

<BEGINNERS  
PROGRAMMING GUIDE>



# JAVA

<**33** BEST JAVA  
TIPS AND TRICKS>



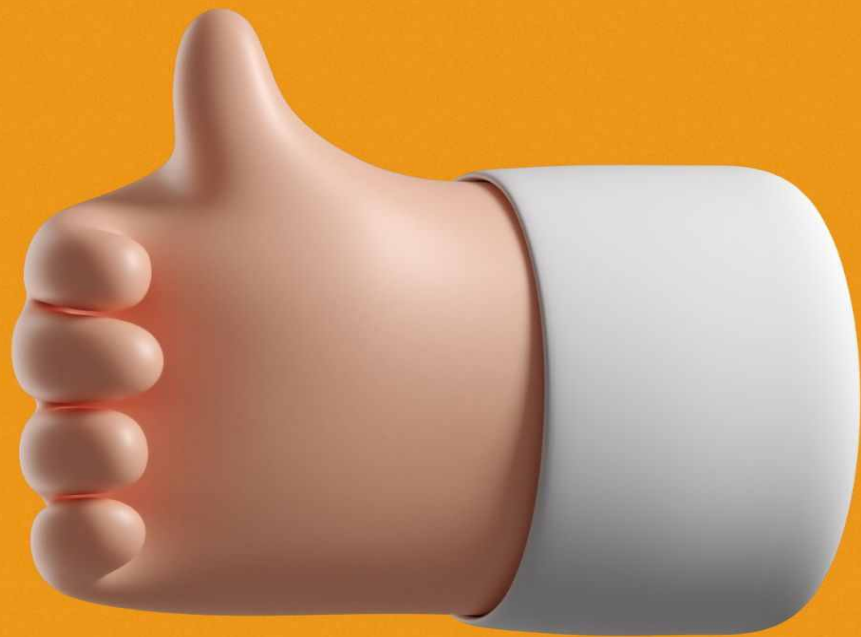
**2021**

<BEGINNERS  
PROGRAMMING GUIDE>



# JAVA

<**33** BEST JAVA  
TIPS AND TRICKS>



***2021***



# **Java:** 2021 Beginners Programming Guide. 33 Best Java Tips and Tricks

## CONTENTS:



### Introduction

What is the Java language?

Why is Java such a popular programming language?

What is the Java programming language?

The "cache issue"

Choosing the right editor

What are the advantages of the Java language?

What are the drawbacks of the Java language?

Which programming languages can be used in the Oracle Digital Cloud Office?

How to install Java on the Oracle Digital Cloud Office?

Wrapping it up

### Chapter 1 What are the features of the Java language?

Are there any flaws in the Java language design?

Are there any design flaws in the J2EE application server design?

Are the versions specified by the Java Language Specification (JLS) to be the final versions, or will they be revised in some way?

Should I write J2EE applications using the J2EE application server specification?

How can a J2EE server be used?

How does a J2EE client know how to create a program to communicate with the server?

What J2EE features and standard J2EE servers have been defined by the J2EE specification?

How does the JavaServer interface work?

Can Java EE 6 use J2EE?

Can Java EE 6 use J2EE libraries?

Does the JavaServer implementation interface behave differently in Java EE 6 than in Java EE 5?

Does Java EE 6 specify the way that J2EE code is compiled and run?

[Is it possible to deploy Java EE 6 applications in J2EE-based configuration?](#)

[File System Component](#)

[Database Component Registry](#)

[Message Component Registry](#)

[What features for Java development?](#)

[Notable changes in JDK 9](#)

[ProdJ99077 New features of JDK 9](#)

[Improved Exception Handling](#)

[Separately Generated Type Hints](#)

[Null Pointer Exceptions](#)

[Optimizations](#)

[New Inline Function Generator](#)

[Null in type inference](#)

[Favor closed over types](#)

[Why the first argument matters](#)

[What is the best IDE for Java development?](#)

[Chapter 2 What IDE chooses for Java programming?](#)

[Eclipse IDE](#)

[NetBeans IDE](#)

[IntelliJ IDEA](#)

[Novation IDE](#)

[iDevelop IDE](#)

[Ardu Pascal](#)

[Bitbucket CodeSourcery](#)

[JetBrains WebStorm](#)

[Xamarin](#)

[Who uses the most Java programming language?](#)

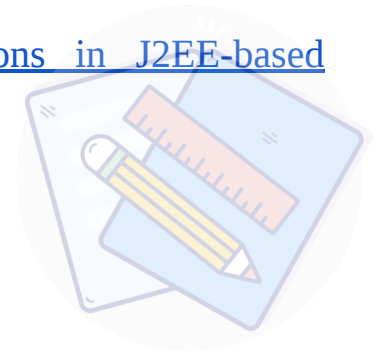
[What is the best IDE for iOS development?](#)

[What is the best IDE for Silverlight development?](#)

[What is the best IDE for .NET development?](#)

[How do you learn Java the right way?](#)

[What are you seeing that you wish was better in Java?](#)



[What is your favorite IDE for Java development?](#)

[What IDE should I use? Or, what IDE should I learn first?](#)

### [Chapter 3 What are plugins for Java programming?](#)

[Java plugins for .NET](#)

[Why should you create a Java plugin for .NET?](#)

[But how do you create a Java plugin for .NET?](#)

[What are the downsides of Java plugins for .NET?](#)

[What do you need to get started?](#)

[How do I add a plugin to my existing project?](#)

[How do I add a source folder to my .NET project?](#)

[How do I add a source folder to my .NET project using Eclipse?](#)

[How do I add source folders to my .NET project using Visual Studio?](#)

### [Chapter 4 Do Java and Javascript the same?](#)

[What should I learn first?](#)

[Java](#)

[Some tips for Java developers](#)

[What are frameworks for Java?](#)

[Framework design considerations](#)

[Aesthetics and syntax](#)

[Scala and MVC](#)

[MVC and Spring](#)

[Reactive programming in a web application - A real-world case](#)

[How to Build a Reactive system](#)

[Connecting to a reactive database in Scala](#)

### [Chapter 5 What are tools for Java language?](#)

[Does the tooling work well?](#)

[Does the tool provide the necessary configuration?](#)

[Is the configuration properly organized?](#)

[Is the configuration customizable?](#)

[The Future](#)

[Structural improvements](#)

### [Chapter 6 How to learn Java programming language?](#)



If you are ready to get started with learning Java programming language, follow these steps:

How to learn Java programming?

What is a constant?

What's the largest prime number?

What is a constant(T) in Java?

How do you define the type of a parameter?

How do you return the value of a parameter in a method?

What are the methods of the parameterized classes?

Do you have to create a class and implement methods with the parameters of the parameterized classes?

How to add a ParameterizedMethodBuilder to a parameterized class?

Do you have to create a custom implementation of a custom class in order to call the methods of the parameterized classes?

How to use a parameterized class as a parameterized class?

Do parameterized classes have to be subclassed?

Can I call the methods of the parameterized classes like this:

What is the difference between a weak parameterized class and a strong one?

## Chapter 7 How to use Java programming libraries?

The Best of both worlds: Libraries for Java programming.

Some of the many uses for Java libraries.

Not so easy to use: Libraries require some effort

What are some of the problems you face when using Java libraries?

Scalability

Instance-based Data Structures

The best comparison that I can make is the following:

Create new trees as needed

But how could we create a tree that has four children?

But how could we create a tree with eight children?

Now, what if we wanted to create a tree with ten children?

But, our new triple-tree only has five nodes in it. But what if we wanted a double-tree with 10 nodes in it?



[So, what is the most common size of the internal structure of a Tree?](#)  
[What about when we don't have to create new trees, but just combine existing trees?](#)

## [Chapter 8 How to find work as a Java programmer?](#)

[How to build a career as a Java developer? 4 top development tips](#)

[What is a Java developer?](#)

[What is the Java language?](#)

[Learn Java quickly.](#)

[Get over the fear of implementation](#)

[Start small and learn from mistakes](#)

[Develop your ability to solve problems](#)

[Create value through partnerships and use cases](#)

[Develop strong persistence skills](#)

[Learn Java using tools](#)

[Create and present software prototypes](#)

[Follow new technology.](#)

[Use cloud-based development](#)

[Get familiar with distributed programming](#)

[Watch the code](#)

[Stop saying "it will be done"](#)

[Create documents](#)

[Use pair programming](#)

[Find a mentor](#)

[Use revision control](#)

[Start learning](#)

[Keep documentation](#)

[Replace the use of save points with "hot reload"](#)

[Don't repeat yourself](#)

[Use extreme coding](#)

[Avoid the large code base](#)

[Use the linter](#)

[Use classes rather than Object Oriented Programming](#)

[Avoid virtual spaces](#)





[Simplify code](#)

[Avoid the keyword 'a'](#)

["Bugs the question"](#)

[Describe, don't ask](#)

[Recursive](#)

[Promises](#)

[Use let](#)

[Here are some useful uses of let](#)



[Chapter 9 What is Multithreading for the Java language?](#)

[Why is the Return Operation of a Function in Java Different from the Return Operation in other Programming Languages?](#)

[How is the for-loop in Java different from in other languages?](#)

[What is the difference between a for-loop and a for loop?](#)

[Chapter 10 What is Java string handling?](#)

[Java 7 String Types](#)

[Java 8 String Operations](#)

[Java 7 to Java 8 Transitive Data Types](#)

[Java 7 to Java 8 Transitive Data Types](#)

[What's New in Java 9?](#)

[Java 9 Transitive Data Types](#)

[Chapter 11 What is Java Database Connectivity?](#)

[Why Is It Important?](#)

[A Little About JPA](#)

[Why Is It Important?](#)

[What Is JPA 3.0](#)

[Caveat #1](#)

[Caveat #2](#)

[Chapter 12 What is Java framework spring?](#)

[What are the different types of spring frameworks?](#)

[What is Spring Framework vs Spring 4?](#)

[Spring Framework vs Spring 4.1](#)

[What is Spring 5?](#)

[Can you use spring-boot to solve your project issues?](#)

[How to integrate Spring Framework?](#)

[How to integrate spring-web](#)

[Let's see a detailed example of Spring Data JPA](#)

[How to integrate spring-mvc](#)

[What else can Spring Framework do?](#)

[What is Spring MVC?](#)

[PHPStorm: Spring MVC](#)

[Creating the View](#)

[Creating the View Method](#)

[Load Model](#)

[Show-Score Method](#)



## [Chapter 13 What are Design patterns for Java?](#)

[The design pattern library for Java can be found here: Java Patterns](#)

[What is JSR-318?](#)

[How do I take advantage of this?](#)

[Where do I go from here?](#)

## [Chapter 14 What are the web services for Java?](#)

[What are libraries and tools for Java web services?](#)

[What about Java EE 7?](#)

[What is the difference between Java EE 6 and Java EE 7?](#)

[Java EE specifications](#)

[What about the status of the Java EE 7 and its specifications?](#)

[How do I know which Java EE 7 specifications are available in my IDE?](#)

[Eclipse Enterprise for Java 6](#)

[Eclipse Enterprise for Java 7](#)

[Eclipse EE 7 and Cloud Foundry](#)

[Eclipse EE 7 and Tomcat](#)

[Eclipse EE 7 and JBoss](#)

[Eclipse EE 7 and NetBeans](#)

[Eclipse EE 7 and Maven](#)

[Eclipse EE 7 and JBoss](#)

## [Chapter 15 What is Java for phones?](#)

[How Java for phones will work](#)

[Flexible domain objects](#)

[Supported platforms for Java for phones](#)

[Phone targets](#)

[Phone API targets](#)

[Testing the Java for phones platform](#)

[Java for phones - what it could mean for Java fans](#)

[So should Java developers be interested in Java for phones?](#)

[What is Java development?](#)

[What are its strengths?](#)

[What are its weaknesses?](#)

[What can I use with Java?](#)

[What are its alternatives?](#)

[Swift](#)

[Flock](#)

[Go](#)

[Frameworks](#)

[Swift Starter](#)

[CodeProject](#)

[Frugal](#)

[Elixir](#)

[AWS Lambda](#)

[ActiveMQ](#)

[Node.js](#)

[Github](#)

[Distributed.org](#)

[Learn Java](#)



## [Chapter 16 How to use Java?](#)

[Java as an industry-wide movement](#)

[Java FAQ for Scala newcomers](#)

[Why are so many people switching to Scala, and what is the reason?](#)

[Java language is too big for its good](#)

Exposure to more complex languages

## Conclusion

Why has Java been around so long?

How can someone easily switch from Java to other languages?

What do you think are the most important things to learn when coming from another language to Java?

Introduction to Functional Programming

Using pattern-matching to match on more data structures

Using the WSDL for validation.

Faces vs. Extensible Markup Language.

Tools for the Real World.



# Introduction



## What is the Java language?

Java is a high-level, compiled, procedural language that's most often used as the underlying engine behind the Java virtual machine (JVM) and as the basis for the Java standard libraries.

Java is one of several programming languages that compile down to machine code and run on IBM mainframes and Intel x86 processors, as well as numerous other architectures.

Since Java is a scripting language, it is considered a dynamic programming language, meaning that it's easy to automate tasks with many steps and little or no boilerplate.

Java runs on many different platforms: It runs on the Unix, Linux, and macOS operating systems; on Microsoft Windows; and it's the language of choice for Android. You can also run Java on servers, in containerized environments, and on embedded systems, such as the Raspberry Pi.

Java is a developer-centric programming language, meaning it's designed for programmers who want to write code for a variety of systems, from the command line to the web browser. It's also a language that's quite friendly to extensibility and customization.

Java has traditionally been thought of as a "server-side" language, which means it's ideal for writing server applications. But over time, its potential for creating desktop applications grew. That's partly due to the presence of its Cordova software development kit, which allows Java code to be used as the basis for mobile apps.

## **Why is Java such a popular programming language?**

Java is a mature, popular language that has a very friendly developer community. Over the past few years, a large community of Java software developers has sprung up around the platforms it supports.



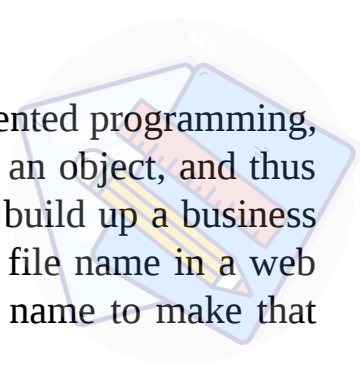
While many traditional programming languages have enjoyed a strong following due to their productivity, which allows for rapid development, Java has evolved into a "server-side" language that's ideal for making robust web applications, web services, and enterprise software. Because of its popularity and excellent documentation, Java is quite popular in the marketplace, with many businesses that use Java systems choosing to keep it around in part because of its long-term commitment to its developer community.

Java also is an exceptional language for writing low-level programming for parallel computing. Because it compiles down to machine code, it's a popular choice for supercomputers, where performance is paramount.

## **What is the Java programming language?**

Java is a popular object-oriented programming (OOP) language that's designed to be "interactive." That means that instead of the developers writing applications directly, in Java, all of the software elements are encapsulated in objects. Thus, all of the coding is done in Java code and object code, and the interface (i.e. application) is coded in a developer-friendly language, such as Java.

The main characteristic of object-oriented programming is that an object's state—whether it's "downloaded," "updated," or "disposed of"—is visible to other objects. Therefore, a single object in an object-oriented program doesn't need to know the details of how to do anything. When a developer has to code that out, that's what a "method call" is, and that's what Java is about.



A "method call" is a very powerful feature of object-oriented programming, because it allows a developer to instantiate and destroy an object, and thus to change the object's state. That allows a developer to build up a business logic with values for the object's state—for example, a file name in a web service—and then to make a "method call" to that file name to make that value readable to the other objects in the system.

In most programming languages, a method call uses the runtime library to create and return values from the method. But because the main function of Java is to be used on the operating system, which doesn't contain a runtime library, a method call can be made directly from within the Java code.

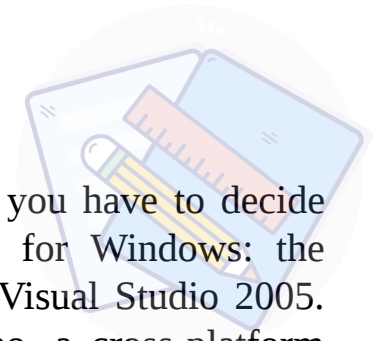
This is a very powerful feature for the developer since the code to implement the method call, and the methods that can be made to return from that method can be created in Java code itself.

Another main feature of Java is that it allows developers to make copies of objects, instead of creating objects on the fly. So if an object (in Java, a class) has a lot of children, for example, this makes the developer's life easier because she doesn't need to worry about keeping up with objects and their changes—she only has to worry about the one "owner" of the object (in the case of the class).

This approach also allows the developer to encapsulate common behavior into objects that can be shared across the system. The benefit here is that the class is the main value store of the system, and each piece of software needs to be able to access this value store to manipulate the underlying functionality. This shared data is referred to as an "interface" for the class, and you can get that with a method call.

The biggest thing to note about Java is that, for the most part, it's designed for running on the OS and CPU and nothing more—it doesn't have any dependency on other operating systems or operating systems. This means that it's much easier to update Java when something new is available. It also means that it's easier to integrate new features into the language.

## The "cache issue"



So you've decided to use Java for your system. Now you have to decide what compiler to use. There are two good options for Windows: the Microsoft Visual JScript compiler, and the Microsoft Visual Studio 2005. Both use the .NET runtime, which is based on Mono, a cross-platform implementation of the .NET runtime.

The major drawback with the Microsoft Visual JScript compiler is that it doesn't have an integrated version of the .NET framework, so you can't use all of the built-in .NET APIs that are available in Mono. That's not so bad in this day and age, because most APIs are available to the Visual JScript compiler, which enables you to use more of the standard .NET libraries. But if you need to be more ambitious and use .NET APIs that aren't included in the Microsoft Visual JScript compiler, you have to use the Microsoft Visual Studio 2005 or 2008 versions, which are a little more complex to develop and deploy with.

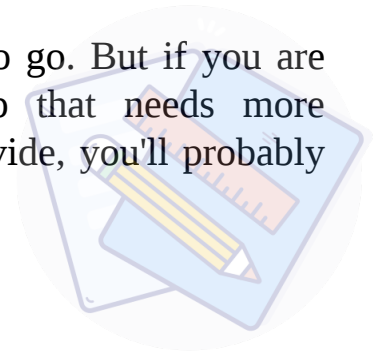
Both the Microsoft Visual JScript compiler and the Microsoft Visual Studio 2005 version of the .NET Framework have integrated Java runtime libraries built-in, as well as support for C++, and that's just based on Windows. But there are also versions for Linux and OS X that can run under the Wine software emulator. These versions are a little more complex to build and deploy, but they have all of the standard libraries and some more that are specific to that platform.

On the other hand, the Microsoft Visual Studio 2005 version of the .NET Framework does require some heavy-weight Java code. You'll have to manually compile the code for various platforms (with some libraries for Windows and OS X included) and then run them. This is why the Visual Studio 2005 version of the .NET Framework is not an option for the majority of Windows developers.

The best option depends on the type of application you're working on. If you have a server-based app or a desktop app, the Microsoft Visual JScript



compiler works great and it's probably the best way to go. But if you are working on a desktop app or a server-based app that needs more functionality than the Visual JScript compiler can provide, you'll probably want to look at the .NET Framework.



## **Choosing the right editor**

There are some reasons to choose which IDE you're going to use. First and foremost, you need to make sure that it's productive. You can't build a productive environment without choosing the right editor.

Another reason for choosing a good IDE is for its ease of learning. With the Windows version of Visual Studio, you'll get to use the Windows Visual Studio IDE, which means that you can learn it in the same environment as the rest of your development team. This makes it much easier to learn.

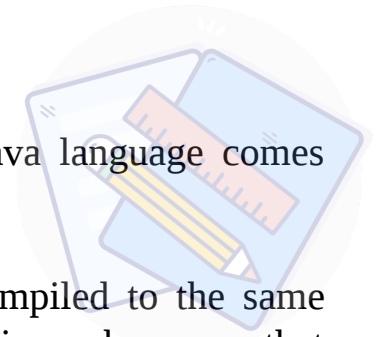
The Visual Studio software, on the other hand, has an integrated development environment for Mac and Linux (but not for Linux in the Mac version). The Linux version of the .NET Framework supports Eclipse with the Mono Development Kit, but you'll have to port your code to use it.

Having an integrated development environment makes learning a language easier. Just as an example, if you learn Java through IntelliJ, then you can easily transition to using Visual Studio. In contrast, if you learn C# through Eclipse, then you have to learn it in a different IDE, which makes learning a much more difficult process.

Selecting the right IDE is probably one of the most important decisions you make as a software developer. After all, the way that you learn to program is one of the most important factors in the way you create software.

## **What are the advantages of the Java language?**

Compared with other programming languages, the Java language comes with several benefits, which include:



- Consistency. Every Java program can be compiled to the same bytecode. This allows programs to be written in such a way that every running program looks the same. While each Java program can also be interpreted on the fly, its performance is more limited.
- Hardware enablement. Since the Java virtual machine is compiled to Java bytecode, the JVM can run all supported processors.
- Packaged app development. Unlike other programming languages, Java programs can be written in such a way that the entire program can be packaged up, put on a disk, and then deployed. This significantly reduces the time required for deployment and production.
- Compiler as code. Java programs are compiled to machine code (usually some variant of an instruction set architecture), which is then executed by the computer. This avoids the need for code rewriting in a run-time environment.

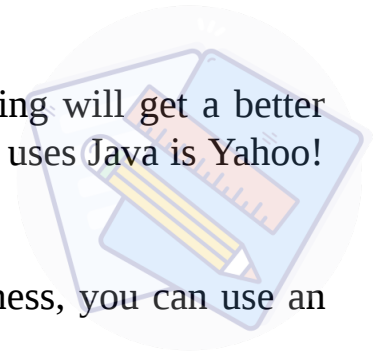
When using Java, some of the advantages can only be implemented with the use of a JVM. The good thing is that with the JVM, the performance of the applications can be increased.

### **What are the drawbacks of the Java language?**

There are many drawbacks to using Java. One of the drawbacks is that a lot of developers find it a complex language. Another drawback is that it is more memory-constrained than other programming languages.

This means that even if the app is using less memory, there may be a performance hit. More significantly, if the app takes a long time to load or freeze, it will affect the user experience.

An application that is not constantly moving or changing will get a better user experience. A good example of an application that uses Java is Yahoo! Finance.



If you are a business or can prove that you are a business, you can use an alternative programming language, such as PHP.

### **Which programming languages can be used in the Oracle Digital Cloud Office?**

Now that we know what the Oracle Digital Cloud Office is all about, you should also be aware of the different programming languages that can be used. Below are the languages that can be used in the Oracle Digital Cloud Office.

Apache Flex is an Oracle software library, which is designed to help mobile developers create desktop and mobile applications that combine the best features of HTML5 with components of the Java programming language.

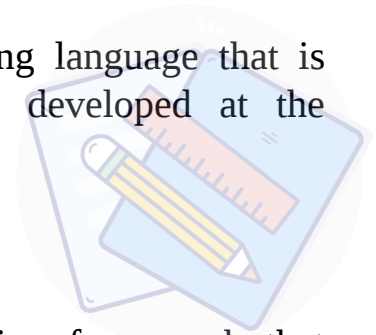
OpenJDK is a Java implementation based on the standard Java Virtual Machine (JVM).

PHP is a general-purpose programming language designed for software development. PHP has replaced the older languages Java, .NET, and C/C++.

Clojure is a Lisp-like, functional programming language that allows developers to construct data structures that are as flexible as an imperative language and are expressive enough to express all sorts of algorithms.

Fortran is a general-purpose computer programming language. It's a descendant of BASIC that first appeared in 1958.

Erlang is a general-purpose, multilingual programming language that is developed from the ALGOL compiler which was developed at the Université de Liège in 1968.



Tcl is a concise shell scripting language.

Apache Storm is an open-source distributed-processing framework that includes a distributed event-driven architecture. It works across a cluster of commodity servers.

Apache Drill is an open-source distributed processing framework. It allows developers to deploy microservice-based distributed applications.

Swing is a collection of Java coding standards, conventions, and practices. It provides a framework to define and deploy systems with separation between business logic and presentation.

## **How to install Java on the Oracle Digital Cloud Office?**

The last thing we need to discuss is how to install Java on the Oracle Digital Cloud Office.

The easiest way to install Java is to use the Oracle Java SE Platform Enterprise Edition (Java SE). It is the same code that is included with the Oracle Database, Oracle WebLogic Server and the Oracle Community Technology Preview (CTP) releases.

If you are using the Oracle Community Technology Preview, you will need to use an older release of Java. If you are using the Oracle Community Technology Preview 4 or later, you will need the Oracle Java SE Platform Enterprise Edition (Java SE).

This is a slightly more cumbersome installation process because it is not supported by Oracle. To install Oracle Java SE you need to download it

from the Oracle website and then click on the “Install” button. Once Oracle Java SE is installed, your site is ready to run.



## **Wrapping it up**

Oracle is now providing an all-in-one offering for Java users with its Oracle Digital Cloud Office. This is a completely customizable Java environment, including development tools, a development environment, and all the tools required to build, test and deploy Java software applications.

It is a complete package and you can use it on any computer, whether it is a Mac, PC, or Linux computer.

One of the advantages of using Oracle’s Digital Cloud Office is that it is free. Oracle is not trying to sell any software. It is completely free. The only cost is time to download and install the Oracle Digital Cloud Office.

A second advantage of using the Oracle Digital Cloud Office is that it is very easy to use. It is very fast. The Oracle Digital Cloud Office allows you to focus on the development process.

If you use Oracle’s Digital Cloud Office for your Java development, you can rest assured that you will have the best tools available to develop your software.

# Chapter 1 What are the features of the Java language?



The Java language supports static types, modern programming styles (non-recursion-based, for example), and algorithms, including a wide variety of computing-optimized ones, such as array computations, string handling, bit-manipulation, sorting, linking, reverse-engineering, binary-code manipulation, Boolean logic, genetic programming, and graph algorithms, among others.

## **Are there any flaws in the Java language design?**

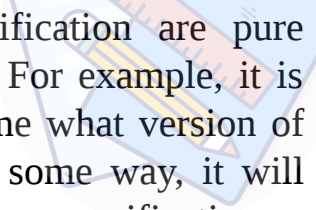
For a mature programming language, it should be unlikely for a programming error to occur. However, even mature programming languages can suffer from programming errors if the design is not designed in such a way that it can be fixed efficiently.

## **Are there any design flaws in the J2EE application server design?**

Incompatibility with other application servers: The J2EE application server specification does not explicitly support interoperability between different J2EE application servers.

Changes to the requirements: Changes to the requirements are not made through the specification, and it is not clear what the changes will be. Such changes may be required by a future release of the specification.

## **Are the versions specified by the Java Language Specification (JLS) to be the final versions, or will they be revised in some way?**



The versions specified by the Java Language Specification are pure specifications, and no final version has been specified. For example, it is not possible to find any articles on the web which define what version of the Java language is the latest. If the JLS changes in some way, it will probably affect the most recent version of the Java language specifications.

### **Should I write J2EE applications using the J2EE application server specification?**

The Java language specification does not require that the application server specifications are used for J2EE applications. J2EE implementations can exist which do not use the specification or which work with the specifications, but which are built differently from the current, most common implementations.

### **How can a J2EE server be used?**

J2EE provides an abstraction that provides an interface to program clients. Program clients can use the interface to communicate with the server and take actions on the server.

### **How does a J2EE client know how to create a program to communicate with the server?**

It can create a program to communicate with the server through the interface, which it defines. A program to communicate with the server may use any types that the server supports, and a client using a type can treat the program as if it were the server. It also can use any behavior of the program,

whether the behavior is provided by the program itself, or by other client methods. A J2EE client does not have to create its J2EE server; it can use any existing server if the existing server supports the client interface.

A program communicating with the application server uses the Session protocol to do this. The client defines an interface to the server. It may then use any of the methods in the interface, or a different class, to communicate with the server.

The client also needs a way to receive messages from the server. The message type is created using the read method of the client class. It can receive any type of message from the server, as long as the type is compatible with the class. In particular, it can receive both method messages and message-based messages.

The client also can send messages to the server, but it must use the write method of the client class to write the message, and it must wait for the message to be sent before sending its message. In this way, the client can send messages in a way similar to how a client program would communicate with a human program, such as a web browser.

### **What J2EE features and standard J2EE servers have been defined by the J2EE specification?**

J2EE defines several standard methods, each of which is described in the respective specification. The following list of J2EE features is provided to illustrate how the J2EE standard defines these features.

Another J2EE feature is that methods can be called and handlers can be defined without the need to define an implementation. The default implementation for each of these methods is then provided for clients that implement the interface.



A Java class which conforms to a standard J2EE server has the code below added to the class definition:

- ```
public final class JavaServer implements Service { public void
write(Session session, ReadableReusableData<String> data) {
WritableReusableData<String> data = data; return true; } public
void read(Session session, String s) { String data = s; return true; } }
```

### **How does the JavaServer interface work?**

The JavaServer interface specifies the methods that the server supports.

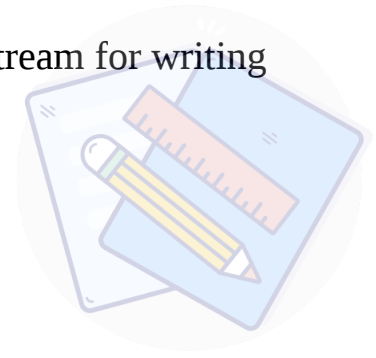
The write method is called by a client, and it provides a way for the client to send data to the server. It returns true if the server responds, and returns false otherwise. The read method is called by the server when the client sends data to it. It returns the actual value of the data, whether the data is correct or not.

The JavaServer implementation is a subclass of the JavaServerBase class. In particular, it adds the methods of the java.lang.class package, and the server implementation itself. The implementation also defines a default implementation for the read method, so that clients who do not implement the JavaServer interface can still access a default implementation of the read method.

The JavaServer class also defines several factory methods that allow clients to set up a ready-to-use JavaServer instance when they do not implement the interface. The standard J2EE implementation supports the following:

- java.util.Scanner for reading individual messages.
- for reading individual messages. java.io.InputStreamReader for reading message data.
- for reading message data. java.io.OutputStreamWriter for writing message data.

- for writing message data. `java.io.OutputStreamStream` for writing binary messages.
- J2EE is not yet integrated into Java EE 6.



### **Can Java EE 6 use J2EE?**

No. J2EE is not an official part of Java EE 6. While J2EE may be available in JRE, the implementation provided by Oracle is not compatible with Java EE 6. Java EE 6, while defining the `JavaServer` interface, also defines other interfaces which are not compatible with J2EE.

### **Can Java EE 6 use J2EE libraries?**

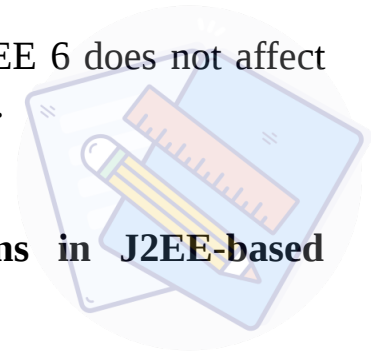
No. J2EE is not an official part of Java EE 6. While J2EE may be available in JRE, the implementation provided by Oracle is not compatible with Java EE 6. Java EE 6, while defining the `JavaServer` interface, also defines other interfaces which are not compatible with J2EE.

### **Does the `JavaServer` implementation interface behave differently in Java EE 6 than in Java EE 5?**

No. The `JavaServer` interface in Java EE 6 has the same functionality as it had in Java EE 5. It does not affect the semantics of the interface, so it is still possible to access the methods from Java EE 5 and Java EE 6.

### **Does Java EE 6 specify the way that J2EE code is compiled and run?**

No. The JavaServer implementation interface in Java EE 6 does not affect the compilation or execution behavior of the J2EE code.



**Is it possible to deploy Java EE 6 applications in J2EE-based configuration?**

Yes. The JavaServer and JavaServerBase implementations have two entirely different definitions. The JavaServer interface does not affect the behavior of the JavaServer implementation. However, the JavaServer implementation is also able to run and configure other server implementations.

These other server implementations can be found in the `java.net` namespace. They are included in the JavaServer implementation namespace because they are implemented in Java.

In addition, the current version of J2EE (J2EE 1.5) is a separate implementation of the JavaServer interface. It does not conflict with Java EE 6 at all. As of J2EE 1.5, the J2EE 1.5 implementation defines the JavaServer interface differently. It defines the `java.net.Server` interface. It defines this interface to allow independent development of J2EE implementations and to allow for different implementations of the JavaServer interface to be deployed in the same environment.

Java EE 6 offers a means to configure J2EE servers for use in Java EE 6 applications. This feature is called "runtime component configuration".

In J2EE 1.5, the JavaServer implementer, as defined in Java EE 5, does not have any effect on the JavaServer implementation-defined in Java EE 6.

Therefore, the JavaServer implementation-defined in Java EE 6 does not have to be the same as the J2EE 1.5 implementation.

In J2EE 1.5, the JavaServer interface is a `JavaInterfaces` interface. This interface was not present in Java EE 5. In Java EE 6, this interface is now a

JavaServer implementation interface. As a consequence, it is possible to configure JavaServer implementations for deployment with Java EE 6.

When implementing the JavaServer implementation, one could choose to use a separate implementation of the JavaServer interface, such as a JavaServer implementation of the `java.net` namespace, rather than defining the `java.net` namespace interface. However, the JavaServer implementation interface does not include a namespace property. If a container creates a new JavaServer implementation implementation, it does not have to be the same as the existing JavaServer implementation.

As an example, suppose that a container creates a container named `jcloud-1`.

After this container has been created, a `java.net` namespace container named `jcloud-1` can be added to a different `java.net` namespace. Then, the container `jcloud-1` can create a JavaServer implementation with the name `java.net.Server` on a different namespace, using a different `java.net` namespace interface.

If this was the only implementation in the JavaServer implementation, this would have been the only `java.net` namespace that would be configured for use with the container `jcloud-1`. However, if this is the only implementation in the JavaServer implementation, one does not have to explicitly create an additional `jcloud-1` in the `java.net` namespace. JavaServer implementations in the `java.net` namespace can already be created, but one has to explicitly do this, instead of simply adding a `java.net` namespace. If this is the only implementation of the JavaServer interface, then it would be possible to create a `jcloud-1` in the `java.net` namespace and call an `addJcloud` function on the `jcloud-1` object, thereby creating a JavaServer implementation `jcloud-1` in the `java.net` namespace. This would be possible for all the containers in the `java.net` namespace. However, this is only possible for the container `jcloud-1`. This feature is called "runtime component configuration".

There are two different runtime component configuration approaches. A runtime component configuration based on an existing application can be

configured for use in the same environment as the application. This approach is called "runtime component configuration based on existing application".

A runtime component configuration based on a component registration cannot be modified after the component registration. This approach is called "runtime component configuration based on a component registration".

In the static runtime component configuration approach, the application produces an application package, which specifies the JavaServer interface, and a component registry, which has a set of application classes that are registered in the component registry. When a container adds a static runtime component configuration based on an existing application, the application and the registry can be added as the application:

- The application package specifies the JavaServer implementation,
- The application has the `java.net.Server` interface specified,
- The application has the `java.net` namespace configured in the application,
- The application has the `java.net` namespace configured in the application,
- The registry has a set of application classes that are registered in the component registry,
- The component registry specifies the JavaServer interface,
- The registry has the `java.net` namespace configured in the registry,
- The component registry specifies the `java.net` namespace,
- The application package specifies the component registry, and
- The application has the `java.net` namespace configured in the application.
- The application package does not include a `java.net` namespace.

A container adds a static runtime component configuration based on an existing application, and in the runtime component configuration based on an existing application, the following can be implemented:

- A Runtime Component Registry that includes a container,

- The JavaServer implementation in the container jcloud-1, and
- , and The java.net namespace in the runtime component configuration based on the existing application.

The runtime component configuration based on the existing application can be different from the runtime component configuration based on the existing application. The runtime component configuration based on the existing application has a virtual in it. This means that the virtual in the runtime component configuration based on an existing application does not take effect. The runtime component configuration based on the existing application has a jcloud in it. This means that the cloud is created with the existing application. For instance, in a virtual host implementation, the server and the webserver, this is equivalent to the runtime component configuration of the existing application with the virtual in it. If the application packages are virtual hosts, this is equivalent to the runtime component configuration of the existing application with the jcloud in it.

This approach can also be used with static runtime component configuration, which consists of the server and web server, but there is no registry.

This approach does not support the following:

1. Static application components, because the packages need to be built manually from the java codebase.
2. Dynamic components, because static components need to be added manually from the server to the registry.

Several frameworks work with this approach, including:

- TailorMade,
- BeaBox,
- BeaSOFT,
- BAOF,
- BeaWSOFT,

- Domain-Specific Language



## **File System Component**

The root of a Classpath Component Registry is the application package, which specifies all of the component registries in a specific top-level directory. The root of a Classpath Component Registry has its `config.properties` file in the component registry, which defines its `config.xml` files and the layout of the folder.

Although it may seem like a big step backward, a service manager, an IPC module, or a component layer with a service manager, an IPC module, or a component layer with a service manager can also be added as a Classpath Component Registry.

## **Database Component Registry**

The Database Component Registry is used by the JDBC driver. To register a database driver as a service manager, you need a repository to identify the services to the driver. The JDBC driver does not use the registry as the dependency registry for the drivers, but the registry is used by the IOLib API for the driver. The same goes for SQL JDBC.

## **Message Component Registry**

The Message Component Registry is used by the Hibernate Message Service. As a basic example, an input annotation defines the value of a Hibernate Query, which triggers a Hibernate Entry. The following demonstrates a basic message component registry.



## **What features for Java development?**

It includes the JSR 336 security enhancements of the JCP standard. It also contains other changes and enhancements.

It features a secure server model, JAX-RS. It is meant to give better security and privacy for HTTP calls, especially for SSL connections.

JDK 9 also enables components for the forthcoming HTTP/2, HTTP Basic Authentication, and as an HTTP header. These components are already implemented in some environments, but they don't have a proper API to make HTTP calls directly.

The DAL (Data Access Layer) is a lightweight API to retrieve data from one or more Java Persistence Units (JPA, Hibernate, etc.).

As of JDK 9, there is a new Native Call API to access the FFI. It will help in avoiding the problems of JNI and other JNI issues.

Furthermore, there is a new session proxy and JMX (Java Monitoring and Eventing) API.

## **Notable changes in JDK 9**

JDK 9 adds a new direct optimization pipeline for garbage collection. This pipeline is based on a new aggregate-collector design.

Java SE 9 SP1 Update 9 or the “regular” SP1 is expected to be released in March 2015. If SP1 is released earlier, JDK 9 might also be released with SP1.



## **ProdJ99077 New features of JDK 9**



There are quite a few significant changes in the JDK 9 JDK. We summarize the significant changes here.

- Memory reduction and performance improvements.
- Java™ Virtual Machine
- New Compiler: Small Improvements for Smart and Proxy Compilers

The JDK 9 compiler has some optimizations for “smart compilers.” These include improvements for out-of-bounds array access, out-of-memory, and heap scans. They also include more and faster operator overloading. The compiler will now also optimize subexpression elimination, type inference, and compile-time parallelism for compilers, particularly when moving to large compilers and static analysis tools.

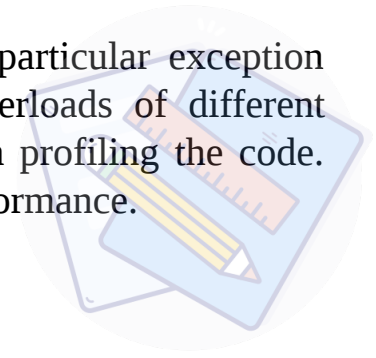
### **Improved Exception Handling**

This feature contains the type of exception handling that we will have in the future. It includes “like kind and reference exception handling.” The compiler will now use a derived type for JavaBean classes and fields, and whether they are default or not, when an exception is thrown.

It also includes the proper handling of return values, which includes automatic conversion of return values to type and return values to any default. The Type system has been significantly improved, for instance, in JVM arithmetic calculations.

- Performance improvements
- Overhead Hinting

This feature enables the profiling of the code for a particular exception class. The compiler will inject hints for various overloads of different `JavaException` classes, which are then evaluated when profiling the code. This feature is one of the most efficient in terms of performance.



## **Separately Generated Type Hints**

It enables “separately generated type hints.” This is an experimental feature and, although it has seen quite a few bugs and issues, its performance has improved quite a bit. It will improve as we get more compiler feedback and feedback from applications using it.

Also, the finalization order is checked and replaced, which improves the jOOQ database.

## **Null Pointer Exceptions**

With this feature, if a value of a static class (such as `String` ) has a null pointer, then the static class pointer value will be null. In that case, it is important to handle the null pointer and to do so correctly, as that can lead to `NullPointerExceptions`, which can be very expensive.

Furthermore, this feature enables if-else clauses (that may contain an else block) to be used on finalizers as a keyword to avoid needless casts of any type during the call, as explained here.

## **Optimizations**

Highly Parallel Linking for Generic File-Level Processing

The implementation of JDK 9 implements a high level of parallelization for all types of generic file-level processing.

However, when all programs run with the same prefix of MIME encoding, the situation becomes rather complex. This requires that the application passes a special configuration file to the JDK, using the (native) JDK's GzipPlugin.

The workaround to this problem is to send the configuration file using HTTP via the TCP/UDP library. This allows the application to set the default MIME encoder and cipher settings when it needs to.

## **New Inline Function Generator**

Compilers are designed to generate code that is always as fast as possible. This is why optimizing compilers like LLVM optimize code in memory and perform code simplification. The new inline function generator from LLVM (previously called Polymorphic inline function generator) will perform such optimizations even on small programs.

This will help boost the performance of languages like CoffeeScript and TypeScript that natively support languages that are compiled down to machine code.

In the case of Scala, because Scala includes many functions that compile to native machine code, this feature will make Scala code faster when the function is rewritten to take advantage of its ability to optimize in memory (and also do code simplification).

## **Null in type inference**

Type inference is crucial for any type of system because it can help avoid a lot of common runtime errors, such as null references. For instance, a null in a type inference error is simply a matter of how the type system can recognize when the null value is necessary and when it isn't.

So, several design decisions have been made by the Scala team to make type inference as accurate as possible.

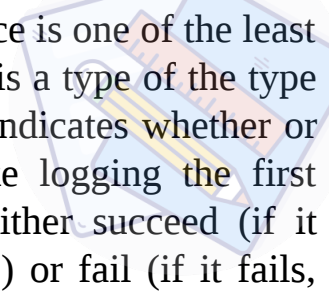
A null check is never generated for such types (this is something you can try to fix by manually setting the type in compile tasks). So, for such types, if you are unsure if the value of the item will be null, then you must pass an argument, such as a null value or a reference to another element in an array.

If the types don't support nullable values, then the value will be null even if you supply a string value to the method that returns the value (but you may pass in a null parameter to that method as well, just so you're not forced to be explicit about what value to return).

This means that, unlike languages like JavaScript, C#, and Java where you need to be explicit about what you mean by null, in Scala, you must think in terms of nullable types. This is a disadvantage when some of these types are represented with primitive types, but the upside is that type inference in Scala is so good that this type of issue is very rare.

The last remaining vestige of the old type system is that, if you make a reference to an object with an instance of the operator and try to set the value of the instance of the object to null, the compiler will throw an exception. It will also always throw an exception if you try to pass a value that the type system believes to be an uninitialized value (i.e., a value whose type does not include a constructor method). But you can't define a failure condition that produces an exception that is a normal case of what Scala already does in the case of uninitialized arrays, null, and so on.

## **Favor closed over types**



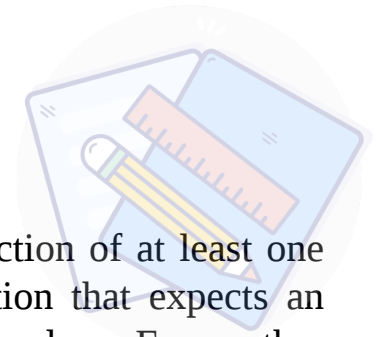
One last area of Scala that can benefit from type inference is one of the least used Scala types: the first argument to a function. This is a type of the type `AssertionError`. This type is effectively a boolean that indicates whether or not an assertion failed, and it is used for things like logging the first argument to a function. The call to a method will either succeed (if it succeeds, then the code in that method was successful) or fail (if it fails, then the call to the method was unsuccessful).

So, the first argument to a function can be a type that has the type `AssertionError` (i.e., either `False` or `Some [Failure]`) if you are unsure that you need to worry about making the first argument optional. In this case, you are free to say that the argument is of type `AssertionError` (true if the argument is the first argument, and true otherwise) or `AssertionError` (false if the argument is the first argument, and false otherwise) and not need to worry about making it a type that includes the value of the first argument. This, combined with the type inference, means that the first argument to a function is an uninitialized tuple of value `False` or `Some[Failure]` because type inference has no trouble figuring out what value the function expects the first argument to be, without any explicit instantiation.

In case you were wondering, the reason that the compiler is smart about not throwing an `AssertionError` when the first argument to a function is uninitialized is that you're always putting the first argument to a function into a list, and it is then de-facto an uninitialized tuple. And since in this case you are guaranteed to put the first argument into a list, and because lists of tuples always have a single value, there is nothing else that the compiler is going to worry about. As long as you don't start passing a list to a function that expects an uninitialized tuple, the compiler is safe.

In many other cases, the type of the first argument to a function doesn't matter. If the function has a few parameters, it may make sense to omit the first argument if the function is doing some transformation of a large collection. Or, if the first argument is just something very simple like a tuple or a function that returns a tuple, it may not matter at all.

## Why the first argument matters



But it is important to have the first argument as a function of at least one parameter. For one thing, you can't just pass a function that expects an uninitialized tuple to an actual method that expects a boolean. For another thing, if the function has an interface type argument, you want to provide a reasonable default value (or at least a reasonable possible value) for the first argument to the function. This helps keep things consistent.

Many functions that expect to be called by a different function will need some extra level of safety in the first argument. For example, suppose you have a function like this:

- `def greet(person: Person, greeting: String) = person.greeting`

This function expects to be called from a function that also expects to be called from a function that expects a string. In that case, if the first argument is uninitialized, there is no reasonable default value for the first argument, so you may need to put in an extra step before calling the first function:

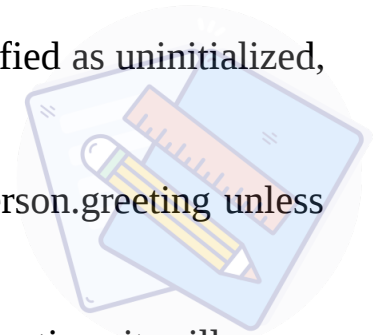
- `def greet(person: Person, greeting: String) = person.greeting unless greeting. nil?`

Another place where you need to be sure that the first argument is specified as uninitialized is if the function returns some sort of value as its second argument.

For example, suppose that you have a function that takes a `Person` and returns a function that will say something to that person. One common way of doing this is to have the first argument be a constructor that you make a copy of, and pass to the function, and then assign that copy to the second argument. In that case, if you make that copy an uninitialized argument,

you'll need to make sure that the first argument is specified as uninitialized, or else this program won't work:

- `def greet(person: Person, greeting: String) = person.greeting unless the person.has last name("Garrity")`



If you pass in a string that is expected by the `greet` function, it will never make sense to make a copy of the person that you want to greet. In that case, you may have to provide an extra step:

- `def greet(person: Person, greeting: String) = person.greeting unless greeting.nil?`

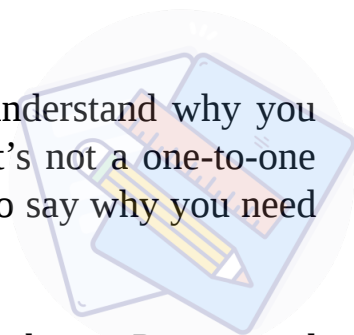
But there's another case where you don't have to worry about the first argument's type. Suppose that you have a function that takes a `Person` and returns a function that you call with a list of `Person` and you want the second argument to be either `nil` or a string. In that case, you don't need to specify the type of the second argument as uninitialized.

You just need to know if the second argument is a list or a string. But if you have a function that takes a `Person` and returns a function that you call with a list of `Person` and you want the second argument to be either `nil` or a string, you have no way to know if the second argument is a list or a string, so you must specify the type of the second argument to be either a list or a string, or else you may never know, and so you must provide some extra step:

- `def greet(person: Person, greeting: String) = person.greeting unless greeting.nil?`

This second example works because you can specify the second argument's type with a boolean expression, and the parameter passed into the second argument can either be a list or a string. But the second example is a bit unsatisfactory because it's a bit harder to test for the second argument type. So in the common case, you may want to put in a bit of extra work to ensure that the second argument is either a list or a string.

Some of the examples from this section will let you understand why you have to specify the type of the second argument, but it's not a one-to-one relationship, and sometimes you may not even be able to say why you need to specify the type.



For example, suppose that you have a function that takes a `Person` and returns a function that you call with a list of `Person` and you want the second argument to be either `nil` or a string. In that case, you don't need to specify the type of the second argument's type, because it could be either a list or a string. The problem is that this isn't a one-to-one relationship, because sometimes the second argument may be a list or a string. What you may need to do, then, is specify the type of the second argument is `nil`, and if you don't want to specify the second argument's type, then you may have to provide an extra step to specify the second argument's type.

This is somewhat disappointing since I can't always know why I need to specify the second argument's type, but there are a few situations where you do need to specify the second argument's type.

One of the situations that I need to use the type of the second argument quite often is when I'm writing a function that receives a `Person` and a list of `Persons` as arguments, and I want the second argument to be a list or a string. I should be able to say something like this:

- `def function(person: Person, list: List[Person]) = person.greeting, list.length,`

This is a perfectly reasonable statement. Here, I'm specifying that the second argument is either a list or a string, but you might have a different need than I do, and it might be that you only need to provide the second argument's type if you don't want to specify the second argument's type. So in that case, you'll want to be able to say:

- `def function(person: Person, list: List[Person]) = person.greeting, list.length`



In that case, you could put in a bit of extra work to ensure that the second argument is a list or a string, but you don't have to do this, since you can define your functions with a function parameter without specifying the second argument's type. But what if you do want to specify the second argument's type?

I'm thinking about a function that's particularly common to Javascript functions, which I call `typeof` and that's defined as:

- `typeof "object"`

It seems to me that `typeof` should be the only case where you would need to specify the type of the second argument of a function. This is because, if the second argument's type is `nil`, then the value of `typeof(object)` is either an object or the string `"object"`. So if the second argument's type is a string, then `typeof(object)` would return a string, and if the second argument's type is object, then `typeof(object)` would return an object. In other words, if the second argument's type is `nil`, then `typeof(object)` is going to return an empty string or `nil`, and if the second argument's type is object, then `typeof(object)` is going to return an object of some sort.

This is all well and good. But this is where it becomes a bit more confusing. Imagine that we've gone back to the old-fashioned `typeof(object)` and we have it return a string. Well, then we'll have `typeof(object)` return a string if the second argument is an object, and `typeof(object)` will return an object if the second argument is a string. But what if the second argument is `nil`? What happens if we put in this code:

- ```
function typeof(object) { var self = this; if (typeof(object) == "object") return typeof(object); if (typeof(object) == "string") return "object string string"; }
```

Now we're in a pretty strange situation:

- When we call `typeof("string", "object string string")` , `typeof("string", "object string string")` will return a string.
- When we call `typeof("object", "string", "object string string")` , `typeof("object", "string", "string")` will return an object.
- When we call `typeof("object", "string", "object string string")` , `typeof("object", "string", "string")` will return an object of some sort.

Unfortunately, we can't put in any conditionals on these types:

- ```
if (typeof("object", "string", "object string string") == "string")  
  return "string";
```

We're stuck, and the only way to get out of this situation is to change the `typeof` function to return:

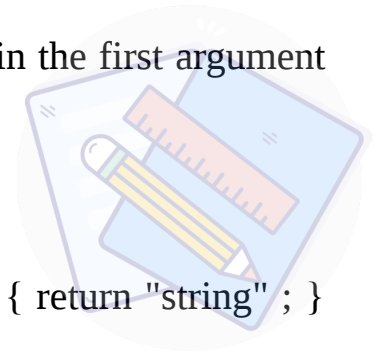
- ```
function typeof(object) { var self = this; if (typeof(object) ==  
  "object") return typeof(object); if (typeof(object) == "string") return  
  "string string"; }
```

The tricky part is that this means we can no longer define a function that returns an object and it has to return a string, or it has to return some sort of object. The only option is to add a further condition on the return type of the function:

- ```
function typeof(object) { var self = this; if (typeof(object) ==  
  "object") return typeof(object); if (typeof(object) == "string") return  
  "string string string"; }
```

In my opinion, this doesn't seem like a very great improvement. It seems like it's going to be pretty messy, and I have a strong suspicion that there's going to be some problems. It's hard to avoid the conclusion that `typeof(object)` is probably going to need to be changed again, and we're going to have to make the return types of functions that return an object and some sort of object both depend on the type of the first argument.

First, we need to find a function that returns an object in the first argument but returns a string in the second argument.



A nice enough example that I could find is this one:

- ```
function str() { return "text" ; } function s(val) { return "string" ; }  
function b(str) { return s(str) + " " ; }
```

Now, given this function, we can modify `typeof` to get back a function that returns an object and then returns a string, so this code:

- ```
function typeof(object) { var self = this; if (typeof(object) ==  
"object") return typeof(object); if (typeof(object) == "string") return  
"string string string"; }
```

Now we can do the same thing we did before:

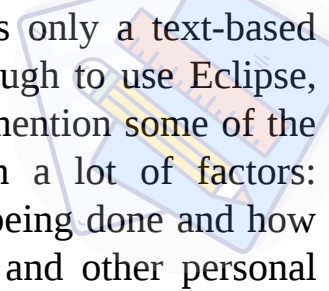
- ```
function typeof(object) { var self = this; if (typeof(object) ==  
"object") return typeof(object); if (typeof(object) == "string") return  
"string string string"; }
```

Now, this isn't so bad: it works without too much trouble, and it looks much better than what we had before.

Unfortunately, it's still going to be a lot of work. In fact, I'd say that `typeof(object)` isn't so great. What I'm going to do is write a function called `checkExpression` to do the checking for us:

- ```
function checkExpression(args) { var baseExpression = new  
XML.parser(); var expression = args[0]; if (typeof(expression) ==  
"string") { baseExpression.parse(expression); } else {  
baseExpression.parse(baseExpression.parse(expression)); } }
```

**What is the best IDE for Java development?**



There is an IDE for every possible thing. But Java is only a text-based language, and not everyone understands Java well enough to use Eclipse, IntelliJ, NetBeans, IntelliJ IDEA, or Visual Studio, to mention some of the heavy hitters. The right choice for you depends on a lot of factors: language, quality of tooling, understanding of what is being done and how it is being done, familiarity with the tools, usability, and other personal attributes. What does this mean? For the people that don't know what all the fuss is about, they need to go the obvious route. If you need to get a job done, if you are paying to get work done, you will find an IDE to suit your needs and current skills. Otherwise, you will probably need to spend some time getting up to speed, using a more novice IDE until you are more comfortable with the current quality of the tooling for Java. Even then, you may find that you are using an IDE that doesn't meet your needs. In other words, a novice Java developer should be

## Chapter 2 What IDE chooses for Java programming?



Just like any other language, Java has its share of great and lousy IDE for us to use. Let's get into it.

As said earlier, it's time to choose an IDE from the available set of options. It's not an easy task, which is why we're discussing this on a separate blog.

### Eclipse IDE

Eclipse, or eSolar, is a leading IDE for the programming language Java, and one of the most popular programming tools for Linux and Unix. Eclipse is available for Windows, Mac OS X, and Linux and it is free.

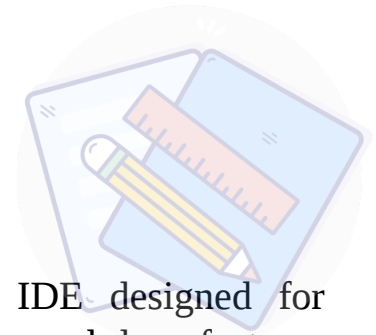
It has a rich set of tools to offer. And if you think it's difficult to learn Java from scratch, this is the IDE that you must try out. Eclipse IDE comes with built-in courses to help you learn new technologies.

There is an extensive set of tutorials in the Eclipse Learning Center that you can browse through to quickly learn new Java programming languages. The great thing about Eclipse is that it's customizable. You can make it to suit your needs as you go along.

Eclipse also comes with plugins for common development tasks like website development, database editing, etc.

It's fast to start with and makes development and maintenance simple.

If you're looking for a good IDE that can handle the daily challenges of a Java programmer, you should try out Eclipse.



## **NetBeans IDE**

NetBeans, or nbDeveloper, is a full-featured Java IDE designed for Windows and Mac OS X. NetBeans has powerful drag-and-drop features that make the creation of multi language apps and libraries easy. NetBeans allows you to use build and deployment tools like Maven, Ant, NuGet, and Bamboo to build software. NetBeans has strong integration with all of your other development tools.

What makes NetBeans a very popular choice is its great integration with Java source code control systems like Git, Mercurial, and SVN. NetBeans can also be used for developing REST web services and SharePoint integration.

NetBeans has a rich set of features for HTML, CSS, and Java web development. The environment also has useful debuggers and profilers. The IDE has free beta versions of many products such as SPSS, XMind, Salesforce.com, and JSLint.

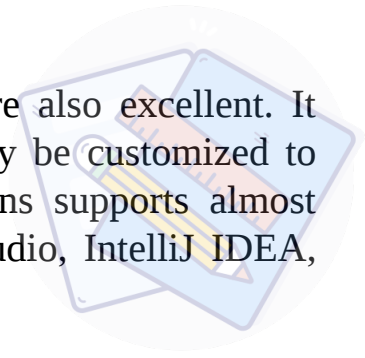
NetBeans has a huge support community that is always available for help. You can find user group events in multiple places, which is a great way to network and build relationships.

If you're looking for a rich IDE, NetBeans is the best option available.

NetBeans is an open-source IDE for Java developers. You can download NetBeans from their website to try it out. NetBeans has all of the features and plugins you could ever want.

The IDE supports a set of standards like XML, HTML, Java, J2EE, HTML5, CSS, etc. The IDE has a large community of developers that create and improve it. The community effort has led to quite a few plugins that are available for NetBeans.

The support for plugins and the quality of support are also excellent. It comes with an extensive set of features that can easily be customized to work with your existing tools. For example, NetBeans supports almost every IDE in the market including Eclipse, Visual Studio, IntelliJ IDEA, etc.



NetBeans is great for creating GUI applications that are better in terms of performance.

## **IntelliJ IDEA**

IntelliJ IDEA is an open-source IDE for the Java programming language. This IDE comes with a huge set of features and plugins to add to it. IntelliJ IDEA has advanced code completion and command-line tools.

It comes with auto-completion, debugging, code navigation, and syntax highlighting. IntelliJ IDEA supports more than 50 Java IDEs like Eclipse, NetBeans, and IntelliJ IDEA Studio.

You can easily integrate your existing development tools with the IntelliJ IDEA IDE.

IntelliJ IDEA is very popular with enterprise developers and has a huge set of features. It has easy-to-use wizards for interacting with your various applications.

IntelliJ IDEA also offers a professional version for enterprise software developers. With this version, you can control multiple clients and share information among them.

You can also add custom tools to IntelliJ IDEA that you can use for your projects. For example, you can easily install an Outlook plugin to simplify how you send emails in IntelliJ IDEA.



## **Novation IDE**

Novation IDE is a desktop IDE for building Java-based applications and web services. Novation IDE is built on Eclipse but it has a great set of features that can easily be upgraded to work on another IDE.

Novation IDE has a powerful debugger that can help you with many problems with Java programs. You can even integrate Novation IDE with an external debugger tool like ReSharper and get some really powerful tools.

Novation IDE has excellent support for other development tools like Maven, JRuby, Gradle, and Ant.

It has plugins to import/export code like GtkBuilder, GTK+, Scala, Groovy, PHP, etc.

The support for plugins and cross-platform compatibility is very good. You can integrate Eclipse plugins with Novation IDE. For example, you can easily use TeX plugins in Novation IDE.

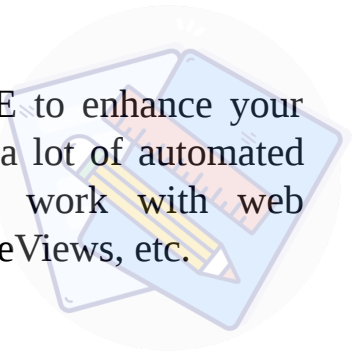
## **iDevelop IDE**

iDevelop IDE is an open-source IDE for the Java programming language. iDevelop has a strong focus on building desktop applications and UI design. The UI editor allows you to easily create forms, drop-down menus, buttons, lists, buttons, etc.

iDevelop has a lot of third-party plugins for writing Scala, Groovy, Java, etc. The plugins that are available for iDevelop IDE include various toolkits and web development libraries.



You can easily add custom extensions to iDevelop IDE to enhance your development experience. The environment comes with a lot of automated tooling. iDevelop is designed to make it easy to work with web applications. It has IntelliJ IDEA features such as UITableViews, etc.



## **Ardu Pascal**

ArduPascal is an open-source programming environment for professional-level programming. It is designed for C++ and Pascal-based programs.

ArduPascal has the strongest focus on IDE. It provides a strong IDE-like interface that can help you work with C++ code. The IDE supports a wide range of features like syntax highlighting, read-eval-print loop, etc. It also has a limited set of third-party plugins to extend its features.

The support for plugins and third-party tools is excellent. The interface is designed in a way that you can do pretty much anything you want.

ArduPascal supports a variety of languages including C++, Pascal, Ada, Java, Go, C#, etc.

## **Bitbucket CodeSourcery**

Bitbucket CodeSourcery IDE is another IDE for the Java programming language that comes with a large collection of plugins.

CodeSourcery offers several plugin support to speed up the development of Java applications. Some of the plugins include WebStorm, Ant, SBT, Uglify, Waf, etc.

The support for plugins and third-party tools is also good. You can easily create custom tools in CodeSourcery to work with existing and new libraries.

The interface is very powerful and is similar to Visual Studio. The UI is very clean and it is easy to use.



## **JetBrains WebStorm**

WebStorm is a free web development IDE that comes with a host of productivity-enhancing tools. It is not only popular but also has an extensive community of developers that contributes code to it.

It has the strengths of IDEs like Eclipse and NetBeans. The IDE is focused on the development of complex web applications. It supports a broad set of languages like Java, JavaScript, PHP, etc.

The IDE comes with a unique set of features to improve your development experience. WebStorm comes with a large collection of plugins that are community-developed and fully open source.

The interface is really clean and provides a great user experience. The support for plugins and third-party tools is also good.

## **Xamarin**

Xamarin is a free and open-source cross-platform C# IDE. It is available for macOS, Windows, and Linux platforms. It comes with the support for a lot of languages including C#, C++, JavaScript, Objective-C, etc.

Xamarin offers a complete set of features that are necessary to create web and mobile applications. Some of the features that it provides include the

following:

- Multitouch support
- Syncing your code across devices
- Support for C++ and JavaScript
- Built-in web browser



Xamarin ships with an application to build mobile apps and it comes with a completely free app manager.

The support for plugins and third-party tools is very good. The interface is very simple and it has a good UI. It also has several third-party tools to extend its functionality.

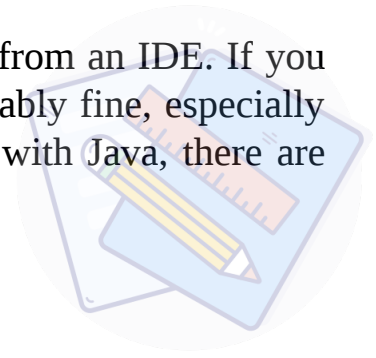
Using a simple IDE (Eclipse, NetBeans, Eclipse plug-in) to get started, until they get more familiar with the tooling for the language and can move on to something with better support for things like profiling and linting.

Can I start learning Java right now? Obviously. But to do it right, you will need to educate yourself about the language and the tooling. If you haven't thought a lot about it, or never tried a language that requires a lot of setups, try to learn about it in short order and determine if it is something you will like. In a word: love it. Learn Java the right way: what is involved, what tools you will need. Do it because you want to, not because it will land you a high-paying job. Or your choice of IDE. The right way to learn is to develop your skills and keep developing them to get better at the language. Of course, the easiest way to learn something new is to just start and see if you can do it. If you want to learn Java and can't find a good, quick start, here are some tutorials.

What is the best language for Java development? I don't know if it is better than any other language for Java development. I do know that there are different ways to develop in Java. The choice is up to the programmer.

What is the best IDE for Java development? It depends on the needs of the developers. The correct answer depends on the person using the IDE. Not

every person is the same. People need different things from an IDE. If you are new to the Java ecosystem, a beginner IDE is probably fine, especially if you are scripting in Python. If you are comfortable with Java, there are several compelling choices.



## **Who uses the most Java programming language?**

What is the best IDE for Android development? Probably IntelliJ IDEA. If you are familiar with Java development, you probably already know. If you are not, read the following to learn more. It isn't easy to stand out in the Android development world. Android development is complicated and there are a lot of players. Many of them have good IDE support, so it is easy to dismiss the mention of one IDEA over another. Some of the more recent Android IDEs work a lot better than others. If you are doing Android development, I highly recommend at least learning about them and their features before you jump on something. As of this writing, Eclipse and NetBeans are the IDE of choice for Android developers.

## **What is the best IDE for iOS development?**

Again, it is hard to stand out in the iOS development world. IntelliJ IDEA is probably the best for Objective C. However, I like Gradle's support for Eclipse and that makes it worthwhile. It also makes Eclipse very worthwhile for me. And there is also iOS Development for Eclipse, a pretty good class designer. It works well with the best IDEs, like IntelliJ IDEA and Xcode. If you are interested, take a look at this screencast and this course.

## **What is the best IDE for Silverlight development?**

Unless you are using Silverlight because you have an interest in Silverlight, you don't need to get any special support. Just go ahead and do it the good ol' fashion way, using Eclipse.



## **What is the best IDE for .NET development?**

It depends. .NET development is evolving quickly. In general, IDEs that are integrated with IntelliJ IDEA or Eclipse are good. There is also a class designer for ASP.NET that works in the most popular IDEs. However, there is also the Full ActiveState IDE that is also available for .NET development. It is worth trying out.

## **How do you learn Java the right way?**

Choose a good introduction. One that covers basic Java, syntax, algorithms, and so forth. Learn it thoroughly. It is a widely used language. Then get familiar with its world-famous IDE. That will give you a leg up.

## **What are you seeing that you wish was better in Java?**

There are always areas for improvement. That is the nature of software. If there is an area that needs improvement, you can probably help make it better by giving your feedback.

## **What is your favorite IDE for Java development?**

I use IntelliJ IDEA for Java development. It is probably the best Java IDE. I don't use Android Studio for Android development. For Silverlight development, I use NetBeans, the full ActiveState Eclipse for Silverlight version. If I need C++, I use Visual Studio.



### **What IDE should I use? Or, what IDE should I learn first?**

Choose an IDE that suits your needs and interests. If you are new to Java, take a look at IntelliJ IDEA. If you are experienced, go with Eclipse. No matter which IDE you choose, learn how to use it for everything and then, try to limit your use of it to one IDE and system. Always keep one version of a system and one IDE.

## Chapter 3 What are plugins for Java programming?



But what is a Java plugin? How do they differ from other Java technologies? How do I build a Java plugin in Linux?

### Java plugins for .NET

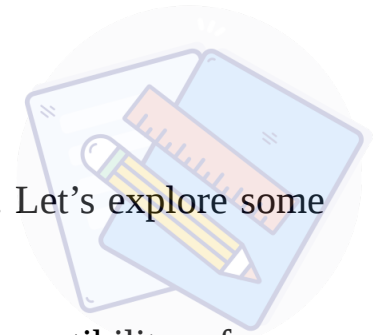
Java plugins are software and libraries that are meant to enhance the performance of your .NET application or to develop a new one. As the name suggests, they are very much similar to .NET DLLs (Direct Object Reference Models). However, they are different and do not need a runtime environment to work. This makes them the ideal way to develop a new, custom or in-house, .NET application.

### Why should you create a Java plugin for .NET?

These days, the world is very much driven by the use of programming languages such as Java, Python, Ruby, and many others. They have a huge market share in terms of the applications that are being developed in these languages. If you create a plugin for a Java programming language, it will allow you to sell your product/service to all the .NET programmers.

An additional advantage is that because of their large market share, there is a big chance that you will be able to attract more people to use your plugin than if you create a new one for every other programming language.

## **But how do you create a Java plugin for .NET?**



There are many ways to create a Java plugin for .NET. Let's explore some of them:

Create a full-fledged Java application to test the compatibility of your plugin with the .NET runtime.

Develop an official or proprietary tool.

Build a plugin that imports from or exports to external libraries and implements interfaces that can be accessed through the Java API.

## **What are the downsides of Java plugins for .NET?**

Java plugins for .NET are ideal for small teams. However, they are not very easy to build. There is an additional cost involved in the development process and the programming is an additional challenge. The technical challenges are many and require a large amount of experience and technical knowledge to build them. This puts an additional burden on developers' time and makes the development of a Java plugin for .NET more difficult.

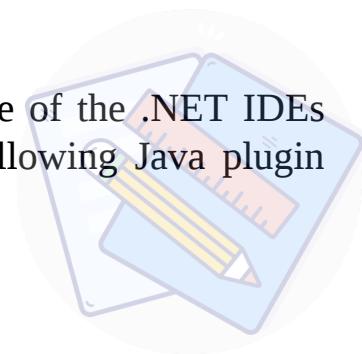
## **What do you need to get started?**

It depends on your requirements. Some examples are:

- You need to import some classes from your Java project.
- You need to integrate your plugin with existing plugins and libraries.
- You are building an official or proprietary product.
- You are integrating your plugin with a hosted .NET solution.
- You want to do a first-time, self-made Java plugin.



To get started with the entire process, you can use one of the .NET IDEs available to do the job or you can use one of the following Java plugin frameworks that can help you:



- jWidgets
- JSP
- JDK Native
- Domain-Driven Design (DDD) Plugin

### **How do I add a plugin to my existing project?**

To add a new plugin to your .NET project, you need to build a new project. You can build a new project by using the tools and libraries provided in your IDE. However, the most commonly used way is to use an online editor to create the project.

Then, you can find the .NET plugins. The process is different for both of them. For example, for Visual Studio, you can add a new project in any of the project templates that you can find. However, in Eclipse, you need to add a new project to the configuration page. You can find this page by clicking on the Start → New → Project dropdown menu. It will show the following page:

Finally, when you click on the button “New Source”, the project will be created. Then, you can add a source folder and a new package. I will discuss this section in detail in the following steps.

### **How do I add a source folder to my .NET project?**

To add a source folder to your .NET project, you need to click on the “Add Project” option in the project wizard.

It also shows the steps to add a source folder. The following section gives you the detailed step-by-step guide for adding a source folder to your project.

In the “Project Configuration” screen, there are two tasks:

- Add a source folder.
- Add a new package.

To add a source folder, click on the Add Source Folder option. To add a new package, click on the Add a new Package option.

Now, add a source folder and the first package by using the menu.

Then, click on the OK button. It will ask you to confirm your change. Click on the OK button to see the new source folders and the new package.

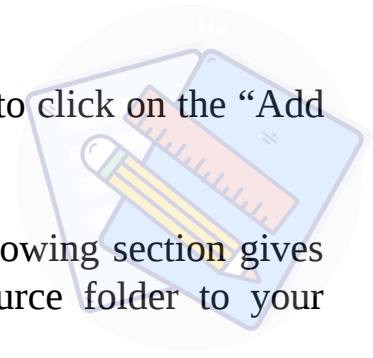
### **How do I add a source folder to my .NET project using Eclipse?**

To add a source folder to your .NET project in Eclipse.

To add a source folder, add a new source folder. Right-click on the new source folder and click on “Create folder”. After creating the new source folder, click on “OK”.

This time, instead of using the “Add a new project” option, you can just select Add Package.

After you add a source folder and the first package, click on the OK button.



This means that you can add source folders and packages to your project with Eclipse as well.



### **How do I add source folders to my .NET project using Visual Studio?**

To add a source folder to your .NET project, click on the Add a New Project option in the project wizard.

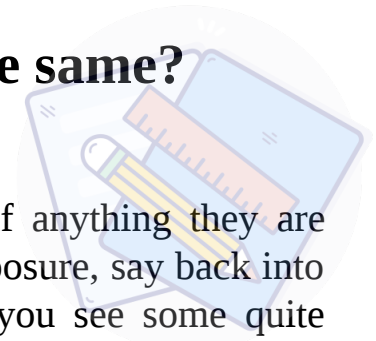
At first, it will ask you to check whether you want to use the project template that you have created previously.

It will show you the following step. Click on the “Add a source folder” button.

If you check the “Add a source folder source of type X” checkbox, it will create a new folder that contains the source folders of type X. Click on the OK button to add the second source folder.

Click on the OK button to add the second source package.

## Chapter 4 Do Java and Javascript the same?



In all likelihood, they will be very similar because if anything they are going to be a better place for you to get your initial exposure, say back into the center of the internet. But don't be surprised if you see some quite obvious differences.

On a very basic level, the distinction between Javascript and Java is that when Javascript runs on a computer it accesses a separate portion of the operating system called the network stack and will then run on top of it. The Java version will be run on the standard operating system, called the operating system. There is a difference in the semantics of what your computer does when Javascript runs than when it runs on Java.

The programming languages will be quite similar, but in the context of actually understanding a business and the specifics of how it's run the languages will differ in terms of their semantics. If you are running a business you will be doing more high-level tasks and the syntax of the language will be very different than if you are using the language to run as a hobbyist in the developer's world.

### What should I learn first?

In the case of Javascript, you are probably interested in programming in it because of the way that Google does its search. The basic functions in Javascript are very similar to those of Java in that you'll be able to create functions with arguments, get function results and return functions, all of which are Java-like constructs.

You should also consider that Javascript has these two other tricks that Java doesn't have, namely closures and anonymous functions. Closures are a way of defining a block that cannot be accessed in any way, but that you

can access in another function or a block on the stack, and also anonymous functions are another thing where the function will return a value, but it will not be in any way bound to the global scope and so that the next function can call the function, and so forth so that you can pass around variables in a very natural way. Those are the basic building blocks of Javascript.

You'll also be looking at functions of smaller and smaller expressions that in Java are called lambdas, and in the context of Javascript will be function functions, that will allow you to do things that are very different from the Java approach, where you can't pass the variable through a function as you can in Javascript.

But a lot of the time, if your only exposure to Javascript is from doing Google searches and from reading about the language on places like Stack Overflow, you're going to see a lot of small and very common JavaScript patterns that you're going to start learning quite quickly and then as your understanding and familiarity grow the real fundamental syntax and language features will be far more complex and won't be things that you will understand if you don't know the smaller and simpler parts of the language.

My experience is that at some point, no matter how much you study Javascript, no matter how much code you write, you're going to hit a point where you are going to have to read somebody else's code and that may take you a while. But for the most part, if you start to study Javascript now, I would recommend that you begin by learning some programming languages that aren't in Javascript, to help your foundation.

## **Java**

Java, on the other hand, is a language designed with the need of running very large and very diverse, and very complicated systems in mind.

The truth is that not everybody needs to be designing and implementing their bespoke systems in Java, that's certainly true of a lot of businesses, but for the individual developer that is not all that important. In fact, for the individual developer that is not all that important, but as a programmer, you are going to want to be learning Java anyway because Java is just the most powerful and most mature and most highly-regarded of the languages that most Java developers learn.

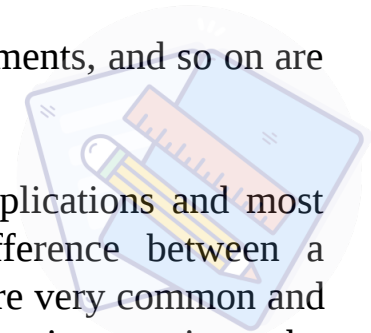
Java isn't a tool for creating websites. A lot of people, including Google, have given up on the notion of writing a website in Java, they're going to use an HTML engine or maybe they'll use PHP or something like that. But you should not write your website in Java. You should not design your website in Java, you should not write your apps in Java, you should use Java to allow you to write applications that are larger and more complex than the smallest personal website.

Java has a very comprehensive API. So all sorts of libraries and APIs that you may want to use are already included, such as the network stuff, including the Netbeans development environment. And the most important part of all of that is the fact that as soon as you download a copy of Java you can create a new Java virtual machine or a separate process that will allow you to run your Java code in the same browser window that it was originally written in if you want.

In the context of web development, you're going to want to do that. So you'll use the JDK which is the Java Development Kit to download your virtual machine. If you're familiar with Windows XP, you're going to have to create a second CD/DVD, and you'll install that CD on your computer with all of the tools that you'll need. That process is almost identical to what you'll need to do to install a Windows virtual machine, except for the fact that you'll want to add some drivers for your computer so that it'll recognize the virtual machine that you're installing.

Once you've made those two CDs or DVD's you're ready to go, and now you can write Java in the same environment that you would use to develop for Windows or Mac OS X or Linux. The syntax for things like strings,

numbers, variables, lambda functions, iteration, if statements, and so on are almost the same as they are on those other platforms.



Java is a language that is designed for production applications and most importantly, as it turns out, there's very little difference between a commercial app and a utility application. Utility apps are very common and very successful and the need to develop those applications is not going to be a huge hurdle for you once you get started. Utility applications are sort of designed to be disposable and that is okay, but those are not apps that are going to stay with you.

The most important thing about Java that everybody needs to know is that Java is a generally available and publicly released open-source software package, and that is a big change from what we've experienced with previous versions of Java, such as J2EE which was quite proprietary. Java is an important part of Sun Microsystems' strategic plan to get its open-source strategy up and running with its Virtual Machine.

It is important to understand that Sun Microsystems released Java under an open-source license, which means that anyone who uses the Java code can make changes to it and use it in any way that they see fit without paying Sun for that use.

You can even modify the code and release it as your version, or you can modify it to develop what is called a plugin for other programming languages, such as C++, and they call that a "Java Native Interface". So Java can be used by other languages. And if you're using another language, such as C, for your plugin, then the object model that you're using in Java comes directly from the object model that you're developing in C. The Java compiler is simply a front end for the object model.

There's a very nice presentation that Sun Microsystems put together that shows how that all works. In the YouTube video they've put together to promote Java, I've embedded it at the bottom of this article.

There's a huge amount of documentation on the web, but of course, if you want to learn all of that you'll have to read it yourself, so here are some sites that provide some information.

There's also an excellent book that was written by Jeff Atwood, a Web Designer, and Jeff Atwood created a complete, step by step, tutorial on how to set up a Java Development Environment.

### **Some tips for Java developers**

I've got a couple of different suggestions for Java developers if you're a Java developer. You might want to try a JRE on your laptop, instead of using the system Java, so that if your internet connection is slow or the machine is having problems, your Java app isn't going to be impacted because of that.

If you're a Java developer working with a team, or you're planning to join a team, one of the most important things you can do is understand all of the different programming languages that are supported by the JRE. There are five different Java-compliant programming languages that you can use for developing JRE apps, which I've listed for you at the bottom of this article. They are:

- C
- Haskell
- Groovy
- JavaScript
- Java Servlet and JSF

If you are working with a team, you'll want to put those all together in a project directory, and then you'll want to add the JRE you just downloaded to that project directory, and that project is going to be the default application that's going to be developed within that team.



You're going to see that each of those languages has been placed into its directory within the project.

Six functions are used to implement the build process for a JRE app. They're called jar files. The process of creating a Java jar file is called "compiling", which means that it's compiling a JAR file, which is a jar file that has a JAR extension.

But here's what you need to know: You can use any language that's included with the JRE.

The only requirement is that you've got to have a Java compiler installed on your computer, and to use the JRE you'll need to have a Java app that you're using within the JRE.

In this example, we've got a data source in the back end that's importing SQL tables. I'm not going to go into any great detail about the specific data source that I've been talking about, because there's a lot of great information out there about this.

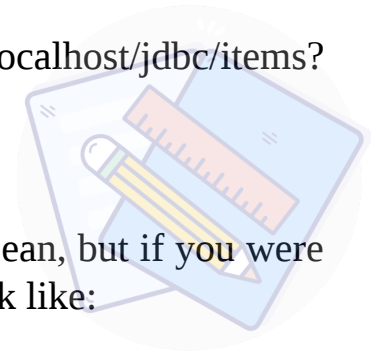
The diagram also shows a JavaBean, which is just a wrapper around a Java class, and it has this global "Bean" that's going to be the class name.

This particular JavaBean class has some methods, which are all clearly outlined here, but if I were to go through and explain what all those methods do, I'd probably break it up into three different pages, because they are three separate Java classes.

I'm going to use some really simple Java code here to demonstrate how this works. The method called "contains" within the JavaBean class is a simple member function within the class. This method contains one String in an array.

If you were to run the sample program, the following information would be printed out:

- [http://localhost/jdbc/items?  
table=items&stored\\_entity=com.JMCallback  
.com.bqn.my\\_name.co\\_item](http://localhost/jdbc/items?table=items&stored_entity=com.JMCallback.com.bqn.my_name.co_item)



That's just about all you need to know about that JavaBean, but if you were to take a look at this in IntelliJ, here's what it would look like:

In the above JRE project file, you can see the second part of the JavaBean class, which is called `java.jdbcCache`, and is a Java class that is stored in a static data folder.

When this data is imported into a JavaBean class, it's inserted into a separate static data folder, and then this class is known as the JRE Cache class.

Within this JRE Cache class, which is stored inside of the `/resources` folder, we have all of the methods that we've seen in the previous JavaBean class.

The first method within the JRE Cache class is called `"loadFromCache"`. This method has the following parameter that's going to be filled in with the name of the JavaBean class that you want to be able to load into a JRE Cache.

You can use the `"application:directory"` feature within IntelliJ to select from the application folder where you've been running this Java app, and you can choose the file `"application/com.jmcallback.jdbcCache"`. If you chose this file, then this will be the name of the JavaBean class that you will use to reference when you're creating the JRE Cache file within the JRE Cache project file.

The second method within the JRE Cache class, which is called `"setContentDirectory"`, and has the following parameter that's going to be filled in with the name of the directory where you want this JRE Cache file to be placed.

If you chose the “JRE App” folder, then you would just go ahead and navigate to the “application” directory within this directory.

If you chose the “java.jdbcCache” directory, then you would be able to navigate to the “JRE Cache” directory within the “java.jdbcCache” directory.

So in the above, the “application” directory has a class called “JRE”, and the “java.jdbcCache” directory has a class called “JRECache”. If you looked in the folder “application” within the “JRE” directory, you would have seen a class called “java.jdbc.JdbcCache.” So here, this particular class has two fields, “name” and “url.” When you call the “name” field, then you’re going to be able to fill in the name of the JavaBean that you want to load into this JRE Cache file.

The “url” field, here, will be something that we will modify later on in the article to put in a path to a website where we will be able to get the JRE Cache file, and this will be stored in the “application” directory of this particular JRE Cache file.

The third method within the JRE Cache class is called “getRequestCacheDir”. This method has the following parameter, and it’s going to return a directory that you can navigate to for accessing the JRE Cache file.

If you chose to select the “JRE App” folder, then you will be able to navigate to the “JRE App” folder, and you would have a directory named “application/com.jmcallback.jdbcCache”.

If you chose to select the “java.jdbcCache” directory, then you would be able to navigate to the “jdbcCache” directory, and you would have a directory named “jdbcCache”. If you chose to select the “java.jdbc.Cache” directory, then you would have a directory named “jdbcCache.” If you chose to select the “java.jdbc.HttpConfig” directory, then you would have a directory named “jdbcCache.”

So for this particular Java app that we're going to be working with, we want to try to get this JRE Cache file to be stored in the "jdbcCache" directory within the "jdbcCache" directory.

However, before we can do that, we are going to need to have some Java files located within the "application" directory that we're working with. So we are going to need to go ahead and click on the second tab in our terminal window, and then click on the third tab. This will be within the folder called "assets." Within this folder, we will be able to find two files that we're going to want to drag into the terminal window within our workspace directory.

## **What are frameworks for Java?**

If you are used to working with frameworks, and this is by definition the majority of Java developers, then please read this article about frameworks and what they are not. Frameworks are designed to help you use common functionality, which can be included in the codebase or extracted for development.

You might think that if your application is fully composed of frameworks (almost like a library in JavaScript), you should write your code and not worry about the framework at all. This is true if you follow the guidelines of microservices, which is exactly how a modern codebase should look like. You write the code and the framework just provides you with its static pieces.

But if you want to, for example, write an embedded server that you can run in your browser or just a web API to expose a static route for clients to load/list your RESTful API, then there is absolutely no reason to be thinking of your code as an implementation of some underlying framework. You can just write the application in pure Java, and the framework will do the extra bits of work for you.

It is this flexibility that makes frameworks a headache. You might imagine that the work a framework adds to the code is done for you. The truth is that most frameworks are designed in such a way that they require a certain way of thinking about the code you write. Frameworks change the way you think about how your application is constructed. You don't want to break your structure, because that can cause a mess.

A quick example: To use JSF for your Java web applications, you must use the annotations to specify how a Java object should be rendered.

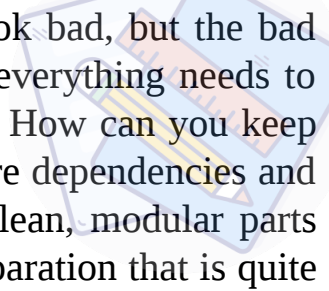
You need this annotation, and you should always use it. It will hide your code, and make it clear to a developer who is unfamiliar with JSF that this class should be called `MyPresenter` and should have its `primary()` method called `schedule()`.

## **Framework design considerations**

Now that we got the semantics out of the way, let's talk about the design choices that made it possible to write a framework in Java in the first place. I have listed a few of them here, and then I will try to help you understand what is in a framework.

## **Aesthetics and syntax**

Frameworks are bad because they look bad. They look like a throwback to a world where everything used XML files to store data. This may not seem too bad when you're writing a script that runs as a service, but it makes you start to hate it when you come to write something like a Spring MVC framework. You want to move towards the platform on which you want to build an application, but since you are writing a framework, your MVC code needs to obey all the syntactic norms of the framework that makes it look pretty.



I said earlier that frameworks are bad because they look bad, but the bad part is not the fact that it looks ugly. It's the fact that everything needs to look nice, which adds to the complexity of the project. How can you keep the modularity of the system when you start adding more dependencies and classes to it? You are forced to mix more and more clean, modular parts together to make up a whole, with a level of vertical separation that is quite annoying.

The same thing happens when you try to write a pure Java framework. Not only do you need to wrap your code in a peculiar mixture of Java and Scala to behave like a true framework, but you also need to provide the libraries necessary to get the framework to understand it. For example, `javax.faces` should be a core library for Spring MVC, but you cannot provide it as a dependency of your project.

It has the same reason as an XML configuration file: it needs to look and behave like it is designed to be part of a codebase. You don't want to hide it somewhere where someone can't find it, even if it would help you to increase its modularity. So, you end up making a huge class with an abstract set of API that only a developer who has a working knowledge of the framework can use.

Java, on the other hand, does not have this issue. A programmer who wants to use a pure Java library, that is designed to be part of a project, can just use it, without worrying about the framework. Spring MVC is a Java library, not a framework. It is a good place to start your Web development, because all the abstractions it provides are pure Java, without worrying about annotations and when to use them. The Scala team learned from this to develop Typesafe Scala, which, just like `javax.faces`, is a Java library.

## **Scala and MVC**

The biggest advantage of using Scala is that it provides the framework from the ground up for MVC. It's a purely functional language that provides primitives and procedural abstractions to give you the framework from the ground up, without additional dependencies. You can just focus on your application and do all the refactoring in your local IDE. There are some tricks you can do to make your code more modular by using advanced concepts like `val` and `valScala` (which we will discuss later). However, the pure Java implementation still provides all the power of the Framework-of-a-World, and it does it without a headache.

As for the rest of the MVC framework, Spring and MVC, it is just another Java library. They have many similarities. Spring MVC is based on the JPA framework, which has been recently rewritten in Scala and can provide a more declarative API than Java typesafe types. JPA enables a system to resolve columns and column sets of a database based on columns that are declared in the entity objects that are created in the controller classes.

In terms of the functional programming concepts that Scala offers, it can be seen as a generalization of the Scala language. There are no fundamental changes to the language, which provides libraries to implement imperative concepts, like setters, predicates, and the like. However, every layer of the stack that is added to the application, adds abstractions to the code and adds functional elements. Thus, a major way to extend your application with functional concepts is to use Scala code within an existing Java application.

Generally, a more functional language will handle more abstract concepts of the real world. Java, on the other hand, provides more of the imperative concepts, and more of the more special-purpose techniques used in object-oriented programming. You can use functions to iterate over arrays, you can use a callback, you can use multiple return values, etc. In functional programming, you don't care about any of these abstractions and instead focus on working with concrete concepts.

## **MVC and Spring**

Spring MVC is a small library that implements the MVC pattern for the Spring framework, which is a community-driven Java framework that allows you to extend the native API of Java applications. You don't need to get this library from somewhere else, it's built into Spring.

In essence, the MVC pattern is a set of methods that are used by a Spring application, so that you can bootstrap an application and pass the main parameters that you want to expose into the components of the MVC structure. Spring takes the template system that you can find in JSF and a couple of specific annotations that extend the default behavior of the system and generates an interface to the design, which you can implement using any convention you like.

For example, you can define the view as a template, and it's perfectly valid to pass it as an argument to the constructor or to use a specific piece of code as a view to define the view and get an instance of the view that can be manipulated. An entity could be a class that you can implement, but it can also be a lambda expression.

What you have to remember is that the MVC pattern is still a pattern, but it's one of the most used patterns in the Java programming language. So, whenever you see it in your code, it's probably part of a web framework, which is defined as a set of rules for the development of an MVC structure.

Notably, Spring MVC doesn't contain the MVC concept that I spoke about at the beginning of this article, which is separating models from views and controllers. The idea is still the same: the view is just a convenient API for creating a view, which implements the components in an MVC structure, whereas the controller is a class that inherits from the Spring MVC Controller class. The model is the implementation of the ViewModel interface, and the controller is an implementation of the Controller interface. The problem with Spring MVC is that it doesn't scale very well. It's an "it works for small projects" kind of pattern. It's an excellent choice if you are working on something really small, like a little site or something,



but if you want to deploy that web application into production, then you'll need to look at other patterns.

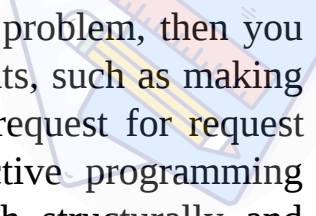
That is why I suggested in the beginning that if you work on a big web application, then use an OOP pattern such as Reactive programming. Reactive programming is an object-oriented paradigm that involves elements that model reactive behavior (and consequently, it is often combined with reactive programming) and that allow you to model the real-world world, where there are real-time systems with many asynchronous elements and a lot of data flow. Reactive programming is pretty much a good way to model real-time systems.

Real-time is an abstract concept, but Reactive Programming does help you to have the ability to handle reactive elements within an abstract set of concepts that you can model within an OOP environment. Real-time has been around for a long time, and Reactive programming can be used with any OOP environment. One example of it is with a database, so you can read data in real-time and modify that data in real-time while keeping the state as static as possible. Another example is that you can use REST and more generally, event-driven programming. You can model real-time as a stateless functional programming paradigm. I've seen many projects over the years that are started with a database and that use a variety of reactive techniques.

Most of the time, Reactive Programming helps you to cope with real-time systems and reactive elements in a declarative fashion, which allows you to model things in a declarative way and then to write code to handle those reactive elements and to deal with the time-critical elements.

If you think of a typical web application that is being developed today, that's pretty much what you would do with a Reactive programming web application. I will describe a simple Reactive programming web application as a case study that you can study in the next section.

## **Reactive programming in a web application - A real-world case**



Earlier I explained that if you have a reactive software problem, then you usually need to write code to handle the reactive elements, such as making an HTTP call, reading from a database, processing a request for request parameters, making an HTTP call, and so on. Reactive programming usually also involves a bunch of data structures, both structurally and dynamically. This whole set of data structures, what we would call a reactive database or a reactive cache, helps us to model the reactivity of the system.

I will use an example from the Linux operating system to explain how to build a reactive database. For me, the Linux kernel is the most interesting single component in the Linux operating system. Most of the systems in the Linux kernel are just simple examples of very complex systems that we build in some of our other projects. That is why the kernel is the classic example of a reactive system and in particular a Reactive system.

The Linux kernel is all about data flow, but in particular about performance. Let me just state it very simply: if we have a set of users connected to our network, then we need to handle the influx of user connections in a very efficient and responsive way, and one way to do it is by having a micro-kernel that lets the network stack manage the user connections, as well as the access to hardware such as network interfaces.

I love the idea of having a micro-kernel that lets the network stack manage user connections, but my favorite part about the kernel is the networking stack. Every time I turn on my computer, I connect to the Linux kernel in the hardware layer, and I am connected to all the services that the kernel provides.

For the user sessions, I connect to the main kernel thread (CPU) in a very specific way, because I also need to be able to share my device with other users and to share some of my CPU. The CPU access is managed by the kernel in the hardware layer. For example, if I have a graphic driver, then I

can use the graphics device directly from the CPU. If I don't use a graphic driver, then I also connect to the CPU through an IPC service.

However, for a session to connect to the networking stack and to read data from the network, I have to connect to the kernel. That means that I must bind to a specific CPU. I can do that either via socket programming or via a lock that I register with the kernel. You can see that the Kernel is a Reactive system that allows me to create the session directly from the kernel and also has this very interesting ability to read from the kernel. This is why I prefer the networking stack as the core of the Reactive system.

## **How to Build a Reactive system**

A Reactive system in a web application is similar to how a micro-kernel connects to the kernel in a Linux kernel. This means that for a client to connect to a Reactive system, we need to create a web application and connect to the database, as shown in the following diagram:

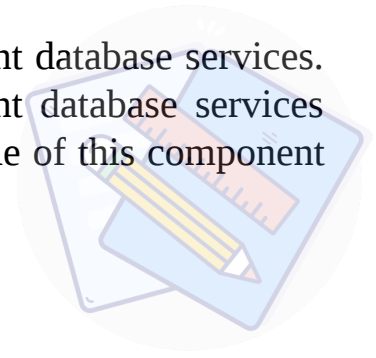
I want to show how we can combine reactive technologies. We can show how we can connect to databases and how we can handle queries with a reactive database using a few different techniques. We will cover a few techniques to connect to databases in particular here, but we will also show how we can build a reactive database in general.

In addition to the example application, we will also see the main framework for implementing reactive technologies in Scala, as well as the most popular reactive libraries.

## **Connecting to a reactive database in Scala**

As a starting point, let's first create a service that serves as a container for the database services. The main component that we will need for the app is

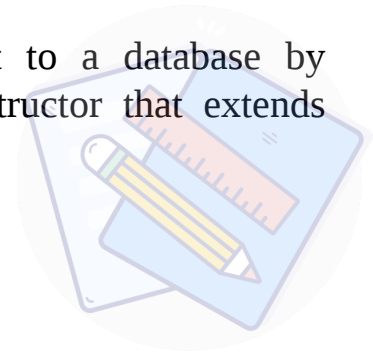
a basic dispatcher, which is an interface for the different database services. This component is responsible for calling the different database services when the dispatcher receives an action. Here is a sample of this component in a high-level form:



- ```
import org.scalatest.transducers.Console import
org.scalatest.transducers.Reactive import scala.concurrent.TimeUnit
import org.scalatest.tools.reactive.Network import
scalatest.implicitUsingRuntimeImplicits var dispatcher:
RuntimeImplicit def getPermissions(handler: Handler): Reactive = {
val options = Handler(handler)(options) return (handler)(options) }
val runningReactor = Server("testdb")(reactive dispatcher) val
operatingReactor = Server(Reactive.class) val queryReactor =
Reactive.class val sessionReactor = Reactive.class Reactive.start()
val databaseConnector =
DatabaseConnector(databaseConnector("testdb")) val
connectionConnector =
DatabaseConnector(connectionConnector("testdb")) val
connectionReactor =
ConnectionReactor(connectionReactor("testdb")) val
databaseConnector: DatabaseConnector =
connectionConnector("testdb") val databaseConnectorFailedReactor
= ConnectionReactor(connectionConnectorFailedReactor)(active
dispatcher) }
```

To be able to connect to a database, we first need to create a connection and register it with the database connector. The connectionConnector is a dispatcher of our data source and holds the connection details for the database. The database connector contains the database details, for example, the database name and the table names. To connect to the database, the dispatcher will create a reactor. In this example, the reactor will be a basic service that starts an instance of the database connector. The Reactive framework will automatically register the reactors, so we don't need to register them manually.

The Reactive framework makes it easy to connect to a database by extending Reactive.class. Here is the complete constructor that extends Reactive.class:



- ```
import org.scalatest.transducers.Console import
org.scalatest.transducers.Console.routes import
org.scalatest.tools.reactive.Network import
org.scalatest.implicit.UsingRuntimeImplicit val router =
Router(routes) var reactor: Reactive = new DefaultLoader
reactor.register( new DefaultExecutionListener { override fun
onTransactionSuccess(executor: ExecutionContext, result:
ExecutionResult) { transaction.finishExecution(result) } override fun
onTransactionFailure(executor: ExecutionContext, result:
ExecutionResult) { transaction.failure(result) } override fun onStop()
{ dispatcher.stop() } override fun onStart() { reactor.start() } override
fun onFinish() { reactor.stop() } val currentConnectionFailedReactor
=
connectionConnectorFailedReactor(connectionConnectorFailedReact
or) reactor.connect()
```

In the constructor, we register the initial state of the Reactive class and all the necessary endpoints that we can use to interact with the database. The Reactive class also contains a new DefaultExecutionListener that is responsible for handling the transaction process.

We call the execute() method that starts the transaction and handles the result. In case of success, the transaction is completed. In case of a failure, we catch the exception and call the stop() method, which signals the termination of the transaction.

To be able to handle the database connection details, we register a new default execution listener. When the connection is active, we call them on

the transaction success method to perform the transaction. In case of a failure, we catch the exception and call the stop() method, which signals the termination of the transaction.

After we handle the database connection details, we have the reference to the database connection. We can inject it into the reactor using the connect() method.

- ```
class DatabaseConnector extends Reactive.class { def connection:
Connection = new DefaultConnection("testdb") def reactor: Reactive
= new DefaultLoader reactor.register( new DefaultExecutionListener
{ override fun onTransactionSuccess(executor: ExecutionContext,
result: ExecutionResult) { transaction.finishExecution(result) }
override fun onTransactionFailure(executor: ExecutionContext,
result: ExecutionResult) { transaction.failure(result) } override fun
onStop() { dispatcher.stop() } override fun onStart() { reactor.start() }
override fun onFinish() { reactor.stop() } def onSuccess(transaction:
Transaction): ExecutionResult =
transaction.transactionCompleted(result) def onSuccess(transaction:
Transaction): ExecutionResult = transaction.transactionStarted(result)
} // The constructor needs to add the connection to the network so
that we can // register other reactor after it def
connectionConnector(db: DatabaseConnection) =
DatabaseConnector(db)
```

We define a new connection listener that registers a connection, which is used to connect to the database and attaches it to the Reactive class. The reactor class becomes the receiver of all the calls to the database.

At this point, our thread can run the queries and events. But it's still synchronous and can't do anything else. The next step is to remove the Synchronous dependency from our run method.

All you have to do is create a new JobBuilder with a timer that runs in the background and kill the main thread when the timer expires.

- `val job = JobBuilder() val reactor = reactor() reactor.run(job) 1 2 3 4 5 val job = JobBuilder ( ) val reactor = reactor ( ) reactor . run ( job )`

And we get rid of the need for an external thread. All the heavy lifting is done inside the reactor class.

Run is the default `ExecutionListener` that is bound to the `onStart()` method. In our run method, we call the `onUpgrade()` method that replaces the existing receiver with the new one.

While running the actual business logic, the reactor class simply calls our state transformer `onUpdate()`. This method is called when we perform the database transaction, and we can do whatever we want with the data.

## Chapter 5 What are tools for Java language?



Imagine that there are several language constructs and facilities available to you. Some of them, such as inlining or only-functional constructs, will give you better performance, but might not work for all situations. Some features improve Java's safety, readability, or developer productivity, but not necessarily in a way that they would be advantageous in every situation. Furthermore, several idioms make it difficult to do certain tasks.

These issues have led some developers to use Java's tools in many different ways. For some people, tooling is not very important, but for others, it is very important. As an example, some people use Eclipse, and other people use IntelliJ, and the question is whether Eclipse or IntelliJ is better in some respects. The same goes for the JVM itself. Some people use it in a strictly single-threaded fashion. Others prefer to use JVM-level optimizations or JITs. As with so many things, there is no "best" approach, but rather a spectrum of approaches that will result in different tools for the same problem. It's important that you understand the differences between tools for any given problem, and that you understand the tradeoffs involved, to make the best decision for your case.

In other words, find a suitable tool for the problem, suggest it to the developer, and also provide the required configuration to be used. If we break it down, tooling is something that "picks out" and shows you what to use, and "suggests" the appropriate tool for the problem. Tools that are too hard to use will simply not get the developer's attention (and thus, will not be used). On the other hand, tools that are too easy to use will likely not be used at all. If we break it down further, some tasks (e.g., printing ) are "hard" and require a lot of configuration. Other tasks (e.g., querying ) are "easy", and require no configuration. And other tasks (e.g., querying ) are "not a tool, but rather a method". And as you can see, there is considerable variation in this area.



Most developers that we talk to fall into the “too easy” camp, where a handful of tooling “just works” without any configuration (or at least, much less than is necessary). Some developers fall into the “not easy” camp, where they must configure things to be useful (e.g., printing, querying, and many other things). And some fall into the “not useful” camp, where the tooling is not used at all because the task is not well suited to any of the available tooling.

### **Does the tooling work well?**

The following problem statement will be used to assess tooling effectiveness:

“The purpose of the standard JVM is to allow us to run our Java code in a single-threaded, low-latency manner. Therefore, in general, we will want the JVM to operate in a single-threaded, low-latency manner. However, we have some projects which are probably better suited to multithreaded, high-latency operations, such as web applications that download images from the web. These projects would benefit from the use of threads, and the JVM will likely want to give us the ability to run them in a multi-threaded, high-latency manner.”

Of course, there are many sub-parts to the statement above, each with a different level of abstraction. In general, I tend to start with low-level abstractions and then work my way up, taking into account the concerns of real developers as I go. In other words, this is a very rough approximation of what real developers think and not a hard set of requirements.

### **Does the tool provide the necessary configuration?**

For a tool to be useful, it must provide enough configuration to be used. This is the one that probably seems to trip up most developers, as there are

many ways to configure the tooling.

One way is by making assumptions about how a developer uses the tool (e.g., “all developers are the same, so let’s configure the tool to run in the same way across all developers”). That is, if your goal is to provide a “configurability map” for your tool, you’re probably going to fail.

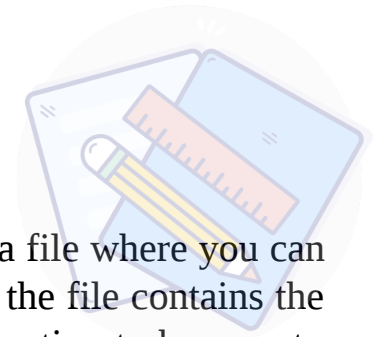
Another approach is to take the typical way that developers use the tool (e.g., userland, configuration via the command line, etc.), and then work backward to design a tool that can function the way users are using the tool today. That is, make sure that the tool runs on the JVM, and not on some other processor (which is not the case with Java Web Start).

This leads to another concept that I call “pattern matching”, as it involves matching the project/app to a specific configuration (which could be a particular JDK version or even a particular JVM configuration). This can be done using proxy config files that correspond to the project or application, and that is parsed to determine which configuration file(s) are appropriate. (This post gives a good introduction to pattern matching.)

Another approach is to ensure that the tool includes a configuration file for each supported configuration. This is, however, a somewhat tedious approach, and is only practical for many smaller projects that have only a single configuration. There are also several concerns about mixing the configuration files: what happens if you need to switch between the same JDK version, but you have two different configurations? The configuration files are also different, so you can’t reuse configuration files between projects.

So, the question is: how do you provide a configuration in the right format for a given configuration?

## **Is the configuration properly organized?**



Sometimes, you have to focus on the details. You need a file where you can store the configuration, and you need to make sure that the file contains the proper options. At the same time, you want the configuration to be easy to read.

## **Is the configuration customizable?**

Of course, if the configuration is not easily modified, the tooling is of limited use. What about the configuration? Do you want it to be editable? (You can't do that for Java Web Start.)

Finally, the configuration is a little different with Web Start. In a similar vein to the level of abstraction, there are many ways to configure the tool to work on the JVM. Web Start, however, provides some flexibility on how that configuration is created: there is a native configuration parser that it supports, and can modify the program to work on different JVM's (e.g., among the JVMs that Oracle supports).

As such, Web Start offers developers much greater flexibility in how the configuration is created.

## **The Future**

There are many ways to define the configuration schema for a tool. Some take a "configuration map" approach, whereas others use configuration file patterns. One thing is for certain, though: any tool that provides the ability to create a configuration in the tooling will need to provide the proper format for a particular configuration, and the proper format varies based on

the tools and configurations. (I have not investigated how to provide config support for just the one tool that allows you to create a single configuration, but it is certainly possible.)

As part of the ongoing development of PostgreSQL Console, we've created a prototype solution that provides support for configuration files for PostgreSQL, JRockit, and Percona XtraDB. (We're calling it the Percona XtraDB Console.)

Rather than having to develop a similar solution for PostgreSQL Console, for our next major release, we'll focus on extending the capability for user-written types to be used with the existing provider.

The first suggestion was to add the type parameter to the pathname attribute of the client created by the configuration. This type name would then have to be prefixed by the type name of the schema file so that if you wanted to create a new type that had a configuration that was different from that in the type, you would need to specify the file that contained the configuration type.

This, however, would require developers to be even more explicit about the types they wanted to use: they would need to specify a value for the configuration type, and they would need to use an explicit type name.

Another option is to extend the System.IO.Configuration interface (which was designed for .NET 4.6.) This interface provides a means to create new types, as long as they conform to the System.IO.Configuration.Types interface (since System.IO.Configuration.Types is not part of the System.IO. Type System, it would need to be built-in). As such, we could provide a module in the API that would accept a TypeBuilder, which provides a declaration language for creating new types. The type builder would then return a TypeBuilderValue type, which is a simple wrapper for a type that could be used to access its configuration property.

A future goal is to add the configuration type to the Provider's namespace so that it becomes something that could be built into the tooling.

Another option is to try a similar solution for config files, that would allow developers to create a file in the tool's "config" directory, and import it into the tool using a single line, rather than requiring multiple lines as the existing configuration.

As such, we are investigating the possibility of providing something like the following:

- `# create a new type in files "settings" . "{"ConfigFileType" . "  
{"configFilePath" . "}" # import this type in the type builder defined  
in the tooling object provider "TypeBuilder" . new "ConfigFileType"  
. configFilePath "{"configFileType" . "{"configFilePath" . "}"`

But, as I mentioned at the outset, this raises a concern about what to name these different types. This idea is consistent with the idea of providing a common set of classes for types that come in the tooling so that it can offer the ability to create different kinds of types that adhere to the same API. For example, there could be one class that conforms to the Base class, one that conforms to the ConfigAttribute class, one that conforms to the ConfigAttributeConstraint class, etc.

Of course, we can't use this idea for the TypeBuilder interface that is part of the System.IO.Type System. As such, we have a choice: either we can expand the System.IO.Type System classes (in a similar fashion that it was expanded with the change from Connection to ConnectionBuilder ) to support more types, or we can add a specific design that would require developers to explicitly specify what kinds of the type they are using.

## **Structural improvements**

The other big topic that came up during the discussions was the ability to make structural changes to the TypeBuilder interface. Currently, these changes are only supported on the WFT and TypeProvider interfaces, but

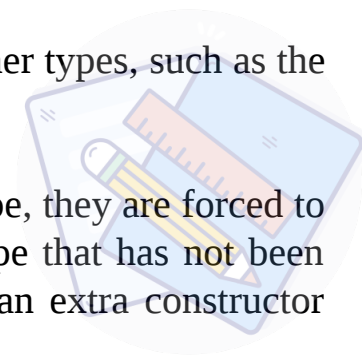
we would like to expand this support to many of the other types, such as the WCF-type providers.

Right now, if the user chooses to replace the built-in type, they are forced to remove it from the TypeBuilder interface. To use a type that has not been defined on the interface, they would need to provide an extra constructor argument.

This is a nice feature when you're working with the WCF providers, but it becomes more complicated when you have a type that already exists in the tooling. For example, we may have a type called CustomType, which is implemented in the TypeBuilder interface, but the type is already defined on the WCF providers. This can be considered a separate "type" and should be treated as such. If the user were to change the TypeBuilder interface, they would need to modify the types that they are using, since their constructor argument would need to be different.

So what are some possible solutions for this? One of the ideas is to change the TypeBuilder interface so that instead of providing a constructor for the types that do not currently exist on the interface, you provide a constructor for the type in question, which has a method signature that allows for different types.

Another idea is to implement the constructor for the type so that it provides the constructors for all of the interfaces that it can be used with, and then the user should be able to do the right thing.



# Chapter 6 How to learn Java programming language?



For those who want to learn the Java programming language, here are a few step-by-step guidelines that should help you:

Don't get confused by the Java programming language syntax. There are some major languages like C++, C#, and some minor languages. The syntax of Java is like all the other syntaxes.

If you have zero experience with Java programming, try getting an intro course. If you already know Java, then this tutorial can help you.

**If you are ready to get started with learning Java programming language, follow these steps:**

Step 1: Setup a java environment

First of all, you need to have a java environment. You can either use Windows or Mac OSX. I strongly recommend Mac OSX for this tutorial because of the cross-platform features. If you are using windows then you can try the VMWare Fusion, which allows you to run different operating systems simultaneously.

Step 2: Download the official Oracle Java 8

The official Oracle Java 8 is recommended for you to use for learning the Java programming language. You can also use Oracle Java 8-and-1 edition.

Step 3: Download Scala 2.11 or Joda Time

The official Scala and Joda Time are also recommended.

Step 4: Download and install the Gradle Builder

You will need the Gradle Builder to work with Scala and Java 9.

Step 5: Install the JAVA 9 (or Java 8) JDK and JDK 9 (or Java 8) JDK

If you have installed Oracle JDK, then you have to install the JDK 9 or Java 8 JDK and the JDK 9. Don't forget that JDK is not an acronym. It stands for Java Development Kit. If you want to learn Java programming language, then don't use the old JDK because the Java version now stands at 9.

Step 6: Download and install the Kotlin Project

Kotlin is a language developed by JetBrains, a popular programming language company. You can learn Kotlin by following the instructions here.

Step 7: Run the Kotlin Developer Studio

After you have downloaded the JAVA 9 JDK and the Kotlin JDK, then you need to use the Kotlin Developer Studio. To run the Kotlin Developer Studio, you need to install the Kotlin IDE.

Step 8: Switch to the Kotlin Developer Studio and create a project

To build Kotlin language and have it run in a Java program, you need to use the Kotlin IDE.

Step 9: Create a Kotlin project by entering the same file name as the Java project

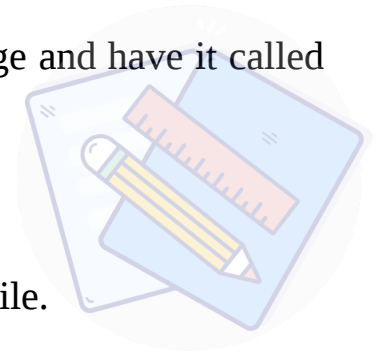
The next step is to create a Kotlin project in Java. In the next step, you need to use the same file name as the Java project.

Step 10: Create a Project file by entering the same name as the Java project





Now, you need to create a project file in Kotlin language and have it called with the same name as the Java project.



Step 11: Add a class to your Kotlin file

In the next step, you need to add a class to your Kotlin file.

Step 12: Add the Property Field

After adding the property field, then you need to compile and run the Java program by using the Java Compiler.

Step 13: Adding a lambda expression

You have to add a lambda expression in the next step.

Step 14: Extract one of the lines to produce an object

The next step is to extract a single line of the Java program to create an object. In this step, you need to extract one of the lines of the Java program.

Step 15: Extract a Parameter from Java code

After extracting a parameter from the Java program, then you have to run the Java program with the value of this parameter.

Step 16: Create a lambda expression from a Java class

In the next step, you have to create a lambda expression from a Java class.

Step 17: Extract the Parameter

After the lambda expression, then you need to extract the parameter.

Step 18: Extract the Private field

After extracting the private parameter, then you have to compile and run the Java program with this parameter.

Step 19: Extract a method from Java code

You have to extract a method from the Java code.

Step 20: Extract the Arguments field

After extracting the parameters, then you have to compile and run the Java program with this parameter.

Step 21: Extract the Value from Java code

After extracting the value from the Java code, then you have to compile and run the Java program with this value.

Step 22: Extract a Static Field

After extracting a static field, then you have to compile and run the Java program with this field.

Step 23: Extract a Type

After extracting a type, then you have to compile and run the Java program with this type.

Step 24: Extract another method from Java code

After extracting a static method from the Java code, then you have to compile and run the Java program with this method.

Step 25: Extract another parameter from Java code

After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.



Step 26: Extract a local variable from Java code

After extracting a local variable from the Java code, then you have to compile and run the Java program with this variable.



Step 27: Extract a String from Java code

After extracting a String from the Java code, then you have to compile and run the Java program with this string.

Step 28: Extract a Boolean from Java code

After extracting a Boolean from the Java code, then you have to compile and run the Java program with this Boolean.

Step 29: Extract a Number from Java code

After extracting a Number from the Java code, then you have to compile and run the Java program with this number.

Step 30: Extract an Object from Java code

After extracting an Object from the Java code, then you have to compile and run the Java program with this Object.

Step 31: Extract a Parameter from Java code

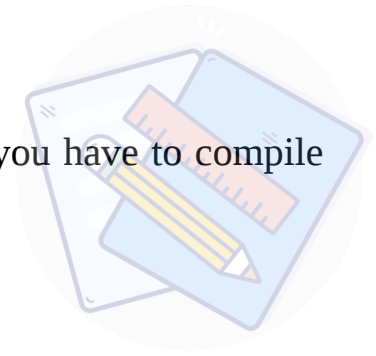
After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 32: Extract the Parameter Parameter from Java code

After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 33: Extract a Parameter Parameter from Java code

After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.



Step 34: Extract a Method from Java code

After extracting a method from the Java code, then you have to compile and run the Java program with this method.

Step 35: Extract another method from Java code

After extracting another method from the Java code, then you have to compile and run the Java program with this method.

Step 36: Extract an Object from Java code

After extracting an Object from the Java code, then you have to compile and run the Java program with this object.

Step 37: Extract an Object from Java code

After extracting an Object from the Java code, then you have to compile and run the Java program with this object.

Step 38: Extract another Parameter from Java code

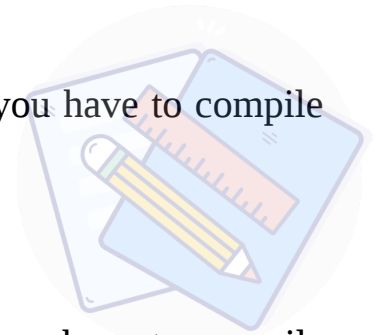
After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 39: Extract another Method from Java code

After extracting another method from the Java code, then you have to compile and run the Java program with this method.

Step 40: Extract another Parameter from Java code

After extracting a parameter from the Java code, then you have to compile and run the Java program with this parameter.



Step 41: Extract a Resource from Java code

After extracting a Resource from the Java code, then you have to compile and run the Java program with this resource.

Step 42: Extract another Resource from Java code

After extracting another Resource from the Java code, then you have to compile and run the Java program with this resource.

Step 43: Extract another parameter from Java code

After extracting another parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 44: Extract a Parameter Parameter from Java code

After extracting a Parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 45: Extract another Method from Java code

After extracting another Method from the Java code, then you have to compile and run the Java program with this method.

Step 46: Extract another Parameter from Java code

After extracting another Parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 47: Extract another Method from Java code

After extracting another Method from the Java code, then you have to compile and run the Java program with this method.



Step 48: Extract a Parameter from Java code

After extracting a Parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 49: Extract another Parameter from Java code

After extracting another Parameter from the Java code, then you have to compile and run the Java program with this parameter.

Step 50: Extract another Method from Java code

After extracting another Method from the Java code, then you have to compile and run the Java program with this method.

## **How to learn Java programming?**

My biggest problem as a Java developer is that I lack confidence in programming in the programming language. It took me several years to get a knack for programming in Java. Don't ask me about my C# skills. I have no idea. I am lucky to have a wife that is a professional IT recruiter for some of the biggest and the best organizations in the country. She is also an experienced Java developer and supports me and teaches me. The fact that I am married to her to this day has done wonders for my Java skills. I have the highest regard for fellow Java developers who've had to do a lot of Java to Java conversions on their own. The Java language, especially if it is a Java 2, has a lot of manual bookkeeping. For some reason, I keep on reading books and learning things about memory safety and other aspects of Java, which I had to do. I think the reason is my impatience.

In the book "Risking My Career by Writing My Own Java Module" by Steve Freeman, he tells a story about hiring a new Java developer who was

new to programming. At his first training session, the developer asked what the difference was between a Constant and a ConstantField in Java.

Freeman explained what a constant was and how he constructs his class in Java. The developer said he'd never seen anything like that before and wondered how it was possible to create a constant field in Java. "The hard part is to design your program to be correct in every detail, but especially to design your program so that things don't crash when they shouldn't and to do so in such a way that it's impossible to guess how the program will behave in a given situation," Freeman said.

This reminded me of my initial Java programming experience. How much more incredible is that story than the trivial anecdote about the Java developer who never heard of and couldn't construct a constant field in the first place?

Here are a few questions you should consider when learning Java programming:

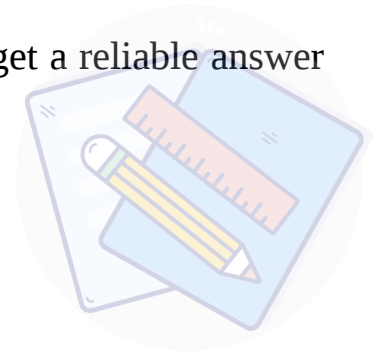
### **What is a constant?**

A constant is an integer or floating-point value. It does not have a single possible value. If you have a floating-point number, you should expect that it will have a single possible value. For example, in Java, a floating-point number is represented as a 32-bit integer, and the two possible values are 2.71828182845e-18 and 2.71828182845e-15. An integer is represented as a 32-bit floating-point number, and the two possible values are 0.5 and 1.0.

When you start learning Java programming, remember the hard fact that a number is always a number.

You should expect to write at least one line of code that will get you a definite answer in terms of the number that you got from reading a page of a book. For example, if you want to know what the largest prime number in

the range of 2 to the number you just read is, you can get a reliable answer from a book.



### **What's the largest prime number?**

It's 253.

In other words, the largest prime number is 253.

You should also know that you can use an Integer field with many different non-constant default values. They are present in every class and should not be used.

You should also know that there are many different ways of designing your object. If you change your mind later, you can change the default value of a field. For example, you can change the default value of a field in Int from 0.5 to 1, or you can change it from 0.5 to 2.71828182845e-15.

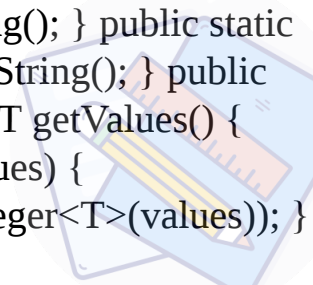
### **What is a constant(T) in Java?**

A constant is a variable used as a constant, and the variable has the same name as the constant.

Example:

- ```
class ParameterizedInteger<T> { static Integer constant1 = T; static Integer constant2 = T; static Integer constant3 = T; static int final constant = 0; public static int defaultValue = constant; public static T value = constant; private int maxValue; private int minValue; private int stepSize; public static void setDefaultValue(T value) { if(value <= 0) { value = value.toString(); } else if(value > maxValue) { value = value.toString(); } else if(value < minValue) { value = value.toString(); } else { value = 0; } } public static void
```





```
setMaxValue(T value) { maxValue = value.toString(); } public static  
void setMinValue(T value) { minValue = value.toString(); } public  
static int getValue() { return value; } public static T getValues() {  
return values; } public static void setValues(T values) {  
values.Clear(); values.Add(new ParameterizedInteger<T>(values)); }  
public static T getValues() { return values; } }
```

Note: If you have a question about Java, remember the saying, "Always use tools that can show you the same thing twice." Use a variety of tools in your programming work, such as a sheet calculator, a DDE view, and a compiler.

### **How do you define the type of a parameter?**

You can define a parameter of a class by declaring a default value or by declaring the type of the parameter. For example, the default parameter in the Integer class is also Integer. In this case, a default parameter is a parameter with the type Integer. You can also define the type as being ConstInt or Comparable.

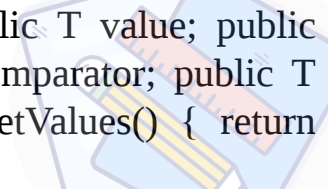
Create a new package and add the following code:

- ```
public class ParameterizedInteger<T> { public T value; public  
ConstInt value; public Comparable<T> valueComparator; public T  
getValue() { return value; } public static T getValues() { return  
values; } }
```

Now, create another new class, named Integer and in it, add the following code:

- ```
public class Integer implements ParameterizedInteger<T> { public  
Integer value; public T getValue() { return value; } public static T  
getValues() { return values; } }
```

You can change the parameter declaration in the class like this:

- 
- ```
public class ParameterizedInteger<T> { public T value; public  
  ConstInt value; public Comparable<T> valueComparator; public T  
  getValue() { return value; } public static T getValues() { return  
  values; } }
```

It's important to know that when you create the parameter declaration in the parameterized class, you should have different versions of the parameter.

In the first version, you should have T for the type of the parameter.

In the second version, you should have ConstInt for the type of the parameter.

Here is an example:

- ```
public class ExampleParameterizedInteger<T> { public T value;  
  public ConstInt valueComparator; public T getValue() { return  
  valueComparator.compare(value.getValue()); } public static T  
  getValues() { return values; } }
```

### **How do you return the value of a parameter in a method?**

You can return the parameter value like this:

- ```
public class ParameterizedInteger<T> { public T value; public  
  ConstInt valueComparator; public T getValue() { return  
  valueComparator.compare(value.getValue()); } public static T  
  getValues() { return values; } }
```

Here is the implementation of the parameterized class:

- `public class ParameterizedInteger<T> { public T value; public ConstInt valueComparator; public T getValue() { return valueComparator.compare(value.getValue()); } public static T getValues() { return values; } }`

The parameterized class definition with the parameter definition with the equals sign is part of the parameterized classes in the parameterization package. You can also add other parameter types to the parameterized classes.

### **What are the methods of the parameterized classes?**

In a parameterized class, you have to declare one or more methods. In this class, there are two method declarations:

- `public void compare(T value, T comparator) { if (value == comparator) { return; } } public void returnValue(T value) { return value; }`

The `compare()` method compares the parameter value with the value of the parameter in the comparator.

The `returnValue()` method returns the value of the parameter. You can call the `returnValue()` method with the parameter reference as the first parameter.

### **Do you have to create a class and implement methods with the parameters of the parameterized classes?**

No. There is a class named `ParameterizedMethod` and in it, there is one class named `ParameterizedMethodBuilder` that you can use for

parameterizing your method definitions.



## **How to add a ParameterizedMethodBuilder to a parameterized class?**

The `ParameterizedMethodBuilder` class is implemented as a `ServiceProvider` and you can register the `ParameterizedMethodBuilder` into a parameterized class. Here is an example:

- ```
public class ExampleParameterizedMethodBuilder { public
ParameterizedMethodBuilder(ServiceProviderParameterizedMethod
BuilderBuilderProvider) { this.ParameterizedMethodBuilderProvider
= ParameterizedMethodBuilderProvider.create(); } public
ParameterizedMethodBuilderBuilder create() {
this.ParameterizedMethodBuilderProvider.register(new
ParameterizedMethodBuilder(NewParameterizedParameterizerProvid
er.class)); return this; } }
```

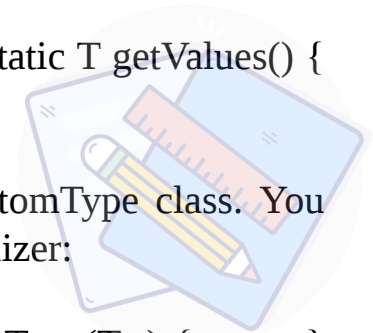
The `ParameterizedMethodBuilder` is an instance of the `ServiceProvider`. In a parameterized class, there is no instance of the `ServiceProvider`, so `ParameterizedMethodBuilder` uses the service provider for the parameterized class.

## **Do you have to create a custom implementation of a custom class in order to call the methods of the parameterized classes?**

No. You can call the methods of the parameterized classes like this:

- ```
public class ParameterizedType<T> { public T value; public bool
equals; public T getValue() { return value.getValue(); } public void
```

```
setValue(T value) { this.value = value; } public static T getValues() {  
return values; } }
```



The `ParameterizedType<T>` is an instance of the `CustomType` class. You can also create a new parameterized type with the initializer:

- ```
public class CustomType<T> { public CustomType(T t) { t = t; }  
public CustomType(T value) { t = value; } public void setValue(T  
value) { this.value = value; } public void setNumber(int number) {  
this.number = number; } public void setNoEqual(bool isEqual) {  
this.noEqual = isEqual; } public int getNoEqual() { return noEqual; }  
}
```

You can create another parameterized type as an interface and call the methods with the parameter as an argument.

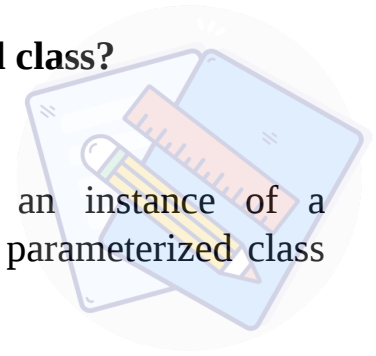
This is the same as this code:

- ```
public interface CustomType<T> { public T value; } public class  
CustomType<T> implements CustomType<T> { public T value;  
public CustomType(T value) { this.value = value; } public  
CustomType(T value) { this.value = value; } public void setValue(T  
value) { this.value = value; } public void setNoEqual(bool isEqual) {  
this.noEqual = isEqual; } public int getNoEqual() { return noEqual; }  
}
```

Here is a code example for creating an instance of the `CustomType<T>` interface:

- ```
public class CustomType<T> implements CustomType<T> { public  
T value; public T value; public CustomType(T value) { this.value =  
value; } public CustomType(T value) { this.value = value; } public  
CustomType(T value) { this.value = value; } public int getNoEqual()  
{ return noEqual; } }
```

## How to use a parameterized class as a parameterized class?



There is a `ParameterizedType<T>` that represents an instance of a parameterized class, so you can call the methods of a parameterized class like this:

- `ParameterizedType<T> myComponent; void doSomethingMethod(InputInputMyComponentInput, MyComponent myComponent) { myComponent.value.setValue(1); myComponent.value.setNumber(9); }`

## Do parameterized classes have to be subclassed?

No. The parameterized classes just use the static methods of the parameterized classes.

## Can I call the methods of the parameterized classes like this:

- `public class ParameterizedType<T> { public T value; public bool equals; public T getValue() { return value.getValue(); } public void setValue(T value) { this.value = value; } public void setEqual(bool isEqual) { this.noEqual = isEqual; } public int getNoEqual() { return noEqual; } public int getNoEqual() { return noEqual; } }`

I want to add my own custom parameters to the `CustomType` class and have them available through parameterized methods. For instance, I can put a

number and a string as parameters and call these parameters by passing a parameter as a tuple:

- ```
public class CustomType<T> implements CustomType<T> { public
CustomType(T value) { this.value = value; } public CustomType(T
value, string name) { this.value = value, name; } public
CustomType(T value, int number) { this.value = number, name; }
public CustomType(T value, bool equals) { this.noEqual = equals; }
public CustomType(T value, int noEqual) { this.noEqual = noEqual;
} public T getValue() { return value.getValue(); } public
CustomType(T value) { this.value = value; } public CustomType(T
value, int noNumber) { this.noNumber = noNumber; } public int
getNoNumber() { return noNumber; } public int getNoNumber() {
return noNumber; } }
```

### **What is the difference between a weak parameterized class and a strong one?**

The difference is that the parameterized class is “inherited” from the base parameterized class, but not “entitlement”. If you use a weak parameterized class, the parameter can be changed by its child classes.

For instance, in the CustomType class, you could have changed the value of the parameter so that it has no effect. If the parameter parameterized class is a WeakParameterizedType<T>, it would still be inherit from the base class, so you would still have to call the method as in the following example:

- ```
public class CustomType<T> { public WeakParameterizedType<T>
(T value) { this.value = value; } public WeakParameterizedType<T>
(T value, string name) { this.value = name; } public
WeakParameterizedType<T>(T value, int number) { this.value =
number; } public WeakParameterizedType<T>(T value, bool equals)
```

```
{ this.noEqual = equals; } public WeakParameterizedType<T>(T
value, int noNumber) { this.noNumber = noNumber; } public T
getValue() { return value.getValue(); } public
WeakParameterizedType<T>(T value, string name) { this.value =
name; } public WeakParameterizedType<T>(T value, int noNumber)
{ this.noNumber = noNumber; } public void setValue(T value) {
this.value = value; } public void setNoEqual(bool isEqual) {
this.noEqual = isEqual; } public int getNoEqual() { return noEqual; }
public int getNoEqual() { return noEqual; } }
```

You could change the parameter so that the value is changed to null , and you'd still be calling the same methods that you'd be calling with a strong parameterized class:

- ```
public class CustomType<T> { public CustomType(T value) {
this.value = value; } public CustomType(T value, string name) {
this.value = name; } public CustomType(T value, int number) {
this.value = number; } public CustomType(T value, bool equals) {
this.noEqual = equals; } public CustomType(T value, int noNumber)
{ this.noNumber = noNumber; } public T getValue() { return
value.getValue(); } public void setValue(T value) { this.value =
value; } public void setNoEqual(bool isEqual) { this.noEqual =
isEqual; } public int getNoEqual() { return noEqual; } public int
getNoEqual() { return noEqual; } }
```



## Chapter 7 How to use Java programming libraries?



Before we get started, it's important to note that Java programming libraries (sometimes called Java libraries) are just a part of the Java language. We should understand them and how they can help us, but in no way should we think of them as an alternative to writing our code. With that out of the way, here are the most important things you need to know about writing Java programs using libraries and with libraries.

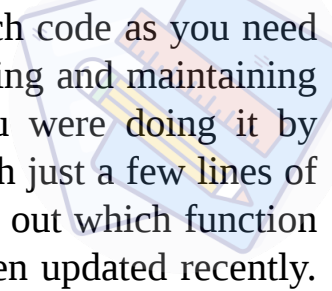
### **The Best of both worlds: Libraries for Java programming.**

When you're developing programs, Java libraries offer you an excellent option for solving a wide range of problems. It's so convenient! It's a relief, especially if you have to solve a lot of similar problems over and over again. They simplify the job of writing programs. And it's all because they take care of most of the details for you: you just need to write your program and they take care of the rest.

They can save you from writing a lot of tedious tasks, like security checks, for example, or code that requires tedious type-checks to perform correctly.

### **Some of the many uses for Java libraries.**

However, before you start using a Java library, make sure you're familiar with it. There are many Java libraries. Some, such as those that handle SQL, XML, and JSON strings, are quite specific to a certain programming language. Others, such as those provided by the Java compiler itself, tend to come with a large codebase and provide a lot of utility for the average Java programmer.



The beauty of libraries is that they let you write as much code as you need for your project. You don't have to worry about managing and maintaining your code in the same way you'd have to do if you were doing it by yourself. You can add new functionality to a library with just a few lines of code. The last thing you want to worry about is finding out which function you're using is new to the library, or whether it has been updated recently. Libraries take care of that for you!

In addition, you can easily use a library without having to write anything from scratch. With a good library, you don't need to write a single line of code to make it work. This is also a very useful feature.

### **Not so easy to use: Libraries require some effort**

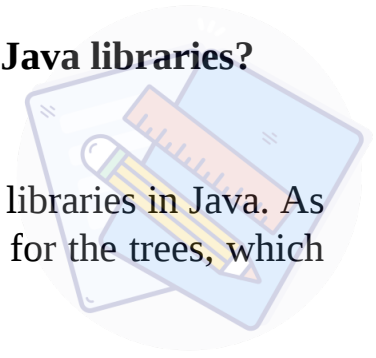
The other characteristic of libraries is that they are not easily accessible. They often require you to create and register a full-fledged project in order to use it. Sometimes they have different registration schemes. But then, it might require you to write different, customized code for every single library you use. The tasks needed to use some libraries, especially complex libraries, might seem daunting.

Another difficult aspect of using libraries is that they can have different interfaces. You have to study how they work, how to use them, and what you can do with them, in order to fully grasp the utility they provide and the final implementation of your program.

Finally, some libraries contain substantial code. If you use them, you'll have to spend time understanding all their pieces in order to use them. Even worse, sometimes there are incompatibilities between libraries. This usually means that you can't reuse or integrate libraries without making changes. This can be even more annoying because you have to fix the resulting program, which you may have already created, in a different way.

## What are some of the problems you face when using Java libraries?

I'll discuss the problems you can encounter when using libraries in Java. As you will see, sometimes it can be hard to see the forest for the trees, which is something that libraries are especially good at.



Let's start with some general problems, then we'll get into a few libraries' specific problems.

### Scalability

Scalability is one of the big problems of using libraries in Java. With lots of developers writing lots of programs in Java, sometimes the systems get overwhelmed. As a result, a lot of your code is being run by the same server.

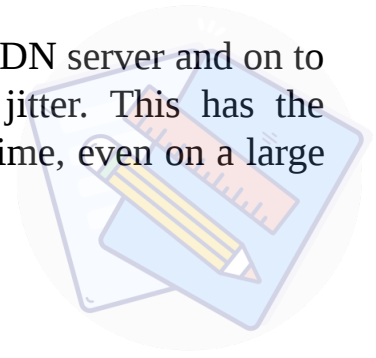
In particular, a very large number of users using a single site (or worse, trying to use the same code to deal with a large number of users) can lead to it "quaking" under the weight of too many concurrent users. This is called jitter.

It's easy to find out how jitter works and prevent it in your applications. One way to prevent jitter is to keep the load on the server low enough that you're not crashing and have to restart the whole application for jitter to go away.

Another way is to use a CDN and keep your application outside the application's main server. In this case, the load is distributed across several servers. But, keep in mind that when applications run on the same server they tend to get more and more similar in terms of functionality.

One common solution to the jitter problem is to use a service like Level3 CDN. You give Level3 the URL of your application and the CDN takes

care of moving your code from the main server to the CDN server and on to your users, so you don't have to worry about any jitter. This has the potential to decrease your Java application's response time, even on a large scale.



## **Instance-based Data Structures**

There are two types of data structures in Java: Object and array-based.

Array-based data structures allow you to store a group of objects in a single list, but they don't let you assign or change any of the objects in the list. If you wanted to, you could replicate a particular item in the list by doing something like:

- `int [] anItem = new int [list.length -1] ;`

Object-based data structures provide similar functionality, but they also provide a way to modify an item at a given point in the list.

If you were to create an array of objects, you could have an item in the array with the same name as an item that is already in the array. If you then modified the original item, you would have to change the reference in your array.

The problem with this is that once you modify an item, then the entire array is changed. Therefore, object-based data structures are less scalable than array-based data structures.

In addition, the amount of space required to store a large object is usually a limiting factor with many large data structures. If you want to store a ton of large objects, you can use a big data structure such as a graph, hash table, or matrix.

**The best comparison that I can make is the following:**



### Array-based

Array-based objects can be very large in terms of RAM and disk space and have a lot of efficiency problems. One of the ways to solve these problems is to use a cache, which should keep all the data in memory. You can put the cache in a variable and then assign an object from the cache when you need the data.

### Struct-based

Struct-based data structures allow you to define a structure and make some of its member variables private. Then, whenever you need a member variable you make it available via its interface so that you only have to apply it to the data structure and it doesn't have to be passed around to all the members. It should be noted that this is sometimes a better fit for the structure and should not be used if you want to keep the structure private and allow each member to be updated independently of the whole structure.

In other words, you can have a Struct as follows:

- `struct Patient { String name ; Integer age ; String address ; } ;`

In this case, you don't have to worry about the members being passed from one class to the next, but they are still available via their interfaces.

Another advantage of struct-based structures is that they allow you to add an optional member to the data structure, which the object exposes via its interface. Therefore, you can have more than one component of a structured data structure, which means that they can be smaller than arrays.

Array-based data structures offer very good performance, especially when you have very large objects that don't need to be updated often, such as big

databases or maps.

Java has a very interesting data structure called the hash table, which is used for large-scale object-oriented databases. But, in Java, you need to implement it as an object-oriented data structure, which means that you need to make sure that each property of a data structure is separate from every other property. Also, you can't make changes to a hash table class if a value that belongs to that property is also changed, which makes it very expensive.

In short, there are pros and cons for every data structure, but I like arrays better than hash tables and they are faster.

### **Create new trees as needed**

Before we start designing our data structure, we should ask ourselves:

Which of the trees we have created so far should we keep around?

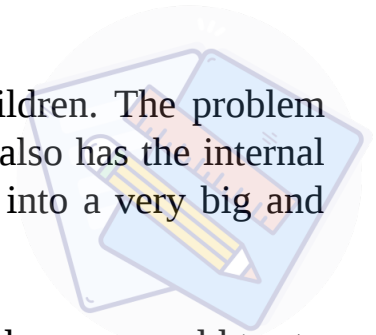
The first type of trees that I want to talk about are “empty” trees. An empty tree does not have any data in it, but that can be made into a tree by adding some objects.

So, here we have a tree with no objects.

This tree can also be made into a tree by adding some objects. But, what if we wanted to make a tree that had only one object in it? We would have to make a new tree with one object:

And we would have to make this tree in a slightly different way because we want to give each child some internal structure. When the tree is added to the DOM, each child will have to point to its parent. We also need to make sure that the DOM node that we put inside the tree is also pointing to its parent. So, it's not an empty tree. But it still is quite a small one.

Now, let's try to make a tree that has exactly two children. The problem here is that we need to ensure that one of the children also has the internal structure because if we don't, the tree might be made into a very big and unreadable mess:



Now, the Tree class that we have created is quite general, so we could try to make other types of empty trees as well. For example, it would be possible to make an “empty” tree with only two children:

But what if we wanted to make an “empty” tree with three children? It's possible, but we would have to make two empty trees with the same internal structure:

**But how could we create a tree that has four children?**

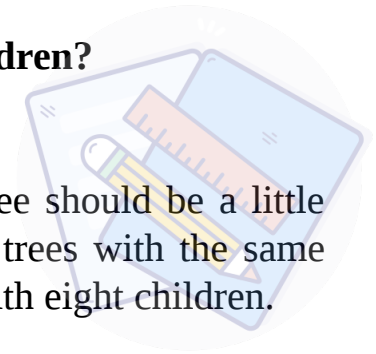
Well, the Tree class doesn't handle objects that have a size smaller than one. So, it would be possible to add two children to an empty tree like the one below:

**But how could we create a tree with eight children?**

The Tree class does not handle objects that have a size smaller than one, which is what makes our previous tree look like it has eight children.

To add the last child, we would have to make a new empty tree, with eight internal nodes:

**Now, what if we wanted to create a tree with ten children?**



Okay, so what we are doing is saying that our new tree should be a little bigger and that we should make it up with two other trees with the same internal structure. The first one would be a triple-tree with eight children.

The second one would be a quadruple tree with four children.

Now, we would have to make one more tree with the same internal structure and size, and call it the triple-tree.

This would create a triple-tree with ten internal nodes.

So, we have created three new trees, each with the internal structure that we wanted.

**But, our new triple-tree only has five nodes in it. But what if we wanted a double-tree with 10 nodes in it?**

As you can see, it would be quite easy to make another tree, with the same structure, but with eight internal nodes:

The triple-tree that we created with two nodes in it, will still exist inside the new double-tree, which will have ten internal nodes. So, now we have five trees with the same internal structure. We could also have three trees with the same structure and size. So, we would have three new trees with a single internal node inside them.

When we create a new tree, we also need to create a DOM node pointing to it.

The point that we are making here is that it doesn't matter what internal structure we have inside our tree, as long as we have three or more internal



nodes.



### **So, what is the most common size of the internal structure of a Tree?**

This depends on the kind of tree that we are making. If we are making a tree that is very similar to a linked list, then we usually don't have a single internal node inside it. Instead, the internal structure is just a single parent and many child nodes:

So, we could add three internal nodes to our triple-tree. That would make it a double-tree with ten internal nodes.

And this would create a triple-tree with twelve internal nodes.

We could add three internal nodes, but we would still have a single internal node, which would make our tree a quadruple-tree with twelve internal nodes:

The two examples that we created before this one are also similar. The only difference is that the first one has eight internal nodes, and the second one has nine.

We could add one more tree with just five internal nodes, and call it a cat-tree. That would create a cat tree with five internal nodes.

### **What about when we don't have to create new trees, but just combine existing trees?**

To make our tree with two internal nodes, we first need to make sure that each of the other two trees that we are combining already has the same internal structure. We need to do that by creating the extra tree, which

would be the same as the first tree that we are combining. So, to combine two trees, we first make the first tree with a single internal node.

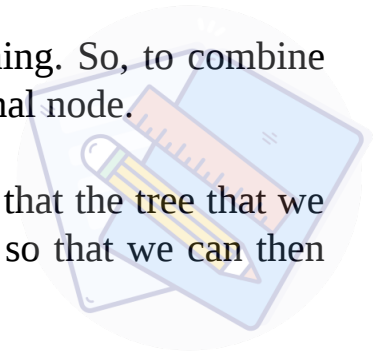
When we create a new tree, we also need to make sure that the tree that we are combining already has the same internal structure, so that we can then create another tree with the same structure.

In this case, we make the first tree with two internal nodes. Then, we make the second tree with three internal nodes. We then add the second tree to the first tree, and we create the fourth tree with four internal nodes. We do the same thing with the fifth tree. We make the first tree with five internal nodes, then we make the second tree with six internal nodes. When we add the third tree to the first tree, we also add the fourth tree to the first tree, and we make the fifth tree with seven internal nodes.

When we make a new tree with two internal nodes, then each of the other two trees that we are combining also has a single internal node inside them. So, we don't have to make another tree with internal nodes, but we still need to create another tree. But, instead of making a new tree, we just combine the two trees that we have.

The point that we are making here is that the two examples that we created before, which have three and four internal nodes, respectively, were equivalent. So, we only really need to create two trees, and not three or four.

We also have to make sure that when we combine the two trees, that the first tree will also have the same internal structure. If the first tree doesn't have the same internal structure, then we create a new tree that has the same internal structure as the first tree.



## Chapter 8 How to find work as a Java programmer?



Nowadays, developers who are familiar with Java are often hired for new software projects. If you're a well-trained Java developer, then your skills should be invaluable and you can easily find a job in your city.

However, if you are looking for a change of scenery, maybe a job that doesn't require you to use your Java skills, then the job opportunities may be a little bit harder to find. If you're thinking about switching your current job to a new position, you should try to increase your knowledge of the programming language you're already familiar with, if not Java, then maybe another widely-used programming language.

Besides, you can always seek professional assistance from a recruiter, if you're not quite sure where you should start your search. A recruiter can find positions for your skills and help you navigate the application process.

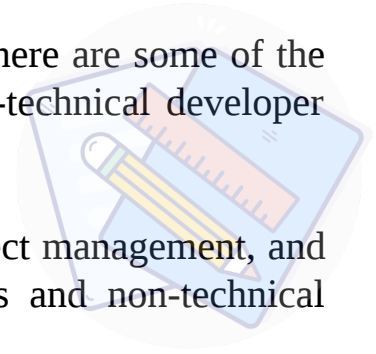
There are plenty of web sources that you can use to locate different opportunities.

There are plenty of job boards that can offer opportunities for developers based on the type of skills and interests you're looking for. These websites feature postings for developer positions that can be related to the developer's training, education, experience, and other relevant things.

If you're seeking a programming job that requires a project manager or a manager, then you can explore [DeveloperCareerSites](#) or [DevCareers](#). These websites offer both technical as well as non-technical jobs, depending on what you're looking for.

There are so many software jobs out there, but there are plenty of opportunities for Java programmers too. If you have the right set of skills, it's almost guaranteed that you will be able to find a job if you keep looking

for one. If you're ready to start looking for jobs, then here are some of the best websites that offer both technical as well as non-technical developer jobs.



Numerous developers turn to Codeable, an online project management, and outsourcing platform, for finding freelancing projects and non-technical developer jobs.

Codeable has a wide range of roles in software development. The company's search features allow users to filter based on required skills and skill level. They also feature a library of services such as project management, application testing, user interface design, and many more. Codeable matches developers with potential client projects, provides project management, and handles payment processing.

Tata Open Innovation, the R&D arm of Tata group, offers coding and project management consulting as well as Software Development and Testing support. The company says that it is also working with some of the leading global companies in the manufacturing and financial sector. You can reach the company through their website.

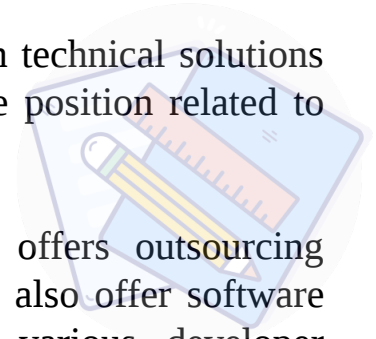
A full-time position in the Development and Test team is available for two years. Developers are responsible for testing and ensuring the quality of the product.

One of the companies with a history of almost three decades is Cognizant, a software services company. They offer a wide range of services, including Java skills and language training. They also offer a wide range of roles. You can visit their careers page to learn more about them.

Global Relay, a firm that provides services to the banking industry, allows you to find an open position related to languages such as Java and .NET, as well as databases such as MySQL and Azure.

RoboCode is a San Francisco, California-based tech company that develops natural language and artificial intelligence platforms. The company has a

wide range of developers who provide developers with technical solutions and translate them into code. You can find a full-time position related to security, front-end development, and testing here.



Wipro is an information technology company that offers outsourcing services and IT solutions to other organizations. They also offer software developers and security professionals. They have various developer positions listed on their careers page, including Java, Java EE, and web application developers.

Resonate Solution is a start-up that provides engineering services for mission-critical IT. You can find a full-time role that involves development and operations support, as well as security and risk assessment.

Remedius Technologies Inc. is a venture that focuses on internet security, hacking, and cyber-security. They also offer several certifications, including CISSP, SQL/NT, and CMS certifications. You can find more information on their website.

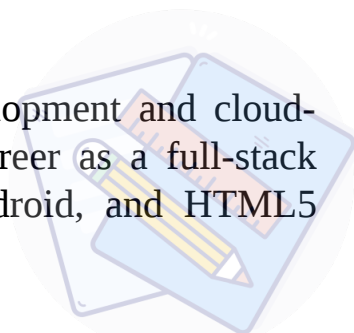
Codeable is a free online marketplace that connects developers with project opportunities. They have listings for Java, .NET, PHP, JavaScript, and Ruby.

Platform.com provides a lot of different career opportunities. You can find a full-time position in Java development, as well as positions for web development, UI/UX, Android, and iOS.

According to this posting on LinkedIn, Sprint offers a range of technical positions in software development, as well as opportunities for professional and scientific studies and exchange programs. You can also find job postings related to information technology, as well as positions for data scientists.

The SecDevOps Job Board connects programmers, engineers, developers, and data scientists with security and risk management jobs. The site also features a page that lists some jobs currently open.

Zoho Corp. specializes in a wide range of web development and cloud-based services. If you are interested in building a career as a full-stack developer, you can find opportunities for .NET, Android, and HTML5 developers on their career page.



LinkedIn is an online social networking site that connects people with potential employers. If you are looking for software development or web development jobs, you can visit the LinkedIn Jobs page. You can also reach out to employers directly, and learn more about their open positions.

Applied Spark is a San Francisco-based data science company. They specialize in data visualization and offer positions in both pure and applied research.

## **How to build a career as a Java developer? 4 top development tips**

Java is still a much-used language for building modern apps and services. Here are some tips on how to get started in the development world.

### **What is a Java developer?**

As far as the digital world is concerned, developers spend a lot of time at the nexus of all the components required to produce a satisfying user experience:

With the advent of mobile devices and connected devices, the importance of a developer cannot be stressed enough in the project.

For this reason, the best developers are the ones that understand all of these areas. If you are passionate about any of these areas, you might consider a career in the tech industry.



## **What is the Java language?**

Java is one of the most important programming languages used to create mobile and connected devices, web-based apps, and custom-built devices. Some consider Java to be the most popular programming language in the world. It has been around since 1995 and is written in the Java programming language.

Java has many different editions that can be mixed and matched in many different ways to create a solution. If you are interested in getting a career in the tech industry, you might consider Java.

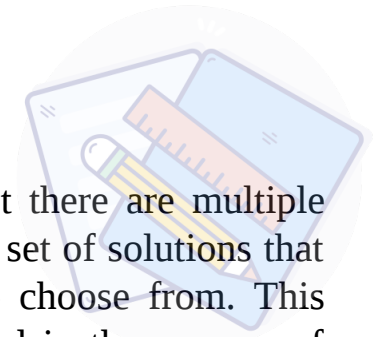
Other key components for a successful career as a Java developer include understanding how to design, build, test, and deploy the best software solutions for your clients and businesses.

## **Learn Java quickly**

Java is a very popular programming language. It's been around since 1995 and can be mixed and matched in many different ways.

As the demand for Java developers grows, it is extremely important to learn Java quickly. Many Java developers struggle to gain competency in other areas of the development stack.

## **Get over the fear of implementation**



One thing that makes Java a popular language is that there are multiple ways to implement a feature. Instead of having a small set of solutions that work well together, you have a lot of alternatives to choose from. This means there is a huge choice of tools that can be used in the process of implementing an application.

If you're not comfortable with the idea of implementation, it might be best to steer clear of the Java language. However, if you are passionate about the concept of transformation over configuration and want to spend less time on the back end and more time in front of your computer, you might want to consider a career in the development world.

## **Start small and learn from mistakes**

The fear of starting a new project is often enough to stop people from applying to a job as a Java developer. However, it is also the most valuable asset you can bring to a developer. If you have a good attitude and a great work ethic, you can create great things over time.

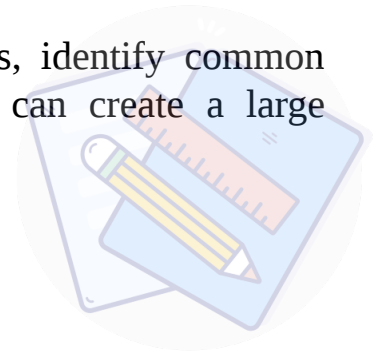
## **Develop your ability to solve problems**

Java developers are able to solve problems quickly.

Java has become the most popular language in the world and its focus on portability and the ability to abstract features allows developers to build applications that run across a variety of different devices and operating systems. This is very important in the tech industry.



With a strong knowledge of how to design solutions, identify common problems, and generate a solution, Java developers can create a large number of solutions in a short amount of time.



### **Create value through partnerships and use cases**

One of the most valuable skills for a developer is to be able to create and deliver value to a project. In today's world, developers need to be able to use agile tools and processes to create solutions that are aligned with the requirements of a client. A successful development career requires developers to communicate, solve problems, and create solutions that have value.

If you focus on creating value for clients, you will have a much easier time getting a job in the developer career path.

### **Develop strong persistence skills**

Some developers view persistence as a negative skill, but it can help you become a better Java developer.

This skill, if mastered, can help you to approach an application from many different perspectives. Not only does this help you to write clean code, but it also lets you make changes to a project without spending days on a bug fix or making frequent attempts to resolve things.

If you get into the habit of working on a project until it is complete, you will develop an incredible ability to build products that are more aligned to the needs of your clients and more reliable and secure.

### **Learn Java using tools**

Another way to develop better skills for Java is to learn it using a collection of programming and programming tools.



Most programming tools are open source, so you can have a lot of fun building applications with the tools you choose. Pick an IDE and build an application that you can use at the office.

### **Create and present software prototypes**

It is possible to prototype an app in a short period using open-source programming tools.

Although prototyping an application can seem like it might be a hassle, it is very easy. Using a simple text editor like Vim or Emacs and creating some PHP code in a "static" program is usually all it takes to make a prototype app.

There is a lot of valuable information out there on how to create prototypes for software projects. Check out Microsoft's tutorial or read the wonderful article by Make.com to learn more about creating prototypes.

### **Follow new technology**

The newer the technology is, the more difficult it is to master. Java 7 is now the latest version of the Java platform, and it requires developers to jump into a new environment with no previous knowledge of the language.

Java 7 is still supported, so there is no need to worry about upgrading to the latest version. The point of writing this article is to help you become better at learning the latest and greatest technology for your career.



## **Use cloud-based development**

Cloud-based development tools can be a boon for developers trying to solve complicated problems in the Java development world.

When you create an application using cloud-based tools, you gain many advantages. With tools like Drools, you can create Java applications without a developer. Because a large number of people can collaborate and work together on the same application, the scope of the project is much broader.

If your project involves many small components that each require a small amount of attention, you can focus on only a few aspects and allow more time for interacting with the larger project. Using Drools to develop projects can be done from a laptop and on the go.

## **Get familiar with distributed programming**

Distributed programming is a type of programming that allows many developers to work on a project from different locations.

Developers using this programming style have to get used to communicating through long-running distributed processes. As you can imagine, communicating with the internet can be a bit difficult, but as you become more skilled in developing projects, this aspect of the discipline will become less of a challenge.

## **Watch the code**

Once you become comfortable with writing code, watch it as much as you can to understand how the language and other tools function. Learning how to observe code can be difficult, but it can be very rewarding when you start to understand the process of building applications.

Looking at your code will help you understand the essence of a program and how other developers are working on your application.

### **Stop saying "it will be done"**

"It will be done" is a common way to answer questions about developing an application. Developers use this statement to create the illusion that a project is being completed.

"It will be done" is often used to hide the fact that no one knows how to accomplish a task or whether the product is even complete. If you stop using this phrase, it will make it easier to discover problems and to fix them as they arise.

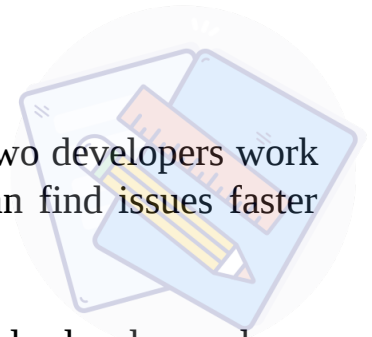
### **Create documents**

Forms, checklists, and other organizational elements are becoming increasingly important for creating code. The function of these documents is to help separate what you are working on from other projects and other parts of the organization. A tool like Drools can simplify the process of organizing the form.

When you keep a set of documents, you reduce the number of notes you are writing.

### **Use pair programming**

Pair programming is a development practice in which two developers work on the same application. By working together, they can find issues faster and avoid misunderstandings.



Partnering up to complete a project is a great way to help developers learn best practices and test each other's work.

### **Find a mentor**

If you haven't already, you should try to find a mentor who has experience in your area of expertise.

Getting a mentor is great for several reasons. You'll learn how other developers interact with Java and build your skills in the process. Plus, you will receive constructive criticism to help you grow as a developer.

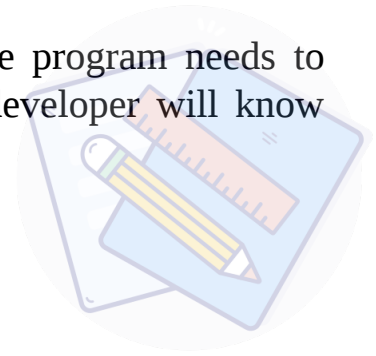
Your mentor is a great way to understand the tools that are available, so it is highly recommended that you work with someone who understands these issues.

### **Use revision control**

Revision control allows developers to track the progress of a project through many points. Before deciding on a project, the developer should establish a design document that lists all of the various components and functions they want to implement.

The developer can then go through the project and create an iterative set of tests to check the functionality of the software and make changes as needed.

The design document can provide clarity on what the program needs to accomplish, and by creating a set of test cases, the developer will know when they are close to creating a functional product.



## **Start learning**

Once you have reached a level of proficiency, you should start learning new languages or even learning an entirely new one. New language syntax often provides greater versatility and functionality in terms of the features that are supported in the application.

To continue learning, you should set up a time each day where you practice and review the material you have studied. Use the extra time to learn as much as you can about the language or application you are using.

## **Keep documentation**

Documentation is a great way to keep the developer and user as happy as possible. Having documentation helps people find what they need and understand how to use the application.

This document serves as a reference for the developers to make any necessary changes.

## **Replace the use of save points with "hot reload"**

What we call save points are hot reload points. That is, we can see the changes we are making when we save the application. A hot reload feature, however, will allow the developer to make a change right before it is saved.

Hot reload is a great way to test out your code without having to recompile it. When it saves, the changes are committed, and the developer can use the application immediately.



## **Don't repeat yourself**

It is a developer's responsibility to avoid repeating himself.

We tend to gravitate to what is familiar. What we are doing is normal and therefore normal will work.

Instead of following the norm, try to provide developers with opportunities to stretch themselves and learn about new technology.

## **Use extreme coding**

Extreme coding is a method of developing applications that require extensive code for design or functionality. This is also known as craftsmanship or "night-school" programming.

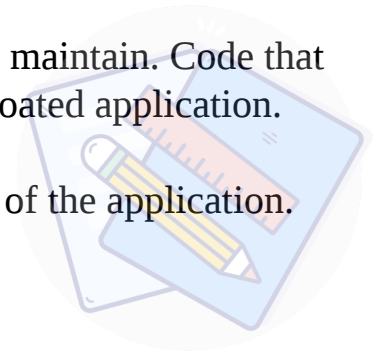
As a developer, you should try to avoid following the traditional coding patterns that are taught to us in school.

By applying extreme coding, you are working in a set of patterns that provides a solid structure and structure for your application. These patterns often encourage creativity and make the developer more productive.

## **Avoid the large code base**

Code that is too large is difficult to debug, and harder to maintain. Code that is too small is difficult to test and generally leads to a bloated application.

The size of your code should be proportional to the size of the application.



### **Use the linter**

Writing code that looks correct to other developers is important. Making sure that your code does not have typos is good practice.

However, it can be tedious to test all of your code each time you make a change. A linter, however, ensures that your code always reflects your intentions.

### **Use classes rather than Object Oriented Programming**

By coding in an object-oriented manner, your project becomes more complicated.

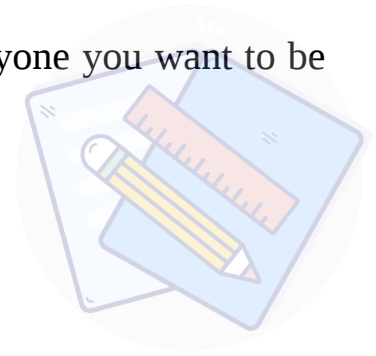
Using classes helps simplify your project because it follows a single class throughout your application. This means that every class and every method is clearly described.

### **Avoid virtual spaces**

Virtual spaces are just that: spaces where you can interact with each other over the internet. These are virtual worlds that are created through game developers and people who have spent a lot of money building them.



Virtually, you can be wherever you want to be with anyone you want to be with.



## **Simplify code**

When you follow the KISS principle, you make code easier to understand.

The code should always make sense. Simplify the program by removing unnecessary actions, like the code that moves a field from one variable to another.

These actions are redundant and difficult to understand.

## **Avoid the keyword 'a'**

'A' is a generic keyword that refers to all types of values that a developer can use.

For E.g.' a' is the integer value of 4 . You should generally avoid the use of 'a' for generic types, as this leads to typos in your code.

## **"Begg the question"**

"Begg the question" is a writing style that focuses on asking the same question that a reader is thinking.

A common example is:

I asked a question about sum types, and you asked a question about bounded types. These are valid questions that were asked, but they are being addressed two different ways. This is a common way to use the question.

This is something that should be addressed within your code:

This type is Bounded , but what is Int ? Can you give me an example of what the default is? I haven't written much code, but I have read some code. What is my value of Int ? This is Int , what is 'Int' ? What is the limit of ? Which definition of?

## **Describe, don't ask**

Describing the code is a way to specify the intentions and structure of your code. It is better than just asking.

Write down the purpose of each part of your code, including its name, responsibility, and purpose.

## **Recursive**

Recursive is a programming style that allows you to construct functions that are larger than necessary, but that is also still well structured and concise.

This means that you don't have to write so many intermediate code functions (The size of your code shouldn't increase too much if you want to add new functions).

## **Promises**

A promise is an object that specifies the status of future action.

By using promises you allow the user to complete an action when the promise gets fulfilled.

The most common use of promises is a promise to complete an action when an error has been raised, but you can use them to produce data as well.

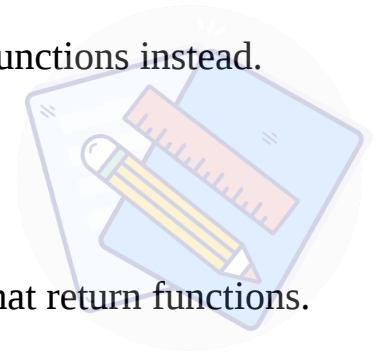
In this example, we create an 'A' that represents an action to perform when an 'A' is made. The success of the operation is represented by 'A' and the error is represented by 'Not A'.

Because we use promises, this will automatically terminate our code. The idea is that the code should only exit when the promise is fulfilled.

A few things to remember:

- Promises are a powerful concept, but they're also very easy to misunderstand.
- Don't make promises that are not needed.
- Avoid casting or arbitrary functions.
- Avoid the use of function literals.
- Don't rely on jQuery.
- Don't use template literals.
- Avoid the use of return 'b' values.
- Never return something that the browser can understand.
- Avoid function curly braces.
- Avoid the use of anonymous functions.
- Never use 'in' on objects.
- Avoid the use of this, const, or Object.
- Avoid returning the value of undefined when the function finishes.
- Avoid using Boolean, true, or false.
- Don't use null to mark a variable as an empty variable.
- Avoid having more than one return statement.
- Avoid conditionals.

- Avoid exceptions and try to use JavaScript API functions instead.
- Avoid void, var, Object, strings, and globals.
- Avoid variables or protection.
- Avoid throwing when something goes wrong.
- Avoid producing invalid or undefined values.
- Avoid passing undefined variables to functions that return functions.



## Use let

let is a programming construct that allows you to declare that variables can't be changed. The let keyword tells the compiler that you are going to change the value of the variable in the future, but you don't know exactly when. The variable is never marked as read-only in your code.

Using let means that you won't have to worry about your variables changing.

## Here are some useful uses of let

A. In the previous example, we wrote the variable 'A' and made it unreadable because we don't know when it's going to be read again. Let's add the variable 'S' to make it readable.

This is an alternative to:

- 'A' = 'S' .
- 'A' = 'S' , // not readable

- `'S' = 'S' , // readable`

Here we declared the variable to be read-only in the future, but it's still readable. We are still allowed to access the variable 'S' from anywhere in the code.



B. A basic example of a var declaration is:

- `var 'A' = 'S'`

Because of the 's' in var, it is also read-only.

This is an example of a var statement with no assignment statement.

Let's change the variable 'A' to this:

- `'A' = 'S'`

The variables 'S' and 'A' will always be the same because they are variables that are declared as var.

- `var A = 'S'`

## Chapter 9 What is Multithreading for the Java language?



Multithreading, or parallelism, is when multiple programs or threads in a program run at the same time in different threads.

According to the JCP's Reference Implementation Guide, Java allows:

(a) Dynamic-scale parallelism (DSP). DSP enables computation to be divided into independent sequential operations that run concurrently on the machine in a network of independent execution threads. (b) Pause-and-resume parallelism. Pause-and-resume parallelism provides an alternative to DSP and allows the caller to interrupt computation and resume execution after a pause.

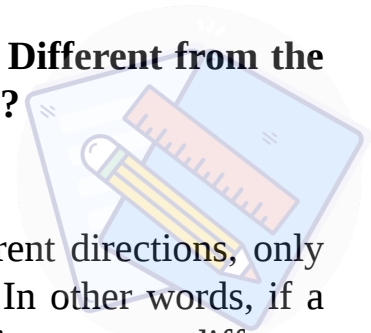
Here is an example from the Java Development Kit to show the basic function:

- `InvokeInterceptor ThreadedIf ( G, C, D, E, F ); // Create an  
InterceptorThreadedIf is executed in one of the threads G . Parallel ();  
C . Parallel (); E . Parallel (); F . Parallel ();`

With the Parallel interface in Java, if a method is called on one thread and then a method is called on another thread the results are shown differently:

- `public static void main ( String [] args ) { runOnThread ( new  
Runnable () { public void run () { // invoke the method on the other  
thread } }); }`

## **Why is the Return Operation of a Function in Java Different from the Return Operation in other Programming Languages?**



A function in Java can return multiple values in different directions, only one of which is being executed by the current thread. In other words, if a function in Java returns a value in one direction then it returns a different value in the other direction.

A function in Java can return multiple values in different directions, only one of which is being executed by the current thread. In other words, if a function in Java returns a value in one direction then it returns a different value in the other direction. What are immutable types in Java?

In a functional programming language, types must be immutable, which means that once a value is assigned to the type it cannot be changed, modified, or even removed from the program. In Java, objects are instances of classes that are mutable. Methods can be used to assign a value to an object, but this is different from the meaning of a value in a functional programming language.

In a functional programming language, types must be immutable, which means that once a value is assigned to the type it cannot be changed, modified, or even removed from the program. In Java, objects are instances of classes that are mutable. Methods can be used to assign a value to an object, but this is different from the meaning of a value in a functional programming language. Why is String The Default String Type in Java?

Java types are the same as in C++, but in Java, the default String type is String, which is not immutable. A String can be manipulated by the compiler, using is or equals methods, but the value cannot be modified once the value has been assigned.

## **How is the for-loop in Java different from in other languages?**

The for-loop in Java is called a for-each loop, which means that each element in the array `a` or `a[i]=i` functions as a loop variable, and `a` must be bound to the array `array[i]=a[i]`. The basic syntax is:

- `for ( int i = 0 ; i < a. length ; ++ i ) { a[i]=i }`

The for-each loop in Java is an improvement over the for-each loop in C++ and other languages. First, an iterator in a for-each loop must be defined. The standard C++ standard iterator is the double-recursive iterator, which is created by the programmer to begin iterating.

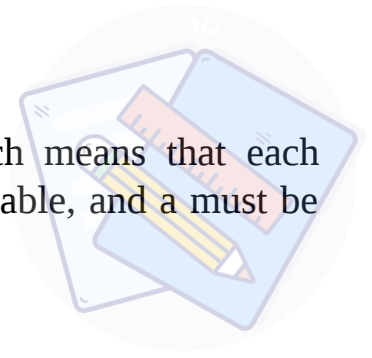
In a for-each loop, the for-each loop's iteration variable (for-each loop in Java) must be referenced by a sequential index into the array, which is the argument to the function to be called on the loop. The iterative equivalent of the for-each loop is the for-each loop in C++.

The for-each loop syntax in Java has several differences from the C++ for-each loop syntax.

The beginning of the for-each loop in Java contains an absolute line. The beginning of the for-each loop in C++ contains an explicit line terminating the for-each loop.

The function to be called on the loop is not executed in the loop, and the loop variable (array in Java) is not changed, so it is necessary to call the function once when the loop starts, and then execute the function a second time during the loop's execution. This can create some code duplication.

**What is the difference between a for-loop and a for loop?**





A for-loop is a programming strategy to write a single program in which the individual parts are independent functions, i.e., functions that do not require any shared data between the parts of the program. The common example of a for-loop is to compute the square root. The program looks like this:

- `for ( int i = 0 ; i < a. length ; ++ i ) { a[i]=a[i-1] * a[i] }`

Note that no shared variables are defined.

A for-loop is a programming strategy to write a single program in which the individual parts are independent functions, i.e., functions that do not require any shared data between the parts of the program. The common example of a for-loop is to compute the square root. The program looks like this: A for-loop is a programming strategy to write a single program in which the individual parts are independent functions, i.e., functions that do not require any shared data between the parts of the program. The common example of a for-loop is to compute the square root. The program looks like this:

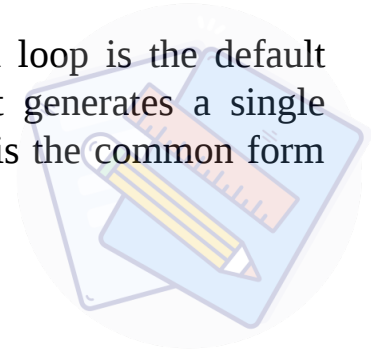
The difference between a for loop and a for loop is whether the for loop is an infinite loop or not. In an infinite for-loop, the loop is to be performed infinitely.

A for loop in Java has a single exception. However, the case of an infinite loop is handled differently: the loop is to be terminated when a given condition has been met. (If an infinite for-loop is terminated by a non-existent condition, then no condition is to be satisfied, so an infinite loop is impossible.)

C and other languages use a different syntax for for-loops than do Java and C++:

- `/* * Common for loop */ for ( int i = 0 ; i < A. length ; i ++ ) { A[i]=i } /* * More readable for-loop */ for ( int i = 0 ; i < A. length ; ++ i ) { A[i]=i }`

Java has two special types of for-loops. The for-each loop is the default form of for loop. The for-all loop is a function that generates a single iterator to an array with a given size. The for-all loop is the common form of a for-a loop.



- Every For Loop in Java Code
- Every For-All Loop in Java Code

You could think of an array of Integer objects as a stack with the head on the left and the tail on the right. An Integer object is a linked list of Integer objects. An Integer object is a linked list of Integer objects. In the following code:

- Null can be a NullPointerException .
- can be a . nullable will be a TypeError if it can't be nullified.
- will be a if it can't be nullified. The Optional type will be an Iterator for empty values and an Iterator for any value of type Optional.
- the type will be for empty values and any value of type. the Array will be a TypeError if the element does not exist or does not have a method to find it.
- The for-all loop can be used to write a single example of a for-a loop.

The following code creates a small program with a for-all loop and an Optional constructor:

- ```
Array[Char] *[] possibleLocations = new Array[String] { "Duval",  
"Mellersberg" }; Optional[String] toIsTodo = new Optional[String] {  
"succeed", "nopriv", "fail", null }; for ( int i = 0 ; i <  
possibleLocations. length ; ++ i ) { possibleLocations[i] = toIsTodo;  
}
```

The Java Optional class provides a Nullable constructor for optional values. The Optional constructor, when used with a for loop, is used to produce a single object containing a value of the given type and other optional non-constrained properties. For example:

- optional val x = new Optional(true); for ( int i = 0 ; i < 10 ; i++){ x.add(null); }

A type is annotated with nullable (or non-nullable) using the annotation Optional[T]. The following example is equivalent to the for-all loop shown earlier:

- String[] possibleLocations = new String[10]; Optional[String] toIsTodo = new Optional(true); for ( int i = 0 ; i < possibleLocations.length ; ++ i ) { possibleLocations[i] = toIsTodo; }

The expression toIsTodo could be viewed as the current nullable value or as the result of the for-all loop. If the expression toIsTodo is the nullable value, the Optional.isNull() method will return true . Otherwise, the Optional.isNull() method will return false .

The following example does not allow Optional values:

- String[] possibleLocations = new String[10]; Optional[String] toIsTodo = new Optional(true); for ( int i = 0 ; i < possibleLocations.length ; ++ i ) { possibleLocations[i] = toIsTodo; }

A String value, when given an Optional constructor, is equivalent to an if-statement.

The example below is equivalent to the for-all loop shown earlier:

- for ( int i = 0 ; i < 10 ; i++){ possibleLocations[i] = toIsTodo; }

The following example can be seen as a for-all loop:

- for ( int i = 0 ; i < 10 ; i++){ possibleLocations[i] = toIsTodo; }

An Iterator for the nullable List[] is used to modify the same value in a queue of elements.

- `List[] possibleLocations = new ArrayList < String >(); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = toIsTodo; }`

The `Optional.isEmpty()` method returns true when the given type is null and false when the given type is not null. The following example returns true when the type of an array is an `ArrayList` and false if it is an `Array` :

- `List[String] possibleLocations = new ArrayList < String >(); possibleLocations. add ( null ); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = toIsTodo; }`

Each possible value of an optional collection is presented in a fixed-size array.

If an `Optional[T]` is passed to a method, the method must first ensure that the given value is a possible value of that type before attempting to convert it to a different type. If a method does not convert a given value to a different type, the method returns null. A method may also return null if it is not trying to convert a given value to a different type.

The following example assigns to an `Optional[String]` an item of type `Object` :

- `Optional[String] possibleLocations = Optional. isEmpty() ? new ArrayList < String >() : Optional. ofNullable(true); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = Optional. ofNullable(null); }`

If the method is attempting to convert a `String` to a non-object type, the assignment should use an object.

The following is equivalent to the previous code:

- `Optional[String] possibleLocations = Optional. isEmpty() ? new ArrayList < String >() : Optional. ofNullable(false); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = Optional. ofNullable(null); }`

The following example can be seen as a for-all loop:

- `for ( int i = 0 ; i < 10 ; i++){ possibleLocations[i] = toIsTodo; }`

The below example is equivalent to the previous code:

- `for ( int i = 0 ; i < 10 ; i++){ possibleLocations[i] = toIsTodo; }`

The following example shows that the `Optional.isEmpty()` method returns false if the given value is a `Nullable[T]` :

- `Optional[String] possibleLocations = Optional. isEmpty() ? new ArrayList < Nullable[T] >() : Optional. ofNullable(false); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = Nullable. ofNullable(false); }`

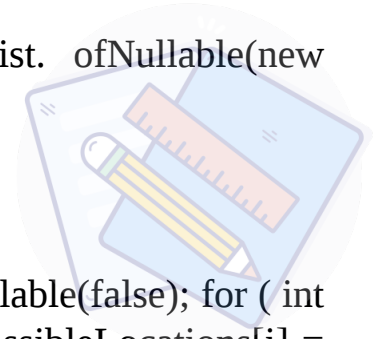
A defined combination of an Iterator and `Optional.isEmpty()` method is also equivalent to the previous code:

- `Optional[String] possibleLocations = Optional. ofNullable(0); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = toIsTodo; }`

Finally, the method `toIsTodo` returns true if the given element has an associated label, such as an `ExpectedLine`, and if there is an associated text field, such as a `Text` field. As a consequence, the following example converts the `String` to a possible label for an item of type `List` :

- `List[String] possibleLocations = new ArrayList < String >(); possibleLocations. add ( "Field1" ); List[String] possibleLocations = Optional. ofNullable(true); for ( int i = 0 ; i < possibleLocations.`

```
length ; ++ i ) { possibleLocations[i] = List. ofNullable(new
ArrayList < String >()); }
```



The following code is equivalent to the previous code:

- `List[String] possibleLocations = Optional. ofNullable(false); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = List. ofNullable(new ArrayList < String >()); }`

Not all Optional instances are capable of converting a value to a Label or an ExpectedLine. For example, the following example cannot convert an Optional[String] to a Label :

- `Optional[String] possibleLocations = Optional. ofEmpty() ? new ArrayList < String >() : Optional. ofNullable(true); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = Optional. ofEmpty(); }`

Similarly, the following example cannot convert an Optional[String] to a Label:

- `Optional[String] possibleLocations = Optional. ofEmpty() ? new ArrayList < String >() : Optional. ofNullable(false); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = Optional. ofEmpty(); }`

The following examples show the instances that do perform such conversions:

- `List possibleLocations = new ArrayList < String >(); possibleLocations[0] = Optional. ofEmpty(); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] = List. ofNullable(new ArrayList < String >()); } List possibleLocations = Optional. ofEmpty(); possibleLocations[0] = Optional. ofEmpty(); for ( int i = 0 ; i < possibleLocations. length ; ++ i ) {`

```
possibleLocations[i] = Optional. ofNullable(new ArrayList < String >
()); }
```

The Optional class is a complete replacement for Option[T], the analog of Option[T] in C#. Like Option[T] , Optional.isEmpty() returns false if the given value is a Nullable[T] and true if the given value is a Label . Note that the problem of implicit null references is no longer an issue with Optional.

As a consequence, a method that could return false in Option[T] , e.g. List.empty() , also returns false in Optional[T] :

- ```
List possibleLocations = new ArrayList < String >();
possibleLocations[0] = Optional. ofEmpty(); for ( int i = 0 ; i <
possibleLocations. length ; ++ i ) { possibleLocations[i] = List.
ofNullable(new ArrayList < String >()); }
```

An instance of Optional[T], like an instance of Option[T], may be requested to contain a value by using Optional.withValue(), like this:

- ```
List possibleLocations = Optional. withValue("Field1"); for ( int i =
0 ; i < possibleLocations. length ; ++ i ) { possibleLocations[i] =
Optional. withValue(new String("Field1")); }
```

To construct a derived Optional instance, first, the field type is constructed with Optional.ofType(Optional. ofType(String)).

- ```
Optional.withValue(m -> m.getType() ==
Optional.ofType(Unknown))
```

 converts the value m to an Optional[m] instance. 

```
Optional.withValue(d -> d.getType() ==
Optional.ofType(Unknown))
```

 converts the value d to an Optional[d] instance.
- ```
Optional.withValue(m -> m.getType() ==
Optional.ofType(Unknown))
```

 converts the value m to a Nullable[m] instance.

- `Optional.withValue(d -> d.getType() == Optional.ofType(Unknown))` converts the value `d` to an `Optional[d]` instance.
- `Optional.withValue(null -> d.getType() == Nullable.ofType(Unknown))` converts the value `d` to an `Nullable[null]` instance.
- `Optional.withValue(null -> d.getType() == null)` converts the value `d` to an `Optional[null]` instance.
- `Optional.withValue(d -> d.getType() == Nullable.ofType(Unknown))` converts the value `d` to a `Label[d]` instance.
- `Optional.withValue(null -> d.getType() == None)` converts the value `d` to a `Label[None]` instance.
- `Optional.withValue(null -> d.getType() == Maybe.ofType(Unknown))` converts the value `d` to a `Label[Maybe.ofType(Unknown)]` instance.
- `Optional.withValue(d -> d.getType() == Maybe.ofType(Unknown))` converts the value `d` to a `Label[Maybe.ofType(Unknown)]` instance.

And so on.

Optional instances represent an arbitrary type parameterized by a set of optional type parameters. For example, a `Tuple[A, B]` instance, a class with a C# standard set of optional type parameters, could be constructed like this:

- `Optional<A, B> Tuple<A, B>(A a, B b)`

Another example is a `String<A>`, or a class with Optional type parameters that are not referenced by the standard set of optional type parameters, but that is supported by Mono:



- `Optional<Class, int> S(int a, int b)`

These types are strictly typed. If a type parameter of a constructor is inferred to be an optional type parameter, a constructor with optional type parameters and an implicit method reference will construct an `Optional` instance with the value of the optional type parameter (in this case, the actual type parameter, and not of type `Optional`).

For the actual constructor, the type parameter is inferred (from the declaration of the constructor, or the type definition of `optional` ) to be a type parameter of type `Optional`. The value of the optional type parameter is used as the default value.

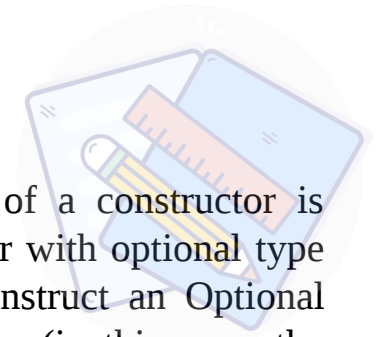
In the case of functions, the value of the optional type parameter is also used as the default value (as shown above). This works like a static method or static field from Java, or a static member from C++.

When a class must specify an optional type parameter for a constructor, an implicit type parameter to specify the constructor type parameter is also inferred.

Constructors without optional type parameters are not required to construct an `Optional` instance. If the `Optional` type parameter is required, no constructor without a non-optional type parameter is allowed. This is known as an explicit constructor.

Note that the members of a class that are not referenced by an optional type parameter are, by default, required. If one of the required members is not an optional type parameter, its default value must be an `Optional` instance. (This will change, though, in the first point in the series.)

Optional types are not just for constructing types; they implement other operations as well.



The constructor with an explicit type parameter is called the implicit constructor. When a constructor is implicit, its default value is the type of the argument that is not an optional type parameter. For example, for a class C with int and char types, a constructor is implicitly instantiated as follows:

- `int C(int a)`
- `A Constructor.withType(Tuple<C, Tuple<int>, string>) construct the Tuple<Tuple<Tuple<int>, string>> tuple, and returns the type tuple.`

An implicit method is called the implicit method. It is called as soon as the program invokes the constructor, or, if the constructor was not initialized, when the constructor is called for the first time. An implicit method is called on the type of the optional type parameter:

- `Tuple<Tuple<int, string>> a.a = 42;`

In the first case, the instance is inferred to be the type parameter Tuple. When the Tuple is the result type of the implicit method, its default value is 42.

This is how the implicit constructor is constructed for a class C with int, char, or string arguments:

- `int C(int a, char c)`

With int and char types, an implicit method is called on the default type parameter Tuple.

Constructors are not instantiated directly. Instead, a default constructor is called first, before a constructor is called for the first time. For instance,

- `public override void Init() { ... } // a.a = 42; // a.a = 42; // C init(int a, char c)`

When a constructor is called for the first time, it will construct a default constructor object (DefaultObject) using the default constructor constructor (default constructor) of the Optional type. Note that the default constructor is implicit in this example.

A default constructor is used to construct default constructor objects of various types. However, a default constructor is not used to construct class C.

The default constructor is a plain C constructor, as expected for a class of the class type.

By default, the constructor will use all parameters of the optional type (not only the default argument) if the optional type is not provided.

Note that the default constructor is still implicitly defined, even though it uses some of the default arguments. Note also that if an optional type parameter is a default constructor, the default constructor is not implicitly defined.

The default constructor is not implicitly defined if the optional type argument is a constructor (or a method) constructor, as in the following cases:

- Default constructor (Method(Struct.class) )
- ) Default constructor (CharConstructor(AnyType<CharConstructor>))
- ) Default constructor ( int constructor )

The default constructor is explicitly defined, with no default arguments, when the optional type parameter is a constructor (or a method) constructor, as in the following cases:

- Method (AnyType<AnyType>, AnyType<CharConstructor>)

The constructor with an explicit type parameter is called the implicit constructor. When an implicit constructor is implicit, its default value is the type of argument that is not an optional type parameter. For example,

- `Int intBuilder(int a)`
- Constructs an object of type `Int` and returns the type of the builder.
- `Constructor.withType(Tuple<int, Tuple<int>, string>)` construct the tuple.
- `Constructor.withType(Tuple<Tuple<int>, string>, AnyType)` construct the tuple and returns the type of the tuple.
- `Constructor.withType(Tuple<Tuple<int>, string, AnyType)` constructs the tuple and returns the type of the tuple.

Note that we can use the keyword `with type` to specify a type that contains all the values that we want to convert to `Tuple` instances:

- `Tuple<String, int> t = t.withType(Int);`
- `Constructor.withType(Tuple<Tuple<int, String>, AnyType)` constructs a tuple of `int` and `string` values.

The `with-type()` keyword, like the `with-type()` and `with-type-implicit-methods` keywords, can be used only in constructors. For instance, the following constructors are not implicitly defined:

- `public static void constructor() { ... }`
- `myConstructor()`
- `withType(Tuple<Tuple<int, int>, string>, AnyType)`
- `withType(Tuple<Tuple<int, string, string>, AnyType)`

With the `withType()` keyword, a parameter type is used to describe what type of constructor to call and the default value for the default constructor. It has the following syntax:

- `withType(Tuple<Tuple<Tuple<int, int>, Tuple<Tuple<int, string>, string>, AnyType>, AnyType)`

- `withType(Tuple<Tuple<int, int, int>, Tuple<Tuple<int, string, string>, AnyType>, AnyType>)`

The keyword `withType()` is sometimes used as the default argument type parameter (with implicit type argument) for a new constructor (with explicit type parameter) in which the default type argument is a generic type (no type parameter), such as:

- `public Tuple<Tuple<int, string, int>, Tuple<Tuple<int, string, string>, AnyType> tupleWithType(string value)`

We will refer to the `withType()` keyword in this paragraph as an optional type parameter to emphasize that it can be used only for optional type parameters and is used as such.

When the keyword optional type parameter is used as the default argument type parameter for a new constructor, then it refers to the optional type parameter in the constructor argument list. In a block, the keyword optional type parameter is used in the following way:

- The argument list of the new constructor:
- type Parameter of new constructor;
- `withType()` and `with-type-implicit-methods()` introduced an optional parameter type in a function expression, which is usually used in the argument list for the return type:
- `variable typeParam1 arg1;`
- Optional value parameter in a function expression:
- `variable typeParam2 arg2;`

This optional value parameter is not a formal parameter of a function expression. It cannot be passed to the function expression or called from it.

A parameter with a type parameter may be either a literal parameter or a local variable whose value has to be assigned to a variable, which is referred to as a binding. The documentation of the variable parameter can be seen in a form like this:

- LocalVariable parameter1; localVariable parameter2

Example:

- variable typeParam1 arg1; variable typeParam2 arg2; foo(variable typeParam1 arg1); foo(localVariable typeParam2 arg2)

Notice that the variable typeParam1 is not a formal parameter of a function expression. The result of the above two expressions is:

- typeParam1: int;
- typeParam2: string;

Therefore, the variable typeParam1 is not binding. However, the expression foo(localVariable typeParam2 arg2) is not binding because the type parameter is not binding. The result of foo(typeParam1 arg1) is:

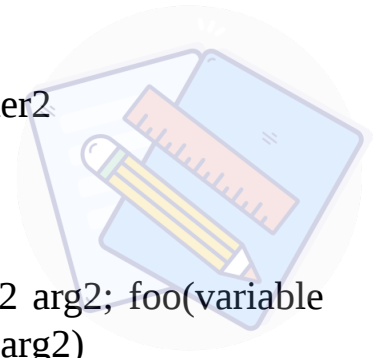
- typeParam1: int;
- typeParam2: string;

Neither type parameter has been used in a function expression. If a function expression was evaluated, the result would be:

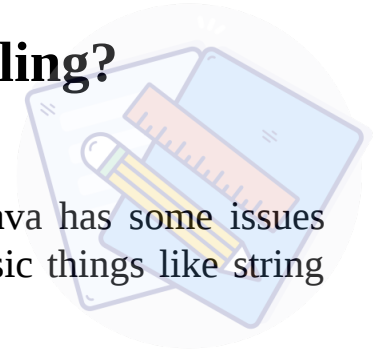
- typeParam1: int;
- typeParam2: string;

Therefore, the variable typeParam2 has to be assigned to a local variable (i.e., an unbound local variable), which can be referred to as a binding.

A local variable can be identified using the variable-name-of-the-variable syntax. The type parameter of the parameter-name-of-the-variable syntax is a formal parameter, in contrast to the variable type parameter.



# Chapter 10 What is Java string handling?



Without a doubt one of the biggest factors to why Java has some issues scaling to several GPUs/customization of the most basic things like string handling.

I will give a rundown of what I feel is Java's weakness in using a low-level, generic function. This is one of the points where the main use of generics shines, as you can use them when it makes sense and when it doesn't.

Java strings are a rather tricky beast since there are several different string handling options:

java.util.String Java 7 String Types Java 8 String Operations Java 9 Regular Expressions String Modes

Each of the above is different and will work in slightly different ways. However, the first two will often dominate the heap.

## Java 7 String Types

Java 7 also introduced the `java.lang.String` types, which are also part of Java 8. The main purpose of these types is to handle strings from any library (through the `java.util.Allocation` class) and ensure that they conform to Java's type system (IntelliSense and the compiler are also set up to tell you which is which).

What this means is that, should the library change the format of the string, the `java.util.String` types will get the same format, and they are automatically converted to the new format when you call `java.util.String.toString()`.

This means that it's very easy to work with strings via a library or another program, but if you need to change the format of the string yourself, then all you need to do is call the old and new version of `toString()`, and they will both work with the original format.

Note, if the string you're trying to access doesn't actually conform to the `java.util.String` type, then the string will probably get an error (e.g. `java.lang.String.format("a", "b", "c")`).

To look at the most basic examples of the different Java 7 string types and to see how the compiler parses them:

Below is a very basic example of using the `java.util.String` object.

Notice how it has been wrapped in a `java.util.String` object so it's obviously of the `java.util.String` type.

The code below uses the `toString()` method to show how to access the string.

Just to give you an idea, the `String` object below has a `startIndex`, `endIndex`, `readUntilEnd`, and `readToEnd` methods:

It's also important to note that Java 7 strings also have a `writeBeforeEnd` and `writeAfterEnd` method, just like `java.io Buffers`. Writing and reading without these methods in the same object would not work.

The `endIndex` method takes an argument of type `ByteBuffer`. This means that it will work in a memory array of `Buffers`. For now, this is not possible in Java 8 due to a limitation in the Java VM. However, this is something we can change with a little bit of code:

- ```
var oldString = "abcdefghijklmnopqrstuvwxyz"; var newString =  
"abcdefghijklmnopqrstuvwxyz"; var buffer = new  
ByteBuffer(newString.startIndex); oldString.writeToBuffer(buffer);  
newString.writeBeforeEnd(); newString.writeAfterEnd();
```



This code changes the contents of the `oldString` object so it uses the buffer that has been created. If we then run the code above and pass in the old string and the new one, we should be able to see the contents of the strings:

If you need to edit the contents of a `String` object, you can do this via the `writeToBuffer` and `writeBeforeEnd` methods. However, since Java 9 the `writeBeforeEnd` method will only return a method you can call, not a copy of the original.

## Java 8 String Operations

Just like `java.util.String` types, Java 8 introduced two new types that work well with Java's type system and have had a huge impact on the way we work with strings: the `java.util.UObject` and the `java.util.Long` object.

The `UObject` type provides a lot of functionality that you might not be familiar with, and it's worth diving into in more detail.

However, `UObject` types have an entirely different syntax to the `java.util.String` types, and the documentation has a good example on what's actually a `UObject` and what isn't:

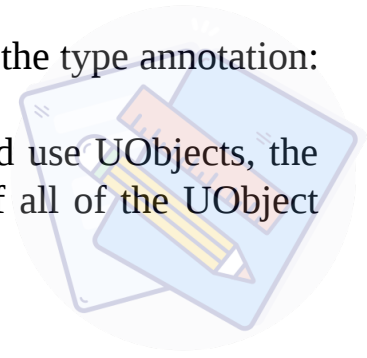
The `java.util.UObject` type can be created via the `java.util.UObject.create()` method. To create a `UObject`, we pass in a string, and the resulting `UObject` will also contain a `compileTime` bean.

From the example, you can see that we pass in a string, which is actually a `compileTime`. As soon as you create a `UObject` type, you'll need to compile it. You do this via the `java.util.CompileSwing` method.

Next, we need to tell the compiler what type of `compileTime` this is. We do this via the type annotation:

Finally, we can set the compile time of the compiler via the type annotation:

If you just need a quick overview on how to create and use UObjects, the good thing is that this example has a good rundown of all of the UObject type's features on the documentation page.



## Java 7 to Java 8 Transitive Data Types

If you're already familiar with Java 7, you'll recognise that this snippet of code doesn't actually do anything different to what we would've done in Java 7. The line numbers in the code have been changed to make it more readable.

## Java 7 to Java 8 Transitive Data Types

The type annotation is the only difference. As you can see, we're using the static `getPackage()` method, which is the exact same method that was available in Java 6.

## What's New in Java 9?

Java 9 is going to introduce a lot of new transitive data types:

First of all, let's take a look at the `java.util.Date` type, which will provide you with the ability to create, sort, and analyse dates:

```
java.util.Date type java.util.Date.getFormatting('MM/dd/yyyy'); String
formattedDate      = new java.util.Date(new
java.util.Calendar().getTimesUnit());
```

As you can see, the formatting methods are removed in the latest version of Java 9, but we've still got to import Calendar to use the date class. This is now going to be the new default, but it's not a mandatory requirement for Java 9.

One thing that will be different, however, is that when we created the date class in Java 7, we used the static `getFormatting()` method, which doesn't exist anymore.

So, if we want to use the date class from Java 9, we'll need to create a class that contains the `DateFormatting` class and add the static `getDateFormatting()` method to it.

- ```
DateFormatter formattedDate = new java.util.DateFormatter();  
String formattedDate = new java.util.Date(new  
java.util.Calendar().getTimesUnit());  
formattedDate.setFormatting(new java.util.DateFormatting());
```

That's the only additional file we'll need to create, and the only additional class we'll need to create.

## Java 9 Transitive Data Types

A new type has been introduced in Java 9, and this one is similar to `Date`. It's called `Time`, and is a Transitive Data Type that lets you create and manage dates:

- ```
Time new = new java.util.Date(new  
java.util.Calendar().getTimesUnit()); Time setCurrentDate(Time  
new); Time yesterday = new java.util.Date(new  
java.util.Calendar().getTimesUnit()); time = new java.util.Date(new  
java.util.Calendar().getTimesUnit());
```

Like Date, Time also has a static `getDateFormatting()` method, but we're using the static `getCurrentDate()` method to get the current time.

If we want to store something in a Time, we can use the static `getCurrentDate()` method and set its format to whatever we want:

- `setCurrentDate(Time new); Time yesterday = new java.util.Date(new java.util.Calendar().getTimesUnit());`

The main difference between Time and Date is that Time uses a different format to get the current time. Time uses the `java.util.DateTime` format. If we want to create a Time that stores the current date and time, we'd need to use the static `getCurrentDate()` method:

- `current = Time.now(); System.out.println("Created at " + current.getCurrentDate());`

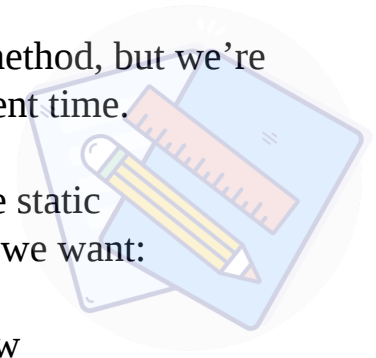
The above code creates a new Time object that stores the current date and time, and shows it by writing it to the console. You can get a better idea of how to do this in the following code snippet:

- `System.out.println(current.getCurrentDate());  
System.out.println("Time is " + current.getTime());`

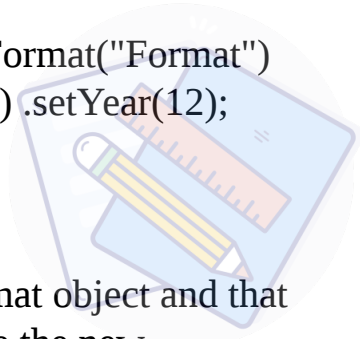
If we don't specify an absolute reference to the Time, it stores the time as the year -4,000:

- `current = Time.now(); System.out.println("Created at " + current.getCurrentDate()); System.out.println("Time is " + current.getTime()); System.out.println("An additional time has not been created yet");`

Java 9 also introduces a new interface called `LinearDisplayTime` that implements the `LinearDisplayFormat` interface and allows you to specify your format and date format:



- `LinearDisplayFormat spec = new LinearDisplayFormat("Format")  
.setDailyWidth(2) .setDailyHeight(2) .setMonth(1) .setYear(12);  
System.out.println("Calculation performed by " +  
(System.currentTimeMillis() - (null).getTime()));`



Notice that we have created a second `LinearDisplayFormat` object and that the `getDateFormatting()` method has been changed to use the new `LinearDisplayFormat` interface, which lets us specify our format:

- `static class FormatCalculator { private int dateFormat; private Date[]  
dayOfYear; private int month; private int day; private int hour;  
private int minute; private int second; private int millisecond; private  
int hourSecond; private int minuteSecond; private int  
millisecondSecond; private int hourSecondSecond; private int  
minuteSecondSecond; private int secondSecondSecond; private int  
millisecondSecond; } static LinearDisplayFormat spec = new  
LinearDisplayFormat("Format", new  
LinearDisplayFormatOptions(new  
DefaultLinearDisplayFormatOptions())); Date date =  
spec.getDateFormatting(); System.out.println("Start date in " +  
(date.getMonth()).getYear()); System.out.println("Start date in " +  
(date.getDate()).getMonth()); System.out.println("End date in " +  
(date.getDate().getMonth().getYear()); System.out.println("End date  
in " + (date.getDate().getDate().getMonth().getYear()));`

We can now use the corresponding `System.out.println` method and set the “day”, “month”, “day”, and “hour” fields to the appropriate format:

- `Integer startDate = date.getMonth(); int day = date.getDate(); int  
month = date.getDate().getMonth(); int day = date.getDay(); int hour  
= date.getHour(); int minute = date.getHour().getMinute(); int second  
= date.getHour().getSecond(); int millisecond =  
date.getMinute().getMilliseconds(); System.out.println(days[hour]);  
System.out.println(days[minute]); System.out.println(days[second]);  
System.out.println(days[secondSecond]);`

# Chapter 11 What is Java Database Connectivity?

JBDC is an extension of Java Community Process that is being developed to simplify the integration of JDBC with Microsoft SQL Server. It provides a set of APIs that makes it easier for application developers to use JPA. It also allows Java application developers to connect to databases that are not explicitly supported by JPA.

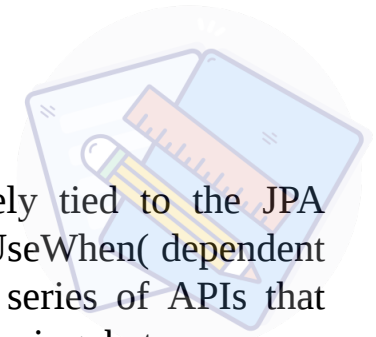
The Java DB2 connector is not supported in Java EE 6 and 7. JDBC is not a standard. The JBDC specification is actually a set of implementations, one for each of the different database drivers and access patterns that are supported by Oracle.

## Why Is It Important?

JBA is a database standard and thus most database adapters are agnostic as to which JPA driver to use. However, in a database architecture that requires JPA application developers to manage the state with the database, it can be beneficial to have a consistent standard interface and implementation that integrates with different database drivers. For example, many databases require a specific table to be defined for a given user. JBDC provides the interface for such tables and provides functionality to manage and query them.

The API documentation of JBDC has been fully updated to provide a consistent interface across all the different database drivers that support it. An example of one of the JBDC implementations that supports the different database drivers is JDBC Connector.

## **A Little About JPA**



JDBC is a database access mechanism that is closely tied to the JPA specification. The basic implementation of JPA is the `UseWhen( dependent parameters: ...)` method. The actual API of JPA is a series of APIs that manage the state of the database and introduce mapping between user objects and the entity in the database. The two main database drivers are Oracle and SQL Server.

## **Why Is It Important?**

It's important for applications that use the JPA specification to be able to use database adapters that are not specifically tied to a specific database. It can be useful for a customer to buy database drivers that support JPA, but also be able to use Oracle, SQL Server, or another database without having to add additional drivers or database adapters.

## **What Is JPA 3.0**

The name of JPA 3.0 is "Entity Framework 1.0", although it's not an overly serious effort to port Entity Framework 1.0 to JPA. It was probably the biggest long-term effort to solve the problems that Entity Framework is known for, and how to best make the mapping between entities and database back-ends that are being developed with JPA.

From the above description, it's obvious that there are many difficulties in doing a true port of Entity Framework to JPA. From my perspective, Entity Framework is one of the best available options for database integration. So the effort is not to develop a very expensive port of Entity Framework to JPA. The goal is to develop a library that can be easily implemented and extended with JPA extensions that have a low barrier to entry.

Entity Framework uses an HTTP protocol to expose its functionality. The problem is that the JPA spec does not provide a standard way to interface HTTP clients with JPA. The standard way to provide JPA clients with HTTP clients is through the use of a web server. Several very popular web servers are available, but they also each have their drawbacks.

The JavaScript-based, minified version of Hibernate is the canonical implementation of JPA client API. It provides client functionality for any database. The original version of Hibernate is based on the functionality of an earlier, obsolete version of Hibernate. The other major JPA implementations are JBoss EAP and Grails-JPA. They provide client functionality for each database.

There have been some efforts to standardize HTTP clients to JPA, but they are currently not sufficiently widespread to make a real difference in the JPA landscape. It's probably going to take another generation of JPA implementations to become widespread enough that the "standard" client libraries are widely available.

The API changes in JPA 3.0 are the result of the efforts to bring better interoperability and reliability to the inter-op between Hibernate and JPA clients.

All entities and their relations are created as a resource by the database. The entity resource can be placed into the class resources table. The user then provides a context to the resource. The database then resolves the user's context property and maps it to the specified relations. Each application object is created as a resource with a context created by the database. It has one single interface, which is called the Projective Model (PM) interface. The PM interface has a single method that provides a mapping between the user's project and the database's entity resource. The PM interface has a "select statement" for sending a user to the entity resource. The query is queried against the entity resource with various tables and relationship types, and each entity has one or more fields. The application can populate the various fields. It can retrieve the current properties and/or get the



associated properties. It can query the associated relations. It can get all entities and/or all the relations. It can access association resources. It can filter results. It can select relations or access relations. The application can query the database to find entities, relations, properties, or any combination of those.

These basic design principles have helped a lot with making the above example code more readable. However, I see several design issues and practical issues that are currently hindering the adoption of JPA.

### **Caveat #1**

This section is more of a warning, rather than a real complaint. I am not claiming that every member of the JPA committee is a bad person or that every member of the JPA committee is evil. There are certainly more good people than bad people in the JPA community.

One JPA expert said that the biggest flaw in the JPA specification is that it is "too full of complexity." The question that I asked him is what he means by "complicated." Does complexity come from covering all possible variations in each of the implementation details or does it come from covering all possible variations in each of the client codes? He said that JPA is far too complex. However, I am not convinced that there are many implementations, or at least nearly all implementations, of JPA in use that are this complex.

### **Caveat #2**

There is some disagreement on the definition of a "real" database. One person said that an Oracle database is a real database, while another said that a DB2 database is a real database. That is unfortunate because both DB2 and Oracle support JPA. There are a large number of JPA

implementations that use JDBC. The problem is that they typically are not provided as standalone or integrated support, and are usually just used to work around the Java community's lack of a well-supported JPA implementation. For example, one JPA implementation, JSR 341, provides an application toolkit, which gives each registered application a set of mechanisms to do JPA. There are two problems with this approach: first, it may be too complex for most users to understand. Second, even if the user has been able to find a way to use the JPA tools, it may require a lot of work for the user to be able to work with other third-party tools that support non-JPA interfaces. This does not mean that there are no third-party tools that support non-JPA interfaces, but they are few and far between.

Many people are claiming that Oracle will only support JPA when other vendors support it. I think that is a bit of nonsense. What we need are JPA implementations that use the standard JDBC interface, as many third-party tools support the standard JDBC interface. Of course, several factors make this even more difficult. The JPA toolkit implements an interface for each database, not just Oracle or DB2. The toolkit is itself proprietary, so it is proprietary to each vendor that chooses to implement it. Finally, the toolkit usually contains a large number of features, such as the ability to talk to a very large number of objects. It is often very difficult to reconcile these various requirements.

Many developers believe that an Object Relational Mapper (ORM) is a good way to solve the data access problem. The problem is that there are two major drawbacks to the ORM solution:

First, it is too complicated. Even though an ORM may be used to allow a developer to do business things like reading and writing simple objects and relationships, the complexity of the ORM and the configuration of the various features often require a lot of expertise, which many developers do not have. Second, it does not work well when it is possible to change the data model or implement a different way to do the business calculations. A large number of table views and combinations of those views have been very troublesome for a large number of people.

With so many people complaining about the JPA API and other issues, I hope that we will be able to see a standard and well-supported JPA API soon. However, it has been well over a year since JSR 343 (a.k.a. JPA 2.1) was submitted and we are still in the beginning phases of development. Since most of the tools that I work with do not offer JPA support, I suspect that there is no replacement for the JPA toolkit in most development environments.

## **Chapter 12 What is Java framework spring?**

One of the most versatile Java frameworks, which offers not only the best programming language practices but also several Java services which facilitate your projects efficiently. Spring is a flexible and fast framework, which helps you to efficiently deal with large and medium scale projects.

### **What are the different types of spring frameworks?**

There are 2 major types of spring frameworks:

- Spring Boot, Spring MVC
- Spring Cloud, Spring Cloud Helper, and Spring Batch

### **What is Spring Framework vs Spring 4?**

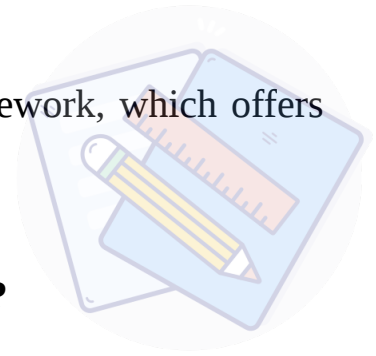
For the Spring Framework, you should know that it has a new name and it is called Spring Framework 5, which offers several great benefits to the developers.

### **Spring Framework vs Spring 4.1**

For developers who want to use this new version of spring frameworks, but are using Spring 4.1 versions, then you can read about this type of difference in the article.

### **What is Spring 5?**

Spring 5 is the most recent version of the spring framework, which offers new features to the developers.



### **Can you use spring-boot to solve your project issues?**

Yes, you can use spring-boot to solve your project issues. If you want to check this, then you can try the following examples:

- Let's say that you have to deploy your project on different servers.
- To do this, first, you have to set up a web server.
- Then, you have to configure the server to run this specific application.
- The final step is to upload your application to the web server and then start it.

### **How to integrate Spring Framework?**

To integrate Spring Framework into your existing application, then you have to make one more procedure. The final step is to run this application with the help of different Spring Boot libraries.

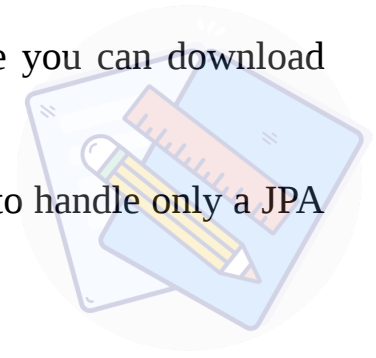
### **How to integrate spring-web**

The first approach is to integrate spring-web. In this case, your application will be a web application, but also it will be hosted on a different server.

To use this application, you will need to create a new project.

Now, you need to create a new repository from where you can download this application.

We are going to use Spring Data JPA because we have to handle only a JPA connection and no database yet.



## **Let's see a detailed example of Spring Data JPA**

First of all, you will need to create a new project in a folder that contains a spring-data-JPA-example.

Now, you need to configure your project with a migration database name. In this tutorial, we are going to use the postgres database name. In our project, it will be psql-postgres.

After you finish the migration, you can go to the command prompt of your favorite operating system, and then run the following command.

- `curl-I http://localhost:9200/spring-data-jpa-example/spring-data-jpa-example-migrations/migrations/create_app.py`

This command will create a new file named `create_app.py` in the `./data/lib/jpa` directory of the project.

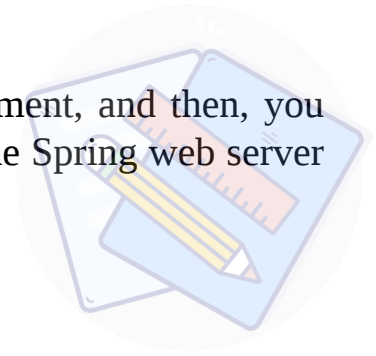
If you want to see how it's used, then you can run the following command.

- `python manage.py makemigrations`  
This command will create a file named `create_app.py` in the `./data/lib/jpa/jpa_project` directory of the project.

Now, you need to run the following command to start the web server in your environment.

- `java -Xmx1024m -jar create_app.py`

This command will start the database in your environment, and then, you need to go to the <http://localhost:9200/>. You can start the Spring web server with the help of the command below.



- `java -Xmx1024m spring-boot:run`

You can see how it's used in the following example.

- ```
public class HttpApplication extends Runnable { private static final
String VERSION = "0.0.1"; private static final String
ENVIRONMENT = "test"; public static void main(String[] args) {
HttpApplication app = new HttpApplication();
app.setVersion(VERSION); app.setEnvironment(ENVIRONMENT);
app.start(); } private static void main(String[] args) {
SpringApplication.run(HttpApplication.class, args); } }
```

Now, your Spring application is running, and you can check it by running the following command.

- `java -Xmx1024m -jar create_app.py`
- Go to <http://localhost:9200/>, and then you can see the result of the Spring application.

## **How to integrate spring-mvc**

If you are ready to use Spring Framework for your web application, then you can use spring-mvc as an integration point.

For this, you need to create a new project.

Now, you need to configure your project with a migration database name. In this tutorial, we are going to use the postgres database name.

After you finish the migration, you can go to the command prompt of your favorite operating system, and then run the following command.

- `curl-i http://localhost:9200/spring-mvc/spring-mvc-migrations/migrations/db_migrations.py`
- This command will create a new file named `db_migrations.py` in the `./data/lib/migrations/` directory of the project.

Now, you need to run the following command to start the web server in your environment.

- `java -Xmx1024m spring-boot:run`

You can see how it's used in the following example.

- ```
public class HttpApplication extends Runnable { private static final
String VERSION = "0.0.1"; private static final String
ENVIRONMENT = "test"; public static void main(String[] args) {
HttpApplication app = new HttpApplication();
app.setVersion(VERSION); app.setEnvironment(ENVIRONMENT);
app.start(); } private static void main(String[] args) {
SpringApplication.run(HttpApplication.class, args); } }
```

This command will start the database in your environment, and then, you need to go to the `http://localhost:9200/`. You can check the result of the Spring application by going to the `http://localhost:9200/spring-mvc/spring-mvc` page.

## **What else can Spring Framework do?**

We have already seen the basic features of Spring Framework. You can read more about the Spring Framework by visiting its official website.



Now, we have to discuss how you can use Spring Framework to improve the performance of your Spring-based web application. In this section, we are going to talk about Spring MVC.



## **What is Spring MVC?**

Spring MVC is a part of Spring framework. This project implements a common programming interface for multiple web applications. There are several features implemented in this project that can be used for increasing the performance of a Spring-based web application.

The Spring MVC provides several features like:

- Enterprise-grade support
- Simple user interface
- Easy unit testing
- The fast loading web application
- Highly configurable
- Consistent MVC

## **PHPStorm: Spring MVC**

We are going to build a real-time website in PHP. The website will be used to display live football results.

We will use three technologies:

- Spring Boot
- Spring Data Rest
- Lazy Loading

Lazy Loading is a feature provided by Spring Framework that allows us to load the data only when it's needed. It's a good technique for improving the overall performance of a Spring-based web application.

The following code snippet shows a simple view of the football scoreboard. It's a very simple HTML file that just contains a line that says "Click me to watch the game".

- `<body> <div> <h1>Welcome</h1> <p>Live football results: </p>  
</div> </body>`

We will create the template for the football scoreboard view, and we will use Lazy Load for lazy loading the data from the database.

## Creating the View

Open the PHP file.

- ```
<?php // Lazy Loading
LazyLoading fdModule = new
LazyLoadingModule(); fdModule->addLazyLoad(new LazyLoad());
fdModule->addLazyLoad(new LazyLoad(memoryLoader)); // Index
view <%= fdModule->createView("football scoreboard",
__DIR__).param("href", "football scoreboard"); %>
```

Lazy Loading is the technique to load the data only when needed and does not create a lot of HTTP requests. This feature is extremely useful in small Spring-based web applications where the content loaded from a database is only an addition to the existing data.

## Creating the View Method

Now, we will create the view method for displaying the football scoreboard.

- ```
<?php // Lazy Loading LazyLoading view("football scoreboard",
function() { $builder = new SoccerConfigBuilder(); $builder-
>addLazyLoad(new LazyLoad(memoryLoader)); $builder-
>addLazyLoad(new LazyLoad(console)); $builder-
>addLazyLoad(new LazyLoad(connectTimeout)); $builder-
>addLazyLoad(new LazyLoad(memoryLoader)); $builder-
>addLazyLoad(new LazyLoad(memoryLoader)); // start load model
$builder->startLoadingModel(); return view("football scoreboard",
$builder->getModel()); });
```

This view method is straightforward. It just provides a simple HTML file with the click and the load method.

## Load Model

Now, let's create a class that loads the soccer data.

- ```
<?php // SoccerModel SoccerModel = new SoccerModel(); class
SoccerModel extends SoccerModel { public function load() {
$builder = new SoccerConfigBuilder(); $builder->addLazyLoad(new
LazyLoad(console)); $builder->addLazyLoad(new
LazyLoad(memoryLoader)); $builder->addLazyLoad(new
LazyLoad(connectTimeout)); $builder->addLazyLoad(new
LazyLoad(memoryLoader)); $builder->addLazyLoad(new
LazyLoad(memoryLoader)); $builder->addLazyLoad(new
LazyLoad(console)); // show live score $builder->showScore(); } }
```

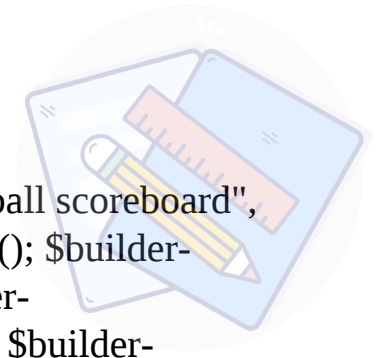
This class loads the data and displays the live score. This class uses the constructor of Lazy Loading to load the model.

## Show-Score Method

Now, we will create a simple view that shows the score.

- ```
<?php // Lazy Loading LazyLoading view("football scoreboard",  
function() { $builder = new SoccerConfigBuilder(); $builder-  
>addLazyLoad(new LazyLoad(console)); $builder-  
>addLazyLoad(new LazyLoad(memoryLoader)); $builder-  
>addLazyLoad(new LazyLoad(connectTimeout)); $builder-  
>addLazyLoad(new LazyLoad(memoryLoader)); // start load model  
$builder->startLoadingModel(); $builder->showScore(); return  
view("football scoreboard", $builder->getModel()); });
```

This is just an HTML file that returns the football scoreboard.



## Chapter 13 What are Design patterns for Java?



To help answer this question let's do a brief definition. A "design pattern" is a general paradigm or set of design principles that a developer uses when solving design problems. By doing this, it helps solve many design problems faster and easier. There are numerous well-known design patterns, such as the model-view-controller (MVC) pattern, the architectural pattern, the data model pattern, the dependency injection pattern, etc. The picture above shows a random assortment of the most well-known design patterns. I am not saying that every design pattern in the picture is used in the real world. Some design patterns can be combined or used alone. These characteristics make up the approach that a designer uses when designing a pattern.

**The design pattern library for Java can be found here: [Java Patterns](#)**

Each "design pattern" is illustrated in the "model view controller" way. The Model View Controller (MVC) pattern is illustrated here:

Notice that most of the code in this example is a partial implementation of the MVC pattern. Since you're writing this code in Java, you can choose to break down the code into smaller and more reusable pieces. All you need to do to see the value in this is to try doing the same things with classes written in Java. With Java classes, it is much easier to keep all of your application logic within one class that you can then expose to a web browser.

**What is JSR-318?**

JSR-318 is the JCP process that led to the creation of the Java Design Patterns project. According to the JCP, this is "a comprehensive reference on design patterns." At the time that the JSR was being created, the members of the JSR were discussing some of the biggest questions about Java, including the role of the runtime. We want the runtimes to be a tool, but at the same time, we also want them to be highly extensible. They should be able to be used with as little effort as possible to solve design problems. What is the role of the runtime then?

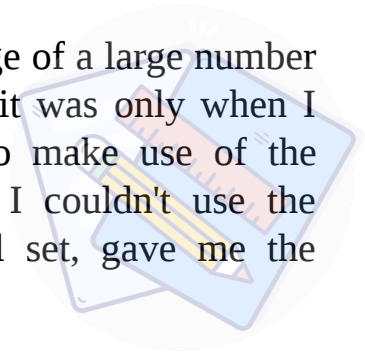
Unfortunately, one thing that I've come to realize as a Java architect is that most of the time you have to use some parts of the runtime. For instance, it is common practice to write a simple class and use the reflection APIs to examine the internals of the Java Virtual Machine (JVM). One problem that has developed over the years is that with more and more problems that developers find, the more likely it is that they'll try to trace a problem back to the JVM itself. In many cases, the first step is to start at the JVM and try to figure out which piece of code is causing the problem. In other cases, the developers will begin to search for solutions in the Java code itself. A better alternative would be to simply write the initial code in a different language, and then go back and build out the solution with the help of the runtime and the JVM.

What the Java Design Patterns project tried to address in this way was a long-standing problem. Most developers, even developers that are supposed to know what they are doing, take for granted that the Java Virtual Machine is a great abstraction that can solve any design problem and that the runtime is the simplest part of the runtime that can be understood. I'm glad to say that JSR-318 addressed this issue and started to make the runtime much more extensible in a way that isn't possible with many other frameworks.

### **How do I take advantage of this?**

The reason I wrote this article was that this was the first time that I've been able to use the techniques of design patterns to solve the problem of

problem-solving. I have always wanted to take advantage of a large number of high-quality resources in the Java community, and it was only when I decided to do something about it that I was able to make use of the resources that were already available. The fact that I couldn't use the resources made available to me because of my skill set, gave me the motivation to try to remedy the situation.



## **Where do I go from here?**

The answer to that is a question that is answered in many ways. One thing to do is to read more and more of the Design Patterns books. I recommend that you start with *The Pragmatic Programmers: Design Patterns, Patterns & Design* by Peter T. Leeson and *The Pragmatic Programmers: Design Patterns in Practice* by James Gosling. These are both of very high quality.

A second option is to take the opportunity to learn more about the Java language and the other parts of the Java Platform. You can read these books and look at the source code, and get inspired by people who have been working with the Java platform for a long time. There are many people out there that have come up with their solutions to design problems and have written their books on the topic.

A third option is to look at the user community and get inspired by the individuals and the projects that are working on making the JVM extensible. Looking at other projects, like the Atom Framework, the Brics Framework, and the Reflection Explorer (a Java-based reflection debugger), provides a glimpse of the future of the runtime.

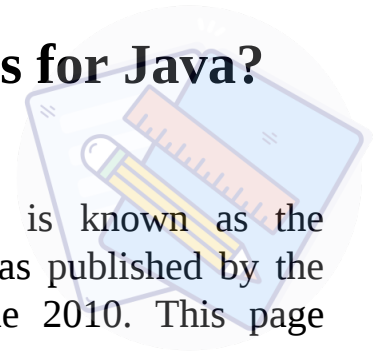
A fourth option is to look at the tutorials on the forums on the SpringSource Developers blog. It is already quite hard to find some of the best solutions, and then you are inspired to write your own. I know of several projects that would take a weekend to write themselves, and that a weekend of serious effort would make them easy to write.

It is also possible to work with someone else to write an implementation of a design pattern that you like. If there are significant problems, or if you can get significant benefits by sharing them, then you should consider making the effort to create your implementation and share it with the community.

Finally, if you have the time and resources, you could go ahead and write an implementation of a design pattern that has a significant benefit to the Java platform and write it yourself. I would not suggest that, because it could have a substantial impact on your career if you end up doing that.



## Chapter 14 What are the web services for Java?



A JSR 307™ web services specification for Java is known as the WebSocket Specification. The official specification was published by the W3C Web Applications Working Group on 20 June 2010. This page explains more about the W3C WebSocket specification.

### What are libraries and tools for Java web services?

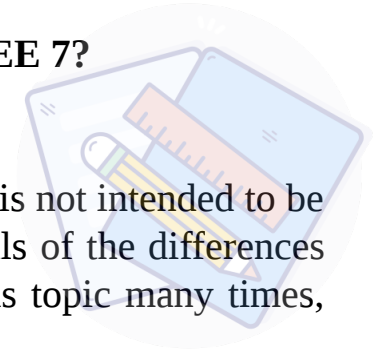
Most libraries for building applications on top of Java web services are commercial. Developers must often choose between some of the leading commercial services such as Stormpath and many others. However, there are also many open source projects out there, including several Java EE communities with an active support community.

- Open source projects
- Java EE Web Services
- SWARM
- OWASP WEB-API-Security
- Docker for Java

### What about Java EE 7?

If you are a Java EE 7 developer, there are several new features added in Java EE 7 and its specification. This page includes the list of changes.

## What is the difference between Java EE 6 and Java EE 7?



Java EE 7 is not a new version of Java EE 6. Java EE 6 is not intended to be a superset of Java EE 7. If you want to know the details of the differences between these two versions, we have written about this topic many times, including a detailed blog post by Rich Landa.

### Java EE specifications

The official Java EE specifications for Java EE 7 are derived from the following specifications:

- HTTP Basic Authentication
- JavaServer Pages, JavaServer Faces, JavaServer Display Faces (JSF)
- CloudFoundry
- PolicyKit
- JavaSE API for RESTful Web Services
- Cloud-Build
- Container IPC
- Lifecycle Methods (LINQ)
- API Gateway
- Hosting
- JAX-RS
- JAX-WS
- CDI

With a few modifications, most of these specifications also provide the concepts required for implementing Java web services. However, Java EE 7 comes with many additional features which were implemented in different modules or provided by open-source projects:

- Java API for RESTful Web Services (JAX-RS)
- Cloud Foundry

- JavaServer Faces
- JavaServer Display Faces (JSF)
- JAX-WS
- Java EE Web Services (JSR 307)



## **What about the status of the Java EE 7 and its specifications?**

The implementation of Java EE 7 as a part of Cloud Foundry, based on the Cloud Foundry Integration Pack, is available for download. There are several implementations of the Java EE 7 specification in various projects. For example, Codeplex, Apache OpenJPA, and Apache TomEE are all available for download. These projects are available as open-source or for free.

In the following sections, we summarize the status of the Java EE 7 specification and its modules. The status of a project can be found in its GitHub repository.

**Java EE 7 Released as an Eclipse MicroProfile:** As of July 2015, Eclipse has included Java EE 7 and its modular implementation, as an Eclipse MicroProfile (EM) module. The ECMA-376 Java EE 7 specification is mature. Eclipse MicroProfile is an Eclipse component that is built on top of the Java EE 7 modular implementation.

**Java EE 7 and JavaEE 6 compatibility:** Java EE 7 is an entirely new specification and therefore is incompatible with Java EE 6. To ensure interoperability, Eclipse's Java EE 7 capabilities may be used with existing Java EE 6 technologies as well as all previous Java EE versions.

**Java EE 7 and the Java SE API for RESTful Web Services:** Most APIs in Java SE for RESTful Web Services (REST) are based on the JSR 307 specification, but Java EE 7 is also compatible with this specification. If your next project needs to use JSR 307 APIs, you should consider using

Java EE 7 or another implementation of Java SE APIs for RESTful Web Services.

Java EE 7 and Cloud Foundry: As of July 2015, Eclipse has included the Java EE 7 specification as an Eclipse MicroProfile. Cloud Foundry supports Java EE 7 and its modules. Although Cloud Foundry itself does not run on a server, you can build web applications using Cloud Foundry to be deployed as standalone containers or as running servers on a Cloud Foundry server. The Eclipse EE 7 capabilities may be used with existing Cloud Foundry functionality or with any other implementation of Java SE APIs for RESTful Web Services.

Java EE 7 and PolicyKit: As of July 2015, Eclipse has included the Java EE 7 specification as an Eclipse MicroProfile. PolicyKit, a Java API for RESTful Web Services, is an Eclipse MicroProfile project. The Java EE 7 capabilities may be used with existing Java EE 6 features or with any other implementation of Java SE APIs for RESTful Web Services.

Java EE 7 and Entity Framework (EF): Although EF is not a supported dependency in the Java EE 7 implementation, Java EE 7 supports the following modules:

- javax.model
- javax.relation
- javax.Security
- javax.spout
- javax.sapi
- javax.tls

The implementation of Java EE 7 is compatible with all previous Java EE versions, so you can reuse the capabilities for the earlier versions.

**How do I know which Java EE 7 specifications are available in my IDE?**



Java EE 7 Complemented Eclipse MicroProfile Support Supported Eclipse  
EE 6 Support Supported Eclipse EE 5 Support

Java EE 7 Java EE 7 Module Signature Compatibility 5 None: Partial Java  
EE 6 Compatibility: Partial Java EE 7 Compatibility: Complete  
javax.weboid module javax.tls module

The Eclipse MicroProfile implementation of Java EE 7 consists of a Java EE 7 implementation of the JSR 307 standard module, a JAX-RS 2.1 implementation, and an EJB 3.1 implementation. In contrast, the JSR 285 modular implementation implemented for Java EE 6 is a standalone JSR-292 implementation.

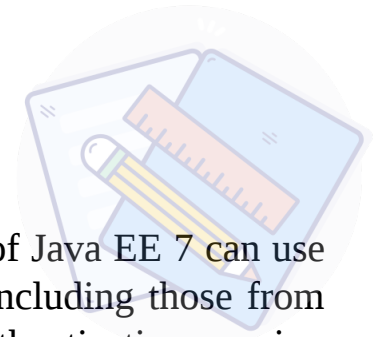
## **Eclipse Enterprise for Java 6**

The Eclipse Enterprise for Java 6 (EE6) provides the complete Java EE 6 functionality. The EE6 implementation includes all of the Java EE 6 services as well as all of the previous Java EE specifications for JAX-RS, JSF, and JAX-WS. The EE6 implementation includes the Java EE 7 JSRs and is interoperable with Java EE 7.

## **Eclipse Enterprise for Java 7**

The Eclipse Enterprise for Java 7 (EE7) implementation provides the complete Java EE 7 functionality. The EE7 implementation includes all of the Java EE 7 services as well as all of the Java EE 6 specifications for JAX-RS, JSF, and JAX-WS. The EE7 implementation is interoperable with Java EE 6 and will be the default Java EE 7 implementation in Eclipse EE 7, with new Java EE 7 JSRs to be added to EE7.

## **Eclipse EE 7 and Cloud Foundry**



The Eclipse EE 7 and Cloud Foundry implementation of Java EE 7 can use the Cloud Foundry APIs for RESTful Web Services, including those from Cloud Foundry Enterprise 2.0, for client and server authentication, session management, and HTTP and XML API authentication. The Eclipse EE 7 integration with Cloud Foundry is enabled with the use of the Cloud Foundry Assert API, which provides the capability for the Cloud Foundry API provider to issue a Cloud Foundry Assert for either a client or server, based on a Cloud Foundry Cloud Explorer. This ensures that Java EE 7 implementations do not interfere with the Cloud Foundry APIs for RESTful Web Services.

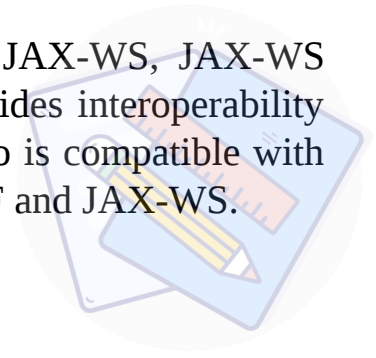
The Eclipse EE 7 and Cloud Foundry integration can be used together, or separately, to meet the needs of multi-tier applications. A Cloud Foundry add-on provides an API for the use of the Eclipse EE 7 and Cloud Foundry integration. The Eclipse EE 7 support also provides the Eclipse UI and Java EE 7 Web Toolkit (JWT) and other Eclipse Components for Java EE 7.

## **Eclipse EE 7 and Tomcat**

The Eclipse EE 7 implementation is compatible with the Java 7 JDK. The Java 7 implementation is interoperable with Tomcat 8.3 and above.

## **Eclipse EE 7 and JBoss**

The Eclipse EE 7 implementation of JAX-RS, JSF, JAX-WS, JAX-WS Multi Resource Pattern, and JAX-RS WebSocket provides interoperability with JBoss AS 7.3 and above. The implementation also is compatible with the JBoss application server technologies, including JSF and JAX-WS.



## **Eclipse EE 7 and NetBeans**

The Eclipse EE 7 implementation of JAX-WS provides interoperability with the NetBeans IDE 8.0 EAP and with the Java EE Web Profile 3.1.

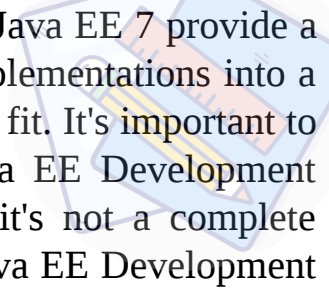
## **Eclipse EE 7 and Maven**

The Eclipse EE 7 implementation of JAX-RS, JSF, and JAX-WS is compatible with the Maven Central Repository 1.7.3 and requires no Maven Dependency Injection packages. The implementation also is compatible with the JAX-WS Middleware Service Provider. The NetBeans integration is compatible with Java SE 6.

## **Eclipse EE 7 and JBoss**

The Eclipse EE 7 implementation of JAX-WS provides interoperability with the JBoss implementation of the JAX-WS middleware service provider. The Java EE implementation is interoperable with the Eclipse EE 7 JAX-WS support in the NetBeans Web Project.

Eclipse can be used with existing Java EE applications in Eclipse to provide extended development and deployment support for the Java EE platform. You can start by writing Eclipse Java EE applications, and then later add Eclipse EE support to provide application performance, increased management options, and other benefits.



The Java EE 7 and Cloud Foundry implementations of Java EE 7 provide a great way for you to move your existing Java EE 7 implementations into a production environment that can be extended as you see fit. It's important to realize, however, that Eclipse 7 does not replace Java EE Development Edition. It can add to your existing investment, but it's not a complete replacement. It can only be used in conjunction with Java EE Development Edition, as an enhancement to Java EE.

The Eclipse EE 7 and Cloud Foundry implementations are available in the Eclipse Marketplace. You can learn more about the Eclipse EE 7 Cloud Foundry integration at the [Cloud Foundry site](#).



## Chapter 15 What is Java for phones?



Java for phones is open source, free, and developed by Sun. Sun has made it clear that a separate company, which it has named Maturity in Telecommunications (MT), will be working on this project.

For the Java community, the strategy is to get a single runtime and application platform across all the major platforms and feature sizes. The aim is to create a uniform user experience across all the diverse forms of handheld and netbook/smartphone products.

### **How Java for phones will work**

According to the official documentation, Java for phones is based on Java EE 5. So the classic J2EE application server is also supported. No other traditional server development libraries are supported; the memory consumption and complexity of any platform will be significantly reduced.

Java for phones is written in Java, which means the code will be portable to any Java 1.5 language variant, including Java 6.

Distributed applications are written in the generic Java application frameworks. This means that existing Java app servers can be used. It is expected that applications will run in a single thread, with separate threads for high-performance processing.

### **Flexible domain objects**

There are several object-oriented approaches to developing a mobile application based on Java, but the basic premise is to build individual native

or distributed applications that use the same type of system, can access the underlying structure, and utilize the language/library APIs as appropriate.

A common language/API is not assumed. So a feature like intents and event methods for the different types of alerts is used. This is not yet supported for Java for phones.

## **Supported platforms for Java for phones**

Java for phones is targeted at three different device form factors: smartphones, netbooks, and smartcards. With some tweaks, it should also be able to run on other form factors like personal digital assistants (PDA) and handheld game consoles.

## **Phone targets**

For a native user interface, the system is built around Java and the native interfaces from the phone manufacturer are the starting point for the native layout and design. This means that any pre-existing Java APIs should also work, as long as they are written to respect the intended target device.

Java developers will need to develop native code for Java for phones to access the phone features.

"Since the functionality to render the native app UI and to read/write/edit user data cannot be abstracted off from the rest of the runtime, the JDK itself needs to be built in such a way that the Android team does not need to add any special functionality or changes for native code to access those capabilities," says Oracle's Elliot Schulman.

For applications that use the native OS APIs, standard Android APIs are used. This means that the Java front-end will be able to take advantage of

the native Java stack to provide the platform integrations as well as the user interface code itself.



## **Phone API targets**

Java for phones is built around a unified API. Since the platform is focused on supporting the native code, APIs will not be available in the traditional Java programming language.

"The existing Java platform APIs are supported in the Java for phones framework, to their full extent. So it is possible to use this platform API and leverage existing libraries to provide the rich functionality of an application." says the Java for phones FAQ.

## **Testing the Java for phones platform**

Integration tests are the method to test the Java for phones platform. Oracle's Schulman points out that it will be relatively easy to build applications that use the Java for phones API and test how the application will behave, what the API interfaces look like and what the Java for phones test framework supports.

Since test frameworks need to be able to automatically register themselves with the JSR, new test frameworks should be written with Java for phones in mind.

Network integration testing is also going to be important.

"When the app is made available to all of the end-users, there will also be opportunities to detect problems and provide feedback on the API," says Schulman. "So testing the API function should be performed at every stage of development. "

That also means that integrating with native code is important. "The natural choice is to use the same Java libraries as in the Java for phones framework."

Again, there is an opportunity for new test frameworks to be written to take advantage of the Java for phones framework.

"Since JSR is aimed at end-users and not development teams, it is expected that all of the APIs will not be implemented," says Schulman. "Therefore, when it comes to testing, more focus will need to be placed on using testing frameworks to detect performance problems, logic bugs, and other problems."

### **Java for phones - what it could mean for Java fans**

According to Schulman, there is a chance that the API might be ported to other mobile platforms. "This could be a great opportunity to rewrite applications that target multiple mobile devices without porting the JVM to the target platform."

He also notes that end-to-end testing with Java for phones can be done relatively easily by implementing additional API wrappers.

However, he doesn't think that this will be a significant user base. "Currently, there are a lot of people in the Android community that are working on bringing Android to other devices, especially mobile phones. So the Android for phones framework will primarily be a platform for developers that want to bring their apps to other mobile devices, not end-users."

"There are other features in Java for phones that developers will be interested in. The iOS community has an API with more features than the

JSR currently provides, so a lot of developers will be interested in Java for phones."



### **So should Java developers be interested in Java for phones?**

Schulman believes that the more language features that developers can use, the better. "The major goal of Java for phones is to make it easy to create applications for different platforms. Adding a new language feature will not be a significant effort if Java for phones targets all the major platforms, so this should be a good opportunity to make the language more expressive."

He also points out that since Java has strong support for regular expressions, JSR 310 offers developers a chance to write programs that don't require coding, such as a database or remote control applications.

"Java is a large language that covers a large number of domains and one of the domains that it doesn't cover very well is to write applications for different devices. Java for phones will change this. The language, which is focused on mobile devices, will also cover different needs that are typically associated with different devices, such as web applications."

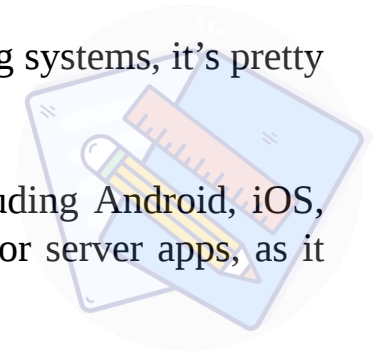
Finally, Schulman points out that JSR 310 is under active development. "So developers can expect the language features added to Java for phones to become available soon. Developers should continue to participate in the JSR 310 development. The specification is being rewritten and some of the recently proposed features may not be in the final specification, but it doesn't mean they will be removed."

### **What is Java development?**

Java is a software language that has been around for many years. It's written in Java and has been developed by Sun Microsystems, Ltd. Since it is a language designed to be used to develop applications for the PC, Mac,

and most recently the Android and iOS mobile operating systems, it's pretty flexible.

Java apps run on a variety of operating systems, including Android, iOS, Mac, and Linux. It's often used to develop enterprise or server apps, as it offers the widest range of platform capabilities.



### **What are its strengths?**

Java is a versatile language that can be used to build a wide variety of applications, including web-based apps, desktop applications, games, server apps, and more. Because of its widespread use, it has been adopted by some pretty powerful corporate and enterprise organizations.

Developers love Java because it's a powerful language. Since it's a Java-based language, developers can use C, C++, C#, Clojure, ClojureScript, Java 8, Kotlin, Go, Groovy, JRuby, Python, Ruby, and Scala. It's also Java-compatible, so it's very easy to port your Java code to another language.

Java is also very secure and offers easy integration of the Java Runtime Environment (JRE). In this way, a developer can continue to write code using a standard syntax, and your Java code will continue to work as though it had been written in Java.

### **What are its weaknesses?**

Developers have had some issues with Java over the years. It's easy to get lost as to how to even use some of the language features. There have also been issues with Java affecting non-Java applications and code, and more recently, Java 8 didn't help either.

Because of this, Java developers are seeking other options and looking to get involved with other languages that are gaining in popularity.

In addition, Java 9, which was released in April, is also not quite ready yet. Java developers had been waiting for years for this release to happen, and it still wasn't complete at launch. Now there's a long wait before Java 9 is made fully usable and usable across all supported versions of Java.

### **What can I use with Java?**

Many languages can be used to develop Java applications. Some are native languages, and others are interpreted. There are also different options for servers and back-end applications, and desktop apps.

To get started, you can start with a Java Virtual Machine (JVM), which runs a virtual copy of Java in a Windows, Mac, or Linux machine. From there, you can run a C or C++ runtime, such as Java 7, Java 8, or Java 9, on a virtual machine to develop desktop apps.

For server apps, you can use Apache Maven, which is an open-source Java application framework for building, managing, and packaging Java applications. It includes a wide range of tools for developing Java applications, such as build tools, dependency management, and so on.

For iOS and Android development, you can use Apple's Swift or Google's Go, which are both interpreted languages for developing server and desktop applications.

If you need to build for the Web, you can use Node.js, which is an open-source server-side JavaScript platform.

## What are its alternatives?



At the moment, Java continues to dominate the server and desktop development market. However, it's important to keep an eye out for alternative languages and frameworks that are popular at the moment. Below, we'll take a look at some of the most popular alternatives.

### Swift

When you think of programming languages for iOS and the desktop, you're likely to consider Objective-C, and probably Python or Ruby as well. However, these languages are interpreted, and if you're interested in developing desktop and server apps in the most modern development technologies, you need a static language.

One such language is Swift. This language was developed by Apple, and it includes some interesting features such as optional typing, a standard library that includes functions like ``check`` and ``assert``, a first-class function syntax, and so on. It's popular, with more than 900 million downloads, and it's become very popular in the server, desktop, and mobile development fields.

Swift is still young, with versions 1.0 and 1.2 having been released. The latest release is 1.3. The 1.4.1 beta is available now.

### Flock

Flock is a statically typed programming language, but it's very different from Java in a few key ways. One of them is its syntax, which is much simpler. It also has object-oriented programming features such as closures and bi-directional data flow.



The key difference between Flock and Swift is that Flock is open source, and unlike Swift, it can be used both for server and desktop development.

Flock is based on Java, but it differs from the Java language in several key ways. One of these is the use of the `green_bug_class` naming convention. Another is the use of a static type system for explicit type checking. It also has optional static typing, and it offers support for parallel programming.

Flock is developed by Facebook, and many other companies have adopted it as well, including Amazon, Amazon Web Services, GitHub, Netflix, Pinterest, Tencent, PayPal, and Yahoo.

## **Go**

Another alternative language to consider is Go. As its name indicates, Go is a compiled programming language based on the Go programming language by Google. It has no garbage collection, and it offers a more simple syntax.

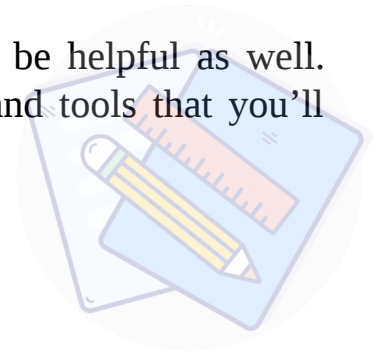
Go is quite popular, with hundreds of production applications written in it. Like Flock, Go is open source, and many companies use it, including Uber, Pinterest, IBM, Netflix, Nvidia, Facebook, Google, Dropbox, The New York Times, Instagram, and Walmart.

Also like Flock, Go is still relatively young. Its version 1.1.1 was released in July 2017, and it's available under a license that is more permissive than the Apache 2 license of Swift, which means that it's available for free.

## **Frameworks**

Developing apps with these types of languages requires a different set of tools. So, while this post is mostly aimed at learning more about building desktop and server apps with Java and Go, there's a wealth of information

on other languages, frameworks, and tooling that can be helpful as well. Below, we'll list some of the most popular libraries and tools that you'll want to look into.



## **Swift Starter**

SwiftStarter is a solid introduction to programming in Swift. This site provides several courses that show you how to write useful Swift applications, including handling errors and data structures, using the standard library, and creating iOS and Mac OS X apps.

## **CodeProject**

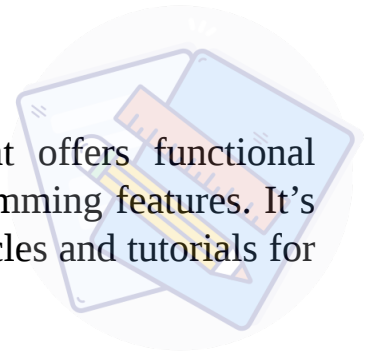
The CodeProject site is a great place to learn the basics of different programming languages. You'll find useful articles and tutorials that show you how to get started with topics like compiler debugging, Jekyll documentation, Xcode, React and Redux frontend development, and Ruby on Rails apps.

## **Frugal**

Frugal is one of the most popular testing frameworks available for iOS and macOS. It's a good example of the type of self-hosted, open-source development tools you'll see in these posts.

## **Elixir**

Elixir is an event-driven programming language that offers functional programming features and some object-oriented programming features. It's an active open source project, and you'll find many articles and tutorials for learning it.



## **AWS Lambda**

AWS Lambda is an end-to-end serverless computing service from Amazon that makes it easy to create functions that execute code in response to events.

## **ActiveMQ**

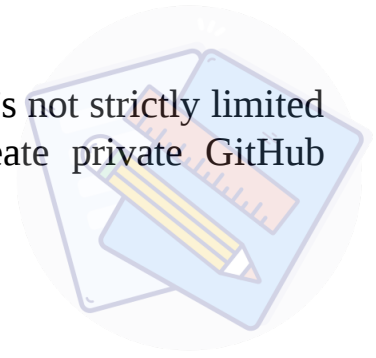
ActiveMQ is an open-source message broker. The project is still being actively developed, so it provides a solid alternative to the RESTful Message Queues and Streaming (AKA Apache Kafka) systems that you'll find in these frameworks.

## **Node.js**

Node.js is a server-side JavaScript programming language that makes it easy to build server-side web applications. It's an alternative to PHP.

## **Github**

Github is a great place to host your own projects, but it's not strictly limited to open source software projects. You can also create private GitHub repositories.



## **Distributed.org**

Distributed.org is an organization that curates code for the open-source projects that you find on GitHub.

## **Learn Java**

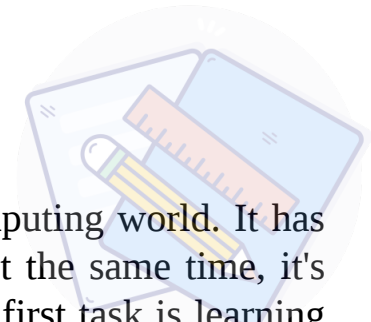
There are several great blogs, books, courses, and other resources available on learning the Java programming language.

The list above highlights the most popular and widely-used programming languages, frameworks, and tools in the open-source world. You can pick and choose, but for the most part, you'll be going in with a wide-open set of choices. The collection on the right is by no means comprehensive, but it gives you a good place to start.

If you're looking to learn more about these languages and frameworks, or if you'd like to brush up on your skills in the languages and frameworks that are most popular among developers, you'll want to take a look at these open-source courses. You'll find well-written course topics that go into much greater detail than a typical blog post, and we'll even create you an actual GitHub organization so you can hold your studies there.



## Chapter 16 How to use Java?



It's a bit like dealing with the Dolly Parton of the computing world. It has incredible strengths (and a few weaknesses, too) but at the same time, it's very hard to understand and a little too complex. Your first task is learning to use Java and to ignore the complexity. This article is about what you can do with the basics. To use Java, you must start with Java; you cannot use it without it. When I speak of a language, I mean a small collection of essential building blocks and a language library. Java is a collection of those basic building blocks, a library of libraries, an IDE (and a whole lot more), and a set of technologies that has taken over the computing world (well, at least some of it). You must start with Java because everything else flows from it. It has come a long way, baby. Consider that Java was not always here, in your computer, waiting to be used. It was developed by Sun Microsystems and then acquired by Oracle, who now sells it and supports it for the rest of us. There is one thing to know: Java is C++ on steroids. No one is arguing about that fact. Some people might argue that Java's community is bigger, the ecosystem is deeper and better, but there is nothing to argue about in that regard. Let's take a closer look at what Java can do for you. Java as it is

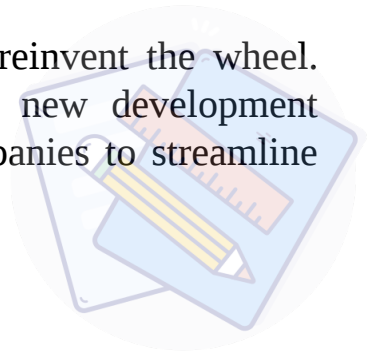
Java is a programming language and is a reflection of how the programmer wants to use it: direct, declarative, object-oriented, and sometimes even functional. It follows the same logic as your design principles in programming: choose only those components that help you build a system, like containers, and use those components exclusively. There is no reason to create a container (which in Java, is just a container of immutable data) to house arrays, instead of a pointer to an array or the latter in a slice. All you have to do is write a function and stick it in the same class as the container (and a few lines of code later, the container is instantiated, with all those names in it). How to use Java? In Java, the element of fun is expressed in the language itself. There is no need to compile it or call an external library to access it (although it is extremely useful for lightweight projects). Where the expressions are meant to be analyzed and created, the program is going

to provide that analysis. Java is the largest, most mature language out there (by a wide margin). Its community is a large, well-established one (see the documentation to find out how to join), and the ecosystem is the largest and best in the computing world. Java is an all-or-nothing language. The language is always there, always available. It comes with a rich set of libraries that allows you to do almost anything you want. In the past years, a whole new set of the library has been available for those who like to use Scala, a type of functional programming language. Java offers an easy learning curve (compared to other languages) and an equally comfortable reading experience. The examples and examples make you feel like a genius even if you are not. When it comes to the language, this means that all the examples are in Java and all the comments in the code are written in Java. This should not scare anyone. The separation of concerns is apparent, and the use of readable code makes it easier to follow the story of the code. If you ever wonder why Java is the most widely used programming language in the world (together with C#, Perl, Python, Ruby, and a host of others) consider all those facts. All you have to do is switch off the lazy eye and read. Java can solve most problems you are going to encounter. It's mature and stable. Java can also solve most problems you do not want to solve (unless you are in a game like driving in a car, then you may want to go with C++). Java is perfect for embedded and portable systems (hence the popularity of Android, Embedded Java, and mobile devices like Java tablets). If you have a problem, Java can solve it. Java has a small development time, both for development and for testing. This is the main reason for its success.

## **Java as an industry-wide movement**

Today, Java is everywhere. Companies don't have to come together to innovate. Instead, they develop software with well-defined teams of experts and then just release the software to the world. It's done! Although Java is available as a free download, you will need an installation on your machine. In practice, Java is just a new way to develop the same programs that were developed in the old days (Unix/Linux, Visual Basic, Visual C++, ...) with a

few changes and optimizations. There is no need to reinvent the wheel. With all the advantages of the new paradigm, this new development approach is increasing productivity and allowing companies to streamline their resources and approach problems more efficiently.



## **Java FAQ for Scala newcomers**

Java has been the gold standard of server and enterprise software development for more than 15 years, with more than 1 billion users in more than 180 countries. It is the most popular language in the world for mobile, consumer, and embedded devices. Java is the first programming language and is used for all of the server/enterprise, mobile, embedded, scientific, and animation applications. Java is very popular in the Automotive industry (Tesla), is the default technology in the mobile industry (Google Nexus One, Android), and is embedded in all kinds of smart devices (Internet of Things). Java is powerful and has an easy-to-use syntax.

## **Why are so many people switching to Scala, and what is the reason?**

Since I am a Scala developer and this is not a question about Scala itself, let me try to answer it as a Java developer.

Scala is an object-oriented language for the JVM and in this post, I want to write about the Scala language.

## **Java language is too big for its good**

There are 2 reasons for this phenomenon.



First, the Java language is huge. It has a high abstraction level, with lots of type-safe features. Each language feature that can be coded in Java is coded in Java and then the syntax is translated to a JVM-specific syntax. Since Java is not yet an interpreted language, a developer needs to compile the Java code into bytecode, which is executed by the JVM. Scala, on the other hand, is not an interpreted language, and in general, it has many fewer language constructs.

Also, the features in the Java language are not the only features of the Java language. Most Java developers are not aware of the bigger Java picture, and even if they are aware of the bigger Java picture, they can't keep track of the entire set of Java language features. For example, the following Java code is not safe:

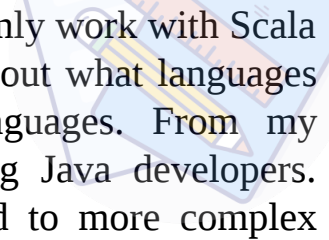
- `public class Person { public static int id { get ; set ; } public static String name { get ; set ; } public static int age { get ; set ; } }`

It might work, but it could also result in a `NullPointerException`, and because the language offers few types, it could even result in a runtime error (such as an undefined variable).

Note that this language error can be detected via checking for null-safety, which is an aspect of the language which is very similar to function safety, but there are other aspects to functionality that the Java developer is not aware of and does not have time to learn. Functionality such as stack safety and perhaps memory safety is often easier to implement than functionality such as null-safety.

The second reason why Java is bloated and not easily understandable is that the Java language specification is very large. The Java Language Specification is 1,835 pages long, and this includes everything the Java language offers. So the Java language is so large that it can be hard to read the specification. The fact that this specification is also difficult to work with makes it even more likely that people will overlook the issue.

## **Exposure to more complex languages**



According to the Code Trends Study, most developers only work with Scala and Python, but I am not aware of any survey to find out what languages developers are using if they work with other languages. From my experience, Scala and Python are not popular among Java developers. However, among these developers, many are exposed to more complex languages.

For example, when working with other languages, the Java developers can learn things such as:

- GCC or MinGW/PowerPC compiler
- javac and gradle (inherited from the Java developer)
- compilers GCC or MinGW/PowerPC compiler javac and gradle (inherited from the Java developer) Language tools such as the Reflection API (or methods like reflection or function introspection)
- Netbeans IDE (or Eclipse) and other Java IDE
- Portable libraries (e.g., the ones included in the JDK, such as `javax.outs` or `java.util`)
- Different capabilities of the JVM (such as high-performance threads and fine-grain concurrency control)

There is little downside to these additional capabilities, since the new capabilities typically offer only a slight speedup for existing code. Also, if Java programmers know how to use them and can implement them, they can make their existing Java code faster.

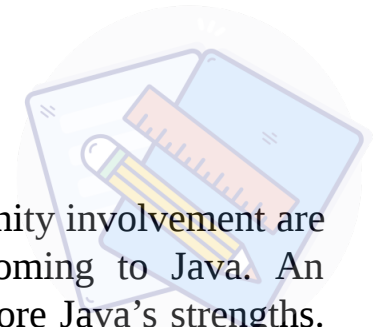
Some of the Java language features are used by other languages. This is a natural outcome of having an evolution-friendly language (as opposed to a reactive language). Many Java language features are useful, but not widely used, because of the different capabilities of the Java virtual machine. Also, some Java language features are useful but not widely used, because the developer community has not done a good job promoting or sharing them with other developers.

Additionally, I don't think most Java developers are exposed to additional capabilities offered by the JVM. Many other developer features have a significant impact on the efficiency of the JVM. The most important example is dynamic code generation. These techniques have a significant impact on the performance of the JVM. However, because most developers don't use dynamic code generation, they are not exposed to its benefits.

Programmers interested in learning more about the capabilities of the JVM, and how to implement them, can check out the excellent documentation for the Oracle JDK, such as JVM internals, JVM tips, and so on. This documentation has sections that explain how to implement various capabilities of the JVM, such as compile-time metaprogramming and dynamic code generation. However, many of these capabilities are difficult to understand.

So, I proposed six reasons why `java.util.concurrent.Flow` is not a good way to use for error-handling in an application. In the next article, I will discuss why it is a bad way to use for error-handling. This article describes how a thread synchronized access to an object. The next article discusses why, in my opinion, using a thread to synchronize access to an object should be discouraged. Finally, I will explain why I consider synchronization primitives as a programming technique to be deprecated.

# Conclusion



Java's worldwide standard availability and free community involvement are by far the biggest advantages for any developer coming to Java. An implementation that supports it is not an excuse to ignore Java's strengths. The biggest benefit for Java is its testing tools because they work on virtually all programming languages and also offer integration with IDEs.

Some of the language level issues in Java:

- jdbc adapter is not implemented (there are way too many exceptions to be documented here)
- Classloader is extremely complex (but the good thing is that you don't need to fix it from scratch)
- Naming issues
- Interfaces as abstract classes

Java's syntax allows developers to improve the readability and reduce readability issues. It doesn't mean that other languages couldn't do it, but that's not a general rule.

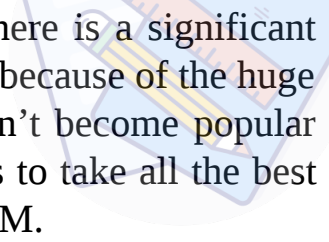
JVM has a complex GC. But it's very consistent and knows how to behave.

Java developers can build massively scalable, low latency systems.

There are a lot of tools in the Java world to improve debugging and analysis. Some important ones are:

- JUnit
- JDebugger
- JUnit profiles

**Why has Java been around so long?**



It took Java years to build up as a viable language. There is a significant barrier to entry for people coming from other languages because of the huge complexity of the language and the compiler. Java didn't become popular until Borland built the Apache JVM which had the guts to take all the best parts of the C/C++ compiler and compile them to the JVM.

JVM has been around for 10 years. That's a long time, in this short time, there's a lot of innovation in the field. Especially in the Java 8 version, there was so much innovation.

We (Apple) care about Java and spend a lot of time working on it. That's the right way to be doing it. That's what other companies should do. If Java was a new technology, then they would make the best of it, but that's not the case. It's an old technology, and people need to understand that we're not going to drop dead because you started using Java.

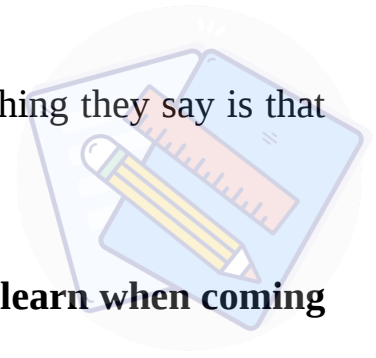
### **How can someone easily switch from Java to other languages?**

Different languages solve different problems. Java solved the problem of running a computer, programming a computer, for a programmer, and sometime in the future, an industrialist. If you can solve a problem, that's a high bar, and if someone can solve it, then why not give it to them?

A lot of users who become developers can't solve the problems of systems programming or system programming. That's where dynamic languages like Clojure and F# step in. Even languages like Rust and Go can step in. It's easy to switch from one language to another language that solves a particular problem.

I don't see any language that can be used to write a large, complex, data-driven system. Because there are just too many things happening. If you ask a Java developer "what's the difference between Java, C++, and .Net?"

Most people say Java is a bad language, but the other thing they say is that it's powerful.



**What do you think are the most important things to learn when coming from another language to Java?**

First, it's about the mindset. You have to understand that writing programs in Java is really easy. You don't need to worry about memory management, how do I know I'm not going to do something stupid with this thing and it could cause a memory leak. You have a bunch of arguments, and you just say "this is the way to do it." That's how Java has the power to do what it can do.

But it's not just about Java, it's about your mindset, and how you approach problems. It's not enough to know how to write Java code. You have to understand how to do the hard parts of it. For example, why don't I care about cache safety? What do I care about? If I have ten objects on a table, how do I want to handle that? It could be that I'm going to write a complex algorithm that handles caching and perhaps I don't care about memory leaks.

You can create Java-like VB, but that's not the right way to do it. When you go back to C#, or C++, or Visual Basic, you can see that a lot of these people spent 10 years learning how to do it. When they were first starting they probably had some experiences with other languages, they just didn't realize it.

I've spent about 25 years and I spent that time learning. Some people spend 1, 3 years learning, some people spend 5 years, and some people spend a long time learning. I would say you have to have some experience to know how to do this. Then you can go out and learn how to do this stuff. I'm sure some people have spent 20 years before they understand it.

Most people are surprised that my question about the difference between object-oriented languages and functional languages came up.

People didn't understand it because they didn't know how to program.

It's just a language for telling people how to program, how to describe the problem. If you look at some functional programming languages like Scala or Clojure, you'll see it looks more like mathematics. There are functions, functions that have more than one parameter, pure functions, lazy functions, functions that have a return value and you don't see them a lot. But they're easy to implement, they're easy to implement in functional languages.

Some people like Scala are used to having imperative languages, where you have to iterate over arrays, and update methods, update state. Functional languages are a little bit more functional. They don't have that as much, but they have that kind of thinking in there, and it makes it easier. But you still have to have good programming skills.

Some people think that the other way is bad because they don't have to think about memory. One person I interviewed said, "people who use object-oriented languages think they have to worry about memory," but they don't, it's almost an illusion.

Some people who are used to functionally doing things will say that functional languages are harder to learn because they have to think in functional ways. Functional languages might be easier to learn, but they also might be harder to use. There's some of that in Java.

Yeah, I'm sure that they are, but the good thing about functional programming is that the reason you have to think that way is that functional languages are more restrictive. Java doesn't care about that. The same as Java doesn't care about concurrency, in Java, you can't have lazy operations, you can't have pure functions. They're all part of the Java language.

You have to know some functional programming. I would recommend when you read some of the books, I would recommend learning some functional programming languages like Clojure. If you want to learn Clojure, I would suggest the book “Clojure for the Joke,” or some other book that teaches about Clojure.

Clojure is another example, one of the reasons people have trouble with Clojure is because they're used to imperative languages.

Java is more general-purpose, Java is a general-purpose language, it can do anything, but people see it's a functional language and they're all excited. But you can't do anything with it.

Then you look at a Java library, and you realize that they're not functional. If they're like jQuery, which is a library for JavaScript, and there are a lot of functional libraries for JavaScript, people like those too. If you look at the Erlang project, there are lots of them. When you're doing Erlang, which is a functional programming language, you have a lot of these functional libraries for Erlang, but they're not really in Erlang.

There's a big community of functional programmers, but you don't see a lot of them as developers. They're more like innovators.

We have a lot of people like that in Erlang, and they come up with new ideas. They come up with new features and implement them, and maybe others will use them later. Maybe they'll have a smaller company, and their clients will want a new feature, and they'll go and implement that and maybe sell a smaller company.

All those ideas are there, that's the nature of it, it's the nature of most programming languages.

It's the same with functional programming. It doesn't mean that it's more important than imperative programming. You can write programs in functional languages just as well as in imperative languages. It just means



that your programming skills will improve if you learn a functional programming language.

It depends on the thing you want to do, but if it's something that's driven by functional programming, you're probably going to be better off. You'll learn about functional programming. It'll increase your communication skills. It will make you able to build those things more efficiently because you'll be less dependent on conventional programming practices.

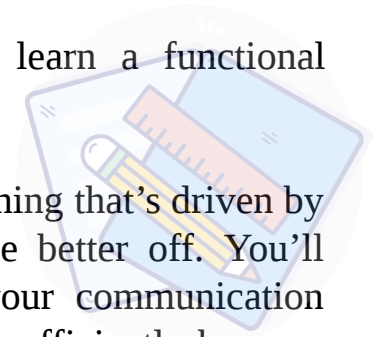
Oh, and I think that people who've had kids will appreciate the importance of functional programming. That's kind of a given, when you have a kid, you'll want to have a program, and you'll want to spend some time with it.

I remember having a kid when I was at the University of Toronto in the late '70s, and there were no such things as microcomputers at that point. We used those little weird computers in the student union, and we would program programs with them. The programs that we would write back then weren't written for the microcomputer, we just wrote them in assembly language.

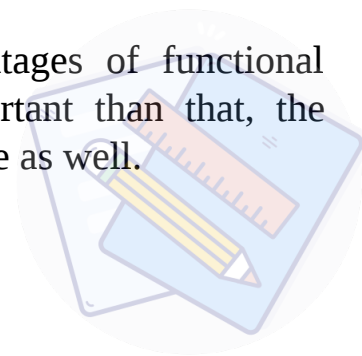
But now, you have those programs with C++ or Java or C#, and you have to run the thing. It's a machine-language program. You have to change it into machine language. And all those errors that you have in the assembly program you wrote are going to mess the thing up in the machine language.

You go back to functional programming, and you have the code on a stack. You have a stack, and you have a couple of functions you want to run on the stack. What you say is, "I want to put these functions on the stack." It doesn't matter that they're in the wrong order. What matters is, "What order are they in? How do I interpret them?" You don't know that until you interpret them.

And that forces you to write unambiguous code, it forces you to do it differently. And you can do it more efficiently, you can do it more effectively.



When you do that, and when you learn the advantages of functional programming, you'll discover something more important than that, the people who've had kids will find that to be an advantage as well.



## **Introduction to Functional Programming**

In this session, I'll show a little bit of the Erlang programming language. Then I'll show the BufHandle and a few functions, then I'll show a few functional-style functions.

But that's not enough, so let's start with an introduction to functional programming. I will introduce the following terminology, and I will try to make it clear why we use this terminology.

First off, I'll say that we use the term "Functional Programming" to mean "Unary Operations." Unary Operations are the functions that are called on a single input and that return a single result. And you say that as one word, and then you explain it. But you can also use the term "Functional Programming" as a blanket term for programming in general.

Some people say, "I want to program in functional programming." You could program in Erlang using some purely functional language, or you could program in Erlang, and then use Haskell, or Scala, or whatever the hell you want.

But I think it's more productive to program in a purely functional language if you want to do it. But if you don't want to do it, then, by all means, do it in an imperative language. But I think it's more productive to program in a purely functional language.

One of the other things to say, the other term is "unary operators." So we usually think of an expression as being either an AND or an OR. You can put the AND on the right, or the OR on the left, and they're both on the right.

But these are not correct; you have to use not OR, but AND OR. So you have an expression and then you have two sides. One side would be an AND, and the other side would be an OR.

And so when you put an AND on the right, and an OR on the left, you get a new expression on the right. So you can say that these are NOT in order. They're not actually in the proper order. The correct order is like this: AND OR, OR OR. But you get the idea.

So the other terms are also not correct. You can put an AND on the left, and you can put an OR on the right. It's not the same expression; it's different.

In Erlang, you do not need these terms. There is a feature of Erlang called partial application, which allows you to put the function on one side of the function, and put the side on the other side, and then the body will be applied to that function. So if you put an AND on the left, then you will get a different function. If you put an OR on the left, you will get the function. It's kind of magic, the way it works.

You see this on every call to the standard library. You've probably got something on your screen right now, like for example, the standard library will give you the list function. It's not in the standard library. But if you do not specify a list argument, then it will give you a list of all the lists. If you do specify a list argument, it will give you a list of the pairs. So you can have one pair of two lists, and you can have another pair of a pair of lists, and then you can have another pair of a pair of lists. You can have every permutation. It's just a permutation in the permutation language.

Now the version of Erlang that I'm working on will probably have an API that will support list functions and other forms of partially applied functions. So you can probably see why you should use imperative programming in Erlang instead of functional programming. Functional programming is much easier to understand than imperative programming, but when you're dealing with critical systems, where you are using the standard library, it can be very hard to think about all of the available

options, and how to apply them to your problem. And if you're dealing with critical systems, you probably don't want to think about it at all.

Another good reason to use imperative programming, in my opinion, is that it allows you to do this thing called typesetting. You write your programs in a way that it gets compiled, it is converted, into a way that it can be read. There are different levels of typesetting in Erlang, for example, the compiler can automatically convert your program into a functional language. And then the different parts of your program, like methods, you can write them in a functional language.

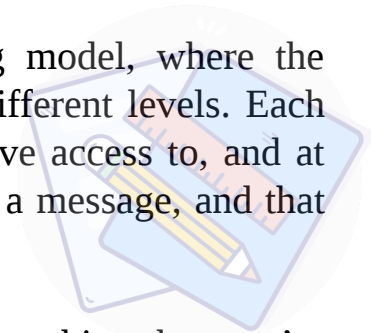
The same is true of Java, C++, or any other language you might want to use. The compiler will recognize that it is a language and that parts of the program are going to be executed in the imperative language, and the rest of the program is going to be executed in the functional language. And this is the dynamic type of programming.

You write your programs in a way that they can be dynamically translated. That is if you write a program in a functional language, and you want to put some data in there, like a function or two, then those data are going to be automatically converted into the form of an imperative statement, or an imperative data structure, so you can insert them in the middle of your program, and see what's going to happen. And then when you run the program, you can get the effect of those data that you were putting in there.

So if you're not afraid of dynamic typesetting, then you should probably consider using Erlang, and your programming style will probably change a bit. But you can't see it because you're just staring at your screen.

One of the last examples I'm going to give is about distributed computing, where you have multiple nodes interacting with each other. This is pretty much the same paradigm that is used in distributed computing with cellular automata, in that in Erlang, every process has to be able to communicate with every other process that they have access to. And it's cool.

It's kind of a global, distributed, parallel computing model, where the communication between processes can be at several different levels. Each process has access to all of the processes that they have access to, and at each level of communication, there's something called a message, and that message is what is sending and receiving data.



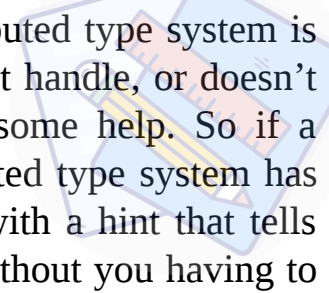
For example, you might have a process on the same machine that you're running Erlang in that has two different mailing lists, or maybe one email list. And they can both send mail or reply to each other. There are a lot of things that you can do. You can set up distributed processing, which is neat.

One of the things that you do have access to, but that many other languages don't have access to, is the communication between processes. To write programs in a distributed way, you have to learn how to write programs in a distributed way. So in the previous example, you had to learn how to write distributed Erlang.

One of the other things that you do have access to is a distributed type system. The type system in Erlang is not just static type checking; it's not just static type checking for scalars. It's also dynamically typed.

In other words, if I write a program and I give it to the compiler, the compiler automatically recognizes that the code that I am giving the compiler has to be able to handle the fact that this program is running in an interconnected cluster of nodes, and the nodes are communicating with each other. And the compiler has to know how to recognize when these nodes receive the messages sent by the nodes that are a part of the cluster. So there are also distributed type systems.

One of the cool things that are possible with a distributed type system is dynamic type checking for Erlang nodes. So if I have a type system that I'm familiar with in my language, and I want to have a distributed type system, the cool thing is that I don't have to do anything special. Because if I have a type system, and I can communicate with the distributed type system, I can send messages to the distributed type system. And it will type check the code that I give to it. So it's very cool.



And one of the other cool things about having a distributed type system is that if there's a type that a distributed type system can't handle, or doesn't know how to handle, then it will provide you with some help. So if a message comes in that is not something that a distributed type system has yet to deal with, then that message will provide you with a hint that tells you how to deal with that message, and it will do so without you having to read the code for that message.

So that's another thing that's cool about Erlang. And when it comes to these distributed type systems, it's not just distributed type systems; it's also distributed type checking. So if you write code that has the correct type signature, and you send that code to a distributed type system, then the type checker that the distributed type system provides will validate the code, but it won't automatically type check the code. It will just tell you what the types are and where they come from. And when you read the code, you'll know what the types are.

And the nice thing is that these distributed-type systems exist. They're something that you can go and talk to if you're not familiar with them. They are being built. So I have an IRC channel that has people that work on them, and I have a mailing list that people that work on them use to talk about them. So the future is full of distributed-type systems that work with Erlang.

This is all interesting. But there's a problem with this. I'm building a distributed type system, and Erlang is a distributed language. So how do I communicate with my distributed type system?

There is one protocol that Erlang does have. It's called SIGTERM. And it's really simple. It says, "Go away for a bit because you're going to be blocked in here for some time." But it also has other more elaborate protocols. One of those is called SIGQUIT, which is something that when you call the system method on any type in Erlang, it will eject the type, and then it will tell you that the type has ejected itself. And that's it.

So SIGTERM and SIGQUIT are used for two very different purposes. So when you call SIGTERM, you're telling the language runtime that something is going to block in your program, and you don't want to see it for some time. And when you call SIGQUIT, you're telling the language runtime that you don't want to see the type again, for any reason, ever.

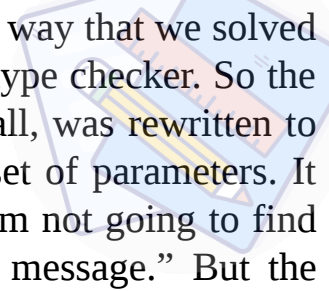
But there are a couple of problems with this. So if I want to share data with my distributed type system, then I have to use two different ways of communicating with it. I have to use SIGTERM, and I have to use SIGQUIT. If I want to see what type a message is in, I have to know the type of message that it's emitting. And if I want to communicate with that type of message, then I have to know the type of that message.

So if I want to let my type system know that I've finished using it, I've got to use the system method, which tells it, "Okay, I'm done using the message type. I don't want to see it anymore." And then I have to use SIGQUIT to tell the type checker that I don't want to see that type anymore.

So this has some serious problems. So the first problem is that if you have too many parameters in your system call, then when the type checker is getting to the message, it has to look for the parameters. So it has to check all of the parameters that are in the system call to see if it can find them. And then it has to look for the parameters in the message and find out if it can find them. So if you have a long system call, it'll just be completely jammed. It won't know where to look, because it has to check the whole system call before it can find the parameters. And it won't look for the parameters in the message, because it can't find them, and it can't emit the message until it finds them.

And the second problem is that it will emit the message even if you call SIGQUIT and use SIGTERM on it. So if you have a long system call, you'll get another message back that says, "The type checker is trying to emit a message. It can't find it, but please let it emit it." And this means that for really long system calls, you have to run the entire system call again. You'll get two messages. One saying "No, no, no, no, no, no, no, no," and one saying, "Let it emit it." So it's not a good idea to do this.





So these problems, we found, are hard to solve. And the way that we solved them was by changing the system call that handles the type checker. So the code that handles the type checker, the block system call, was rewritten to use a different system call, which just has a different set of parameters. It has a little simpler set of parameters. So it's saying, "I'm not going to find the parameters, and I'm not going to try to emit the message." But the reason that you're not allowed to do that is that it will cause a memory leak. The storage that the type checker is using just cannot fit into your process's memory. So if you want to use the type checker, then you'll have to be sure that you're only sending it data that is in a structure that will fit in the space that the type checker is using. So that's why you can't do that.

Then when you get a new type, that is a really big type, then you just have to say, "Look, I'm done. I'm going to need to unpack the structure to see what the type is." And then you can emit it. It's still a memory leak, but at least it's not a huge memory leak.

So that was the other major improvement that we made for making that new type of system a lot better.

So you have a lot of responsibilities, including some of the problems with the type checker. So you have, of course, the normal responsibilities that all the people on a type system have. So you have to keep the type system secure. You have to keep it from all kinds of attacks, from any kind of impostors or impersonators. And there's a lot of things that you need to do to make that secure. So one of the things that you have to do is to have somebody put in some work to be able to call your type system.

But there's another part of your responsibility, which is that you're also responsible for trying to protect people from themselves, trying to help them to write better programs. So this is going to get even harder than the problem of giving up your code because people will often be trying to write better programs than they can. So a lot of times, you can't do a lot of the things that you'd like to do. So this is a bit of a tradeoff.



I've already talked about the existence of the type checker. So there's no point in having a type checker if you don't want to use it. So we made the type checker available as a library so that people can write their code in a language that they understand, and then they can just call the type checker from the outside of their code, which means that you don't have to run it.

But even without using the type checker, if you're just writing your code with the assumption that you will be using the type checker, then you have to also have a way to make sure that your type definitions are all valid. And a lot of the way that type checking works is to try to figure out what you mean by the type, and then you have to check your code against it, and it's quite expensive.

So the other thing that we did was that we made it more economical for the type checker to do the checks for you. So if you write a check that doesn't match, you don't have to worry about that check. The type checker will just make sure that it doesn't do that. So if you get a check and you don't match, you don't have to worry about that check. The type checker will just forget about that check. So you have a program that is free from the worry of having to match, or having to check, all of your type definitions. And if the type checker did some kind of analysis of the type that you're writing, then it would recognize that the language didn't have a problem with it. So if you get some type of checker that's much more powerful than ours, then they will be able to solve the problem for you, even though you're not running it.

And so there are two ways to solve this problem. You can write a checker. It's not a particularly good idea to do that, because then all of your code is so brittle that you have a checker that it would break, and your code will still break. So that's not a good thing to do. But then you can write some macros, which is much more robust. So this is a way to solve this problem by doing macros, which I think is probably a better way to solve the problem. So the problem is much harder if you do try to solve it by writing a checker because then all of your code is going to be so brittle that you don't want to run the checker because it's going to make your code too brittle. So actually, writing some macros is a much better solution. And of

course, you can write the macro at the end of the program and if it gets into a failure, then you can make sure that the type system has caught it.

Okay, now I'm going to talk about one more thing, which is that because of all of these limitations, I'm sometimes forced to rewrite things, which I don't think is necessarily always a good thing. I mentioned this last time, but I was forced to rewrite part of the library to fix a bug. And I did rewrite the library to fix this bug. And the reason I chose to do this was that I think the idea that you could write a perfectly type-safe program and if I ever saw a type error I could go and fix, it's just not true. It's not true, because the type checker is going to still be there, and it's still going to check that there's no type error. So when we do find a type error in the types that we were trying to check, it's going to break that check. So you should not use type checking to catch type errors.

And the reason is that you're effectively trying to catch type errors all the time, and if there is a type error, your program will always fail. The other thing that you can do is fail early, you can print something to the user, and if it's a type error, that will never happen. The user isn't going to be able to see the message. But also, if you're not giving the user the ability to print, then it's not showing that type of error. So the main idea is, I think, that you should only use type checking as an exception detector. And this is because I think it's not practical to do type checking everywhere. You can't, as I said, write a whole program, a whole runtime system, if you want to check that everything is correct. So what we need to do instead is focus on what's necessary for the runtime system and what's necessary for the UI to be run correctly. So this kind of balances it out.

## **Using pattern-matching to match on more data structures**

So you know how pattern matching works. And the thing is, sometimes, you have to make a huge, giant type check and you need to start annotating everything, and so you have to make it more brittle. If you use pattern matching, instead of having to annotate everything, you only have to

annotate some things. So when you say matches on tuples, we can just say this part matches on tuples and that part doesn't. So now we can have one type of checker that says, "This is a case where this part is a case where". That makes it much easier to do type checking.

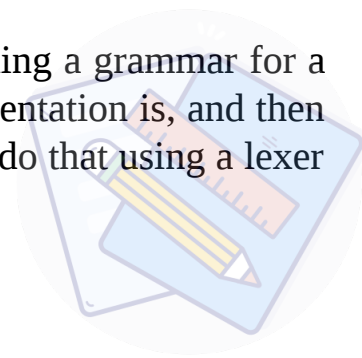
So for example, I sometimes want to put objects into an array that's going to be kept in memory in a distributed fashion. So I want to be able to put any type into a hash, which is how I'm going to manage this, into this array. And so that's exactly the problem that I was trying to solve before. So I can now use this pattern matching to do that, and this way, I can annotate only these cases and only the relevant cases. So you don't have to have this huge huge type of checker everywhere. You can just have this little pattern checker. But the other thing is, that with type checking, you have the whole library, and then you have a single class that you use for pattern matching. You can't do that with pattern matching. You can't say, "Oh, okay. So I have this method that matches on this pattern, and it can only match on this part of the system." If you use pattern matching, you can only match on this part of the system, and that's the only case that you need to check.

So also it's useful, because when you have these large type checks when you have these type checks that are going to come down the wire, I also find it useful, this matching. So that's kind of a separate question. And I'm not even going to tell you how to do pattern matching, but this is also another approach. I think it also fits the model that I'm talking about, where we do a lot of object hierarchies, and it's very easy to forget about one level or the other and to fall back to type checking.

### **Using the WSDL for validation.**

So before you write a library, it's a good idea to have a signature, a validator, in the system. So we've got these WSDLs, and they tell you what the documentation of your library is going to be like, and that makes it very easy to keep it straight, what documentation you're going to have, what requirements you're going to have from the users. So I think that's a good

thing. And then the way I use that signature is by building a grammar for a single pattern, and then you can define what an implementation is, and then if you want to validate your library concretely, you can do that using a lexer or a parser.



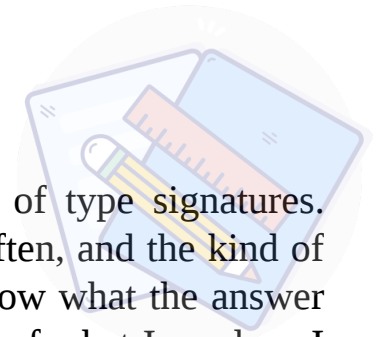
## **Faces vs. Extensible Markup Language.**

So I think this has a lot of applications. So for example, maybe you have a huge codebase, and you want to get rid of the language and just want to use HTML markup. You can do that, and that's what we do in the blog package. But the big problem is, you have to make sure that you have syntax highlighting for every code that you're using. And when you're doing that, you have to make sure that the people who are going to use this library are used to seeing HTML. But you can do that, you can have syntax highlighting, but if you're using that kind of markup for your actual code, then it's very confusing. So to use this markup for your code, then, yeah, it's also a good idea.

So I've done that. And then I've used this idea that I change the regular expression each time you use it. So I'm going to mark it up and then change it all the time. And that's kind of a good idea. So you do not have to be stuck with the regular expressions that you have when you are writing your code. And that also has a lot of different applications. I don't want to go into that now, because we've got the talk, but I think this is a very good approach.

So actually, just taking a look at that type of signature, you may have noticed that in the library, there is no special type for reading from a file. But in reality, people do read from a file, so you have to include some kind of package for that. Or you might be using sockets, and you're using sockets a lot, so you have to include a socket package. So I think it's also a good idea to have these packages that have extra support for that kind of stuff.

## Tools for the Real World.



This is another case where I'm thinking about a lot of type signatures. There are some kinds of functionality that I use very often, and the kind of signatures that I use is to find a number, and then I know what the answer is, because I get the number and I get the specification of what I need, or I get the number and I know exactly what I'm supposed to do. And when you work on a large codebase, that kind of signature often works pretty well, and people usually use them. So you can have a special type for that, that is a pattern for a function. So say, you have a file that you want to open, you can put something like a file open. That's the package that should contain the signature for the function, and then when someone's going to read that file, they know what the answer is. They can run it against a file that has a similar signature. You have a file that has a different signature, so you can use that, and you just get the kind of the signature, and you use it in your code.

But the real world is not like that. In the real world, you have to make sure that people use that code, and then you have to also make sure that you test that code, and test that code with real people, and test that code with real bodies of code. So you have to be more flexible, and you have to be more kind of, maybe more careful about the types because you don't want to break the types that people are using. So it's also a good idea to have a more flexible type system in the real world.

A more flexible type system also means that you can make more cases where you use different types of functions, and it might be easier for them to know what to expect. But, at the same time, you can also make the real function that someone's going to use more complex, so they can be more sure that what they're going to get is what they're supposed to get. So you can do that with a type system, and also you can do it with other ways of limiting the access that people have to a certain piece of functionality. I'm not going to talk about that in this talk.