Tools

Introduction

In this article from my free Java 8 Course, I will be discussing the programming tools that I will be using during this course.

Java IDEs

Let's start with something that I've discussed before, our Integrated Development Environment or IDE. The three most popular IDEs are Eclipse, IntelliJ IDEA, and Netbeans, in that order. Eclipse is free and open source, so it is very popular, especially among companies. I have been using Eclipse since 2002, but it's starting to fall behind other major IDEs. I'm using this course as an opportunity to show off IntelliJ IDEA, which is becoming more and more popular. From what I've seen, IntelliJ IDEA is much more powerful compared to Eclipse, and unlike Eclipse, doesn't require additional plugins to allow you to start using it. Also, Eclipse has many bugs that are frustrating to deal with and is infrequently maintained.

Finally, NetBeans is another free IDE that is also popular in corporate software development. It has great autocomplete and autoimport features. These three and many others are all valid choices, and I always feel that each individual should pick the IDE they are comfortable with. In my opinion, a seasoned developer should be at least somewhat familiar with *all* three. Here are the download links for the three IDES:

Eclipse
https://eclipse.org/downloads/

IntelliJ IDEA
https://www.jetbrains.com/idea/download/

NetBeans
https://netbeans.org/downloads/

Testing Frameworks

Next, let's talk about the different options for creating Tests. In all the tests I've used so far, I've used JUnit, which is a testing framework. Frameworks are a set of classes bundled together. JUnit is open source and free, making testing really easy. Most IDEs also have it preinstalled which makes it the most easily available choice.

Another Testing Framework is TestNG, which is very similar to JUnit. IntelliJ IDEA and Eclipse actually support both testing frameworks, so you can easily try both, but I'm not going to go into more detail about using TestNG, purely out of preference.

Maven

Another program I'd like to talk about is Maven. Maven is a build management tool; as the name implies it manages your build. Basically, a build is the process of turning your program into one file. Imagine that your program consists of hundreds of classes that you need to release or send to a client. You don't want to send each file seperately, so you gather them in a package and turn that package into a single file. Before the original build management tool Unix's make, this was a manual task requiring various scripts to compile software, but now we have tools like Maven to do it for us! There are other build tools, but for this course I'm using Maven.

I highly recommend that you also read through the documentation on the website to understand how Maven works.

Text Editors

The final thing I'll talk about in this article is text editors. I've mentioned before in the course that you should start learning Java with a text editor. An IDE is great for a seasoned developer; it provides many features that can greatly assist programmers, such as warnings and error highlighting in your code, automatic code completion and tools to help with code refactoring. However, it also acts as a crutch and causes a beginner to avoid actually learning many things that they should at least be aware of while coding. Using a text editor, you can write code, save it, and build your project in the console, which is really all you need if you're learning Java for the first time.
For Windows, I recommend that you use Textpad. It is very simple to use, yet very powerful. For all other operating systems, use Sublime. Both offer features like code highlighting and advanced search and replace features. Again, I recommend that you use one of these text editors throughout this course as it'll help you retain more Java and focus on writing high quality code.

Thanks for reading!