use it when you don't need the task to return a result, for example, when the task works on positions of an array, it doesn't return anything because it worked on the array. The method you should implement in order to do the job is **compute():void**, notice the void return.
- **java.util.concurrent.RecursiveTask**  
Subclass of the abstract task class, use it when your tasks return a result. For example, when computing Fibonacci numbers, each task must return the number it computed in order to join them and obtain the general solution. The method you should implement in order to do the job is **compute():V**, where  $V$  is the type of return; for the Fibonacci example,  $V$  may be `java.lang.Integer`.

When using the framework, you should define a flag that indicates whether it is necessary to fork/join the tasks or whether you should compute the work directly. For example, when working on an array, you may specify that if the length of the array is bigger than 500\_000\_000 you should fork/join the tasks, otherwise, the array is small enough to compute directly. In essence, the algorithm you should follow is shown next:

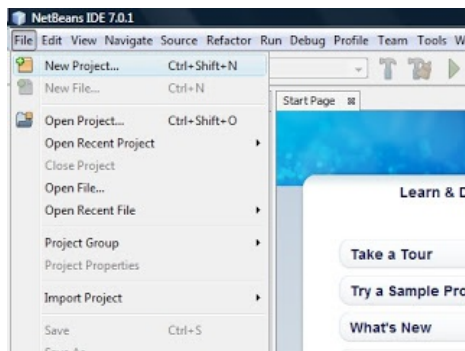
```
if(the job is small enough)
{
    compute directly
}
else
{
    split the work in two pieces (fork)
    invoke the pieces and join the results (join)
}
```

OK, too much theory for now, let's review an example.

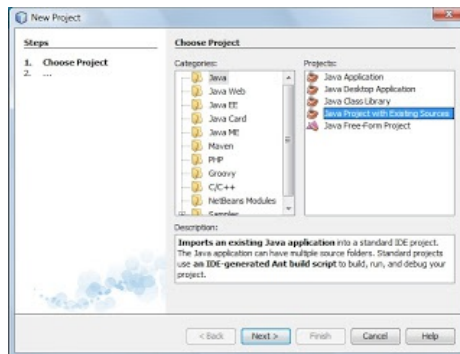
### The Example

Blurring an image requires to work on every pixel of the image. If the image is big enough we are going to have a big array of pixels to work on and so we can use fork/join to work on them and use the processing resources to the maximum. You can download the source code from [the Java™ Tutorials site](#).

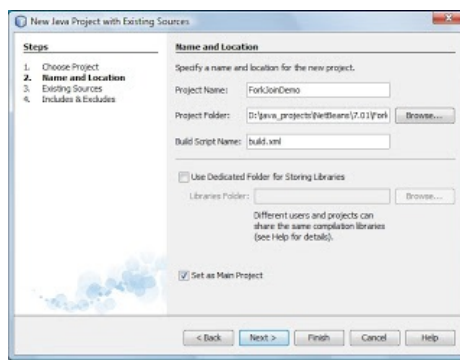
Once you download the source code, open NetBeans IDE 7.x and create a new project:



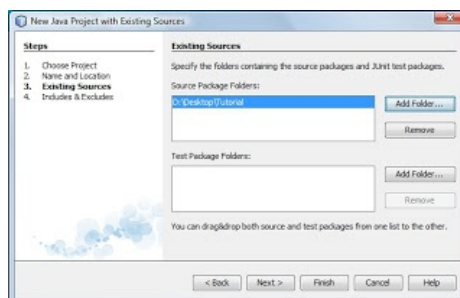
Then select *Java Project with Existing Sources* from the *Java* category in the displayed pop-up window:



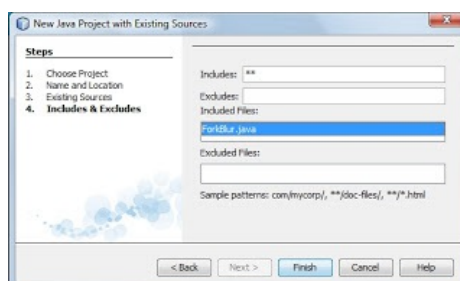
Select a name and a project folder and click **Next >**



Now, select the folder where you downloaded the source code for the [Blur on an image](#) example:

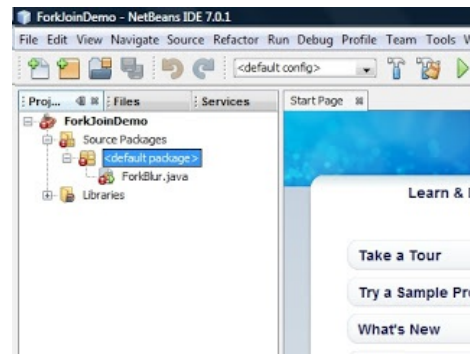


And select the file `ForkBlur.java` then click finish:

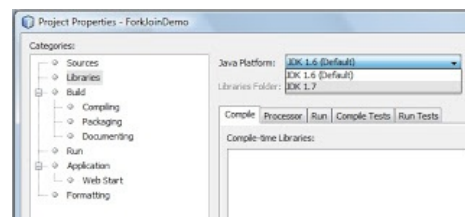




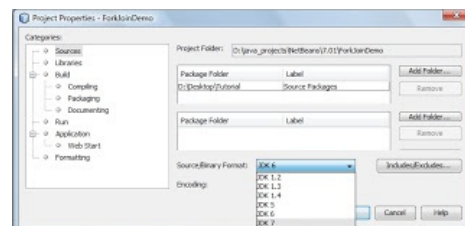
The source code will be imported and a new project will be created. Notice that the new project is shown with errors, this is because Java 7 is not enable for default:



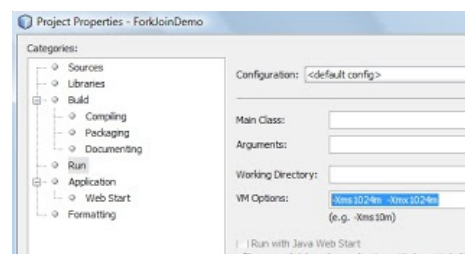
To fix this, right click on the project name and select the option *Properties*. On the pop-up dialog, go to *Libraries* and select **JDK 1.7** from the *Java Platform* ComboBox:



Now, go to the option *Sources* and select **JDK 7** from the *Source/Binary Format* ComboBox:



Last but not least, increase the memory assigned to the virtual machine when running this application as we'll be accessing a 5 million positions array (or more). Go to the option *Run* and insert **-Xms1024m -Xmx1024m** on the *VM Options* TextBox:



Click *OK* and your project should be compiling with no errors. Now, we need to find an image bigger enough so we can have a large array to work on. After a while, I found some great images (around 150 MB) from planet Mars, thanks to the curiosity robot, you can [download yours from here](#). Once you download the image, past it on the project's folder.

Before we run the example, we need to

modify the source code in order to control when to run it using the Fork/Join framework. In the ForkBlur.java file, go to line 104 in order to change the name of the image that we are going to use:

```
//Change for the name of the image you pasted
//on the project's folder.
String filename = 'red-tulips.jpg';
```

Then, replace lines 130 to 136 with the following piece of code:

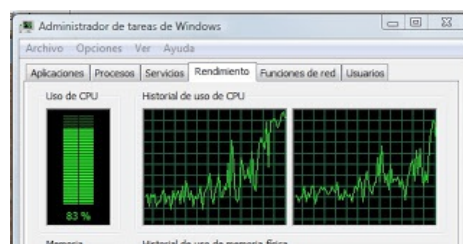
```
ForkBlur fb = new ForkBlur(src, 0, src.length, dst);
boolean computeDirectly = true;

long startTime = System.currentTimeMillis();
if (computeDirectly) {
    fb.computeDirectly();
} else {
    ForkJoinPool pool = new ForkJoinPool();
    pool.invoke(fb);
}
long endTime = System.currentTimeMillis();
```

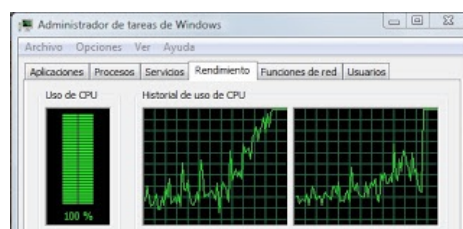
Notice the **computeDirectly** flag. When **true**, we'll **NOT** be using the fork/Join Framework, instead we will compute the task directly. When false, the fork/join framework will be used.

The **compute():void** method in the *ForkBlur* class implements the fork/join algorithm. It's based on the length of the array, when the length of the array is bigger than 10\_000, the task will be forked, otherwise, the task will be computed directly.

Following you can see my 2 processors when executing the [Blur on an image](#) example without using the Fork/Join framework (**computeDirectly = true**), it took about 14s to finish the work:



You can see that the processors are working, but not to the maximum. When using the Fork/Join framework (**computeDirectly = false**) you can see them working at 100% and it took almost 50% less time to finish the work:




This video shows the complete process:

I hope you can see how useful this framework is. Of course, you cannot use it all around your code, but whenever you have a task that can be divided into small tasks then you know who to call.

**Reference:** [Java 7: Meet the Fork/Join Framework](#) from our JCG partner Alexis Lopez at the [Java and ME](#) blog.

Do you want to know how to develop your skillset to become a **Java Rockstar**?



Subscribe to our newsletter to start Rocking **right now**! To get you started we give you our best selling eBooks for **FREE!**

1. JPA Mini Book
2. JVM Troubleshooting Guide
3. JUnit Tutorial for Unit Testing
4. Java Annotations Tutorial
5. Java Interview Questions
6. Spring Interview Questions
7. Android UI Design

and many more ...

Email address:

[Sign up](#)

Tagged with: [FORK/JOIN](#)

## Leave a Reply

### 1 Comment on "Java 7: Meet the Fork/Join Framework"

Notify of



Join the discussion

Sort by: [newest](#) | [oldest](#) | [most voted](#)



Guest

[Tomasz Nurkiewicz](#)

11 screenshots explaining how to create a project in NetBeans and almost no code in an article about Fork/Join pool?





🕒 4 years 9 months ago

**ZEPHYR** 

---

**VU A LA TV**

**Gravure Laser 3D**  
Des centaines d'idées  
**CADEAUX**

What do you want to monitor?  
**AppDynamics Free Trial**

Applications

User Experiences

Databases

Java

.NET

PHP

Node.js

Python

Test User

Start Your Free Trial

Databases

Servers

Services

**APPDYNAMICS**

## NEWSLETTER

**179,260** insiders are already enjoying weekly updates and complimentary whitepapers!

**Join them now** to gain exclusive access to the latest news in the Java world, as well as insights about Android, Scala, Groovy and other related technologies.

**Email address:**

☒ Receive Java & Developer job alerts in your Area

**Sign up**

## JOIN US



With **1,240,600** monthly unique visitors and over **500** authors we are placed among the top Java related sites around. Constantly being on the lookout for partners; we encourage you to join us. So If you have a blog with unique and interesting

content then you should check out our **JCG** partners program. You can also be a **guest writer** for Java Code Geeks and hone your writing skills!



## KNOWLEDGE BASE

---

[Courses](#)

[Examples](#)

[Minibooks](#)

[Resources](#)

[Tutorials](#)

## PARTNERS

---

[Mkyong](#)

## THE CODE GEEKS NETWORK

---

[.NET Code Geeks](#)

[Java Code Geeks](#)

[System Code Geeks](#)

[Web Code Geeks](#)

## HALL OF FAME

---

["Android Full Application Tutorial" series](#)

[11 Online Learning websites that you should check out](#)

[Advantages and Disadvantages of Cloud Computing – Cloud computing pros and cons](#)

[Android Google Maps Tutorial](#)

[Android JSON Parsing with Gson Tutorial](#)

[Android Location Based Services Application – GPS location](#)

[Android Quick Preferences Tutorial](#)

[Difference between Comparator and Comparable in Java](#)

[GWT 2 Spring 3 JPA 2 Hibernate 3.5 Tutorial](#)

[Java Best Practices – Vector vs ArrayList vs HashSet](#)

## ABOUT JAVA CODE GEEKS

JCGs (Java Code Geeks) is an independent online community focused on creating the ultimate Java to Java developers resource center; targeted at the technical architect, technical team lead (senior developer), project manager and junior developers alike. JCGs serve the Java, SOA, Agile and Telecom communities with daily news written by domain experts, articles, tutorials, reviews, announcements, code snippets and open source projects.

## DISCLAIMER

All trademarks and registered trademarks appearing on Java Code Geeks are the property of their respective owners. Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries. Examples Java Code Geeks is not connected to Oracle Corporation and is not sponsored by Oracle Corporation.



Java Code Geeks and all content copyright © 2010-2017,  
Exelixis Media P.C. | [Terms of Use](#) | [Privacy Policy](#) | [Contact](#)