

Forum:[Hack Forums](#)

Thread: [\[Infamous\] The Java Toolkit Repository](#)

MavenGradle - February 28, 2020, 11:20 am

This informative thread is brought to you by Infamous and serves as a collection of tools, frameworks and general tips and tricks in regards to Java programming. This is by no means an extensive or sponsored thread. All suggestions are made by me and are not influenced by advertisements. Do note that the list is not complete, and as such, some tools and features that you're used to might not be listed here.

You are free to add, correct or remove entries from this list if a general consensus can be reached.

Java has quite a few useful tools, IDE's, extensions, build tools and general plugins available that can help make your life as a developer easier. The Java ecosystem is large, if not one of the largest ecosystems there are in regards to computer programming. This serves as an easy guide and gateway into the larger ecosystem, offering suggestions and features that perhaps you were not aware of before.

Java IDE's

Let's start off with general IDE's that you can use to make your dreams a reality, using Java.

IDE stands for Integrated Development Environment. They are generally regarded as software applications that provide comprehensive facilities to computer programmers for software development in an attempt to make life a little bit easier.

The more popular Java IDE's include (but are not limited to):

[IntelliJ IDEA](#)

[Eclipse](#)

[Netbeans](#)

Although [Visual Studio Code](#) is not generally regarded as an IDE, you can still use it to develop, debug and deploy Java applications.

Generally speaking, IntelliJ IDEA is currently the most popular IDE on the market, offering licenses for free to open source developers and students. You can apply for free licenses here:

<https://www.jetbrains.com/idea/buy/#discounts?billing=yearly>

If you do not meet the requirements for a free license, I highly recommend purchasing a license.

If you'd like to keep up with the development and features of IntelliJ IDEA, it is highly recommended to follow their blog, which is available through the following link:

<https://blog.jetbrains.com/idea/>

Eclipse would be the second most popular choice when it comes to Java IDE's, offering a wide range of plugins, extensions and tools to make your life easier. Additionally, Eclipse is an open-source editor, which means you can contribute to the project if you feel so inclined:

<https://git.eclipse.org/c/>

If you'd like to keep up with the development and features of Eclipse, it is highly recommended to follow their blog, which is available through the following link:

<https://blogs.eclipse.org/>

NetBeans is generally regarded as the 'official' IDE for Java development, provided to you by Oracle and the Apache Software Foundation. While it's not as extensive as the IDE's above, it shines in its ease of use and general friendliness to newcomers.

If you'd like to keep up with the development and features of NetBeans, it is highly recommended to follow their blog, which is available through the following link:

<https://blogs.apache.org/netbeans/>

Build Tools

"Build tools are programs that automate the creation of executable applications from source code(eg. .apk for android app). Building incorporates compiling,linking and packaging the code into a usable or executable form." - [Ritesh Gune](#)

You might ask yourself why you'd need a build tool if your IDE has the ability to compile Java for you. The answer is quite simple. Manually invoking builds for larger projects is impractical and can add to the development time. Sometimes, builds need to be performed in a certain order and it can be extremely hard to fully go through this process manually when developing for larger projects. In addition, they offer a great way to integrate DevOps and continuous integration tools.

With every great language, comes a great build tool. There are two that immediately come to mind when talking about Java build tools. Those would be [Apache Maven](#) and [Gradle](#).

Both are entirely free to use and offer a wide range of functionality. Although Gradle generally has better performance, Maven has been around for longer and as such has more plugins available. You can find a great comparison between the two on the Gradle website:

<https://gradle.org/maven-vs-gradle/>

In general, it is believed that Gradle offers increased performance (mostly due to the gradle daemon running in the background) over Maven while still offering a similar set of features.

Apache Maven configuration is usually done using XML files such as the pom.xml file which could look like:

```
Code <?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
```

```
https://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.2.5.RELEASE</version>
<relativePath/> [b]<!--[/b] lookup parent from repository -->
</parent>
<groupId>com.example</groupId>
<artifactId>demo</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>demo</name>
<description>Demo project for Spring Boot</description>

<properties>
<java.version>13</java.version>
</properties>

<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
<exclusions>
<exclusion>
<groupId>org.junit.vintage</groupId>
<artifactId>junit-vintage-engine</artifactId>
</exclusion>
</exclusions>
</dependency>
<dependency>
<groupId>org.springframework.security</groupId>
<artifactId>spring-security-test</artifactId>
<scope>test</scope>
</dependency>
</dependencies>

<build>
<plugins>
<plugin>
```

```

<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
</plugins>
</build>
</project>

```

Whereas a Gradle build file makes use of the Groovy language and could look something like this:

```

Code plugins {
    id 'org.springframework.boot' version '2.2.4.RELEASE'
    id 'io.spring.dependency-management' version '1.0.9.RELEASE'
    id 'java'
}

```

```

group = 'net.hackforums'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '13'

```

```

repositories {
    mavenCentral()
}

```

```

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-actuator'
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    testImplementation('org.springframework.boot:spring-boot-starter-test') {
        exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
    }
}

```

```

test {
    useJUnitPlatform()
}

```

If you were paying attention, you might have noticed how both Maven and Gradle allow for dependency management. They do so, through the usage of (artefact) repositories. MavenCentral (or the Maven Central Repository) is one such repository. You can look through this repository and see which artefacts and resources are available through the following links:

<https://search.maven.org/>
<https://mvnrepository.com/repos/central>

BinTray jCenter is another popular repository that you can browse through and utilize in your Gradle and Maven projects:

<https://bintray.com/bintray/jcenter>

Finally, a developer could also decide to use Apache Ant in combination with something like ivy although Ant is generally no longer recommended due to its complexity and slow development cycles. You can find a great comparison between the available build tools here:

<https://devopscube.com/list-of-popular-open-source-java-build-tools/>

Code analysis

Code analysis and feature suggestion tools are a great way to improve your code quality and might even help you develop some good practices while creating Java applications. Most IDE's, such as IntelliJ Idea and Eclipse, offer code analysis and code suggestions out of the box although this is not true for all IDE's and definitely not a required feature for an IDE.

There are tools and plugins, however, that can help you detect code smells, dead code, unnecessary casts and more such as [SonarLint](#). SonarLint is free and completely open-source. It integrates nicely in most of the IDE's mentioned here already. Some of its features can even be integrated into your CI environment using [SonarQube](#).

[PMD](#) is another code analyzer that's entirely open-source and integrates nicely with build tools like Maven or Gradle.

Frameworks and toolkits

You shouldn't have to reinvent the wheel. Java is one of the most popular programming languages around and as such, there are so many libraries, toolkits and frameworks available that can help you to quickly develop and realize your applications, be it a web application, desktop application or an application for IoT devices, Java and its community have you covered.

The list of Java Frameworks, tools and libraries is so enormous that it's almost impossible to list them all. That's why I'm only going over the most popular frameworks, libraries and toolkits that are available on the market today.

[Akka](#): simplifying the construction of concurrent and distributed applications on the JVM

[Apache ActiveMQ](#): Messaging and Integration Patterns server.

[Apache Commons](#): Provides reusable, open source Java software that prevent you from reinventing the wheel

[Apache CXF](#): CXF helps to build and develop services using frontend programming APIs, like JAX-WS and JAX-RS. This is useful for building webservices that utilize SOAP

[Apache Hadoop](#): Used for storing and processing big-data processing

[Apache Kafka](#): Stream-processing software platform

[Apache Log4j2](#): Logging framework

[Apache PDFBox](#): Useful for creating and editing PDF documents

[Apache POI](#): A useful toolkit for creating and editing Microsoft Office documents (e.g. Word, Excel, Powerpoint)

[Apache Spark](#): Cluster-computing framework

[Apache Tomcat](#): An open-source Java Servlet Container

[Apache ZooKeeper](#): A service for distributed systems offering a hierarchical key-value store

[GlassFish](#): Application server and official reference implementation for Servlets 3.0

[Hibernate](#): Object-relational mapping tool for the Java programming language

[Jackson](#): JSON processor (serialization/deserialization)

[JavaFX](#): The modern way to develop Java Desktop applications

[Jetty](#): HTTP (Web) server and Java Servlet container.

[Mockito](#): Unit testing framework (Mocking)

[Netty](#): Non-blocking I/O (NIO) client-server framework

[RabbitMQ](#): Message broker software that implements the Advanced Message Queuing Protocol (AMQP)

For a more extensive list, please take a look at:

https://en.wikipedia.org/wiki/List_of_Java_frameworks

Additionally, the following frameworks deserve a list of their own due to their extreme ease of use and generally excellent craftsmanship. They are extremely useful when, for example, developing microservices, RESTful API's and more:

[Spring \(Boot\)](#): Framework and inversion of control container for the Java platform

[Quarkus](#): Cloud Native, (Linux) Container First framework for writing Java applications

[Micronaut](#): Modern, JVM-based, full-stack framework for building modular, easily testable microservice and serverless applications

DevOps/CI

There are various CI tools available that are compatible with Java and the many tools and frameworks that come along with it. The most popular one to date would be [Jenkins](#). Jenkins is completely open-source and extensible using the many plugins that are available for it. For more information, please see:

<https://github.com/jenkinsci/jenkins>

[Travis](#) is also a great option, offering a service to build your open-source GitHub projects for free: <https://travis-ci.com/plans>

Azure DevOps will have to be the alternative option when it comes to the Java ecosystem. Relatively newer than the other options listed here, Azure DevOps also offers pipelines and free

build minutes for open-source projects:

<https://azure.microsoft.com/en-us/services/devops/?nav=min>

In case you'd like to host your own repositories, artifacts, or docker images (and more), a tool like SonaType Nexus is always a great addition. It can be used to host a multitude of packages and formats, including but not limited to: jar artifacts, docker images, NuGet packages, NPM packages and many more. The free edition is available here:

<https://www.sonatype.com/nexus-repository-oss>

External resources

As stated before, the Java community has you covered. As such, there are more extensive guides and general entry points available. For example, if you'd like to read more about the various Java tools, frameworks and more, this would be a great place to start:

<https://github.com/akullpp/awesome-java>

<https://github.com/Vedenin/useful-java-links>

In regards to Java Desktop application development and the JavaFX platform, take a look at:

<https://github.com/mhrimaz/AwesomeJavaFX>

That's it for now. If you'd like to see a tool, framework or otherwise added to this list, feel free to leave a comment and I'll do my best to keep this updated as long as possible. Feel free to extend upon the data that's presented here.

I hope you find this at least somewhat useful, and perhaps improve your code using the code analysers listed above. They're mighty useful and I wouldn't want to live without them any more.

GenericSkid - February 28, 2020, 1:58 pm

Perhaps you could list some of the "Awesome" lists for further reading. For example:

<https://github.com/akullpp/awesome-java>

<https://github.com/mhrimaz/AwesomeJavaFX>

<https://github.com/Vedenin/useful-java-links>

Also you can discover some libraries on <https://mvnrepository.com/> through the tag system. Most stuff isn't tagged, but its still a very useful tool site.

Some java blogs about current events: <https://blog.jetbrains.com/idea/author/trishagee/>

And keeping up with the latest features: <http://openjdk.java.net/projects/jdk/>

MavenGradle - February 28, 2020, 2:37 pm

(Feb 28, 2020 02:58 PM)GenericSkid Wrote: [\[CITE\]](#) Perhaps you could list some of the "Awesome" lists for further reading. For example:

<https://github.com/akullpp/awesome-java>

<https://github.com/mhrimaz/AwesomeJavaFX>

<https://github.com/Vedenin/useful-java-links>

Also you can discover some libraries on <https://mvnrepository.com/> through the tag system. Most stuff isn't tagged, but its still a very useful tool site.

Some java blogs about current events: <https://blog.jetbrains.com/idea/author/trishagee/>

And keeping up with the latest features: <http://openjdk.java.net/projects/jdk/>

Thanks for the suggestions, they've been incorporated into the thread and more improvements in general have also been added. The JEP's (that are accepted) all deserve threads on their own though. There's a great opportunity here for someone to take the initiative in that regard!

Also, did you know that the Maven Central Repository is hosted by Sonatype? Incidentally, the thread here has already linked to resources on how you can host your own repository using Sonatype Nexus! Here's a great blog post from Sonatype from a little while back and the wonderful work they've been doing:

<https://blog.sonatype.com/central-repository-moving-to-https>

Pawnhub - March 1, 2020, 2:52 am

Very nice in depth look at some of the notable libraries and frameworks. I really do like the explanation of separate build tools and why under certain circumstance you'd need one & now I know why you're name is MavenGradle ? it's all making sense now!

On the note of JavaFX I seen the Deep Space Trajectory Explorer from NASA on there github that simulation is very impressive.

This thread in my opinion deserves to be pinned in this section and it's going to save me a lot of time when I have to reference this thread in the future.

Well done Maven & to infamous for a very HQ release.

MavenGradle - March 2, 2020, 12:29 pm

(Mar 1, 2020 03:52 AM)UnknownPawn Wrote: [\[CITE\]](#) Very nice in depth look at some of the notable libraries and frameworks. I really do like the explanation of separate build tools and why under certain circumstance you'd need one & now I know why you're name is MavenGradle ? it's all making sense now!

On the note of JavaFX I seen the Deep Space Trajectory Explorer from NASA on there github that simulation is very impressive.

This thread in my opinion deserves to be pinned in this section and it's going to save me a lot of time when I have to reference this thread in the future.

Well done Maven & to infamous for a very HQ release.

Thanks for the kind words [Pawnhub](#) . JavaFX is mighty useful, but seriously underestimated in its capabilities for desktop development in general by developers. I'm not saying it's heaven on earth to work with, but it's one of the few platforms out there that allow for true multi-platform desktop application development. If only it was as easy to work with as Electron, but then again, Electron is resource hungry on a system. JavaFX is still rather conservative when it comes to consuming

system resources and with GraalVM support, it runs like absolute butter.

In regards to the NASA application, for those who are wondering, here's a nice article about it:
<https://jaxenter.com/netbeans/developing-nasas-mission-software-with-java>

Aussie - April 2, 2020, 12:29 am

Awesome thread, apologies I'm a bit late to post.

Very useful and descriptive, would love to have OSGI added or a modularity section.

Love the fact you mentioned DevOps, as that's a part often forgotten about by developers, even though highly necessary for any SRE.

Top notch thread, would be well worth a sticky!

SIM-1729678 - April 2, 2020, 12:31 am

Very nice thread Maven. HQ as always.

MavenGradle - February 11, 2021, 7:57 am

(Apr 2, 2020 12:31 AM)Screems Wrote: [\[CITE\]](#) Very nice thread Maven. HQ as always.

Thank you. It appears as though the information inside this thread is still relevant in 2021.

Juoku - February 14, 2021, 5:02 am

Awesome guide filled with useful tools, I recently started studying Java and app dev and this is super good!

will definitely take a closer look at this and get down to it, luckily for me I've downloaded the right tools so far ;)

SIM-2501407 - March 15, 2021, 7:14 am

Nice guide.

I liked the `Frameworks and toolkits`; gave a good one sentence summary of frameworks.