JAVA

The Complete Guide for beginners to Learn Java Programming *FAST*

LEONARDO GORMAN

Java:

The Complete Guide for Beginners to Learn Java Programming Fast

Table of Contents

Introduction

The aim of this book is to help assist you with learning how and what programming and Java are. Most of us have heard the terms “computer programming” and “coding” but may not know much else. Computer programming has been around since the invention of computers—as it is the manner in which a computer is organized in order to run various programs. Coding is the ability for someone to speak the language of the computer through a computer programming language. Just like with humans, computers have multiple languages they can use to speak to each other.

If I have gained and peaked your interest with this first paragraph—this book is most definitely for you. Most of us are greatly interesting in why and how we could learn how to gain more knowledge of control over a certain subject or thing—computers are actually quite easy once we know the basic philosophies! Through the course of this book, I hope to have you comfortable to the terminology and basic functioning of computers in the fields of programming and using Java within the coding and programming world. Although I cannot promise you will know everything about Java and the computer—I do promise that you will have enough skills and comfort with basic principles you will be able to play in your own computer sandbox (more about that in the first chapter)!

Towards the end of the book, I go over a lot of helpful hints and resources that can be used to get a greater grasp on Java itself and how to use it. Although some of the many hints and terms may not make much sense to you know, remember that you are building a foundation and resource list to come back to as you run into speed bumps in your coding and programming journey. I encourage you to read through all of them and try to retain what you can—and come back for the rest when you need it.

I encourage you to follow along with some of the ideas in this book in the Java program itself. Using this book as a how to guide through your programming experience that can be used in tandem to create the most long term memory for learning new things. Make sure to try things multiple times to make sure you understand the concepts completely and can easily replicate them at any given time, even if you need to look up a resource to be able to find exactly what you are looking for. Although it is not humanly possible to memorize everything, this book should give you general knowledge you can use in order to feel like you have a firm grasp on the beginning concepts of Java.

With that, I look forward to you working your way through these chapters. The aptitude for learning skills like computer programming are really only inhibited by the amount of work you want to put into them. We shall start the adventure into programming with Java with an entire chapter devoted to the basic terms and functions of programming, giving you a firm foundation in the virtual computer world.

Chapter 1- What is Programming?

More than likely, you have heard the term “programming” before in your life, either on the internet or in a conversation with someone about technology. If you are like me, you may have just smiled and nodded your head, in order to appear that you may know what the person is talking about—rather than admitting that you have absolutely no firm knowledge of what it entails. When it comes to computer programming, Java is the language we are looking at within the world of computer programming. It is important to have a basic understanding of programming before we can start getting into the nitty gritty of what Java is.

Most people can operate a basic graphic interface on a computer by clicking on icons or typing in a Word document to prepare a paper for school or a resume to gain employment. However, the original computers did not have these graphic icons and looked more like blocks and blocks of different texts strung together. In order to understand what all of these countless characters on the screen meant, people had to learn how to speak the language of the computer. This is what programming (or computer programming in formal terms) really boils down to—an understanding of how to speak a computer’s language to get it to carry out the tasks you want it to do.

When you program, you always want to remember that although there are language rules and structural syntax rules that only allow you to do certain things, you can always improve upon the existing if you want to. There may be something that works well for others and not so much for you. If for whatever reason the way something is done does not make sense to you, look to outside resources in order to inform you as to why things are done in a certain way. Within the pages of this book, my goal is to give you a pretty good understanding of why and how things are done in Java; however since this is a beginners guide, there are certainly a lack of more intermediate and advanced options and descriptions that you may be ready for in a short period of time.

In order to learn how to speak Java to a computer in order to program it, first a somewhat in depth idea of what computer programming is invaluable for a new programmer in training. The basic activities of computer programming are developing an understanding of programming itself, how to generate algorithms, ability to understand the computer analysis, idea of how to verify that your algorithms are correct and finally to implement (or code) your desired algorithms, in this case, Java. First here are some key terms that are also helpful to get used to using while developing your computer programming language skills.

Common Helpful Programming Terms and Definitions

- Algorithm- a way in which to use logically or mathematical skills in order to solve problems
- API- stands for application programming interface. It is essentially a set of rules that allow code to communicate with other websites or software
- Code- also known as source code. It is a collection of computer instructions written using a programming language

- Code Editor- a type of text editor program specifically designed for computer programmers. Can be used as an application or can be integrated into software or a web browser. You essentially are able to check the code you have written, like a traditional copy editor in a newspaper or magazine, just computerized!
- Complier- This is a program that allows for you to take code written by you and is translated into what is known as binary (essentially ones and zeros that make up the actual machine code)
- Concatenation- this is the ability to combine strings of text together (things like two lists)
- Control Flow- this is the computers control over the order that sections of coding are run. Essentially it is like rules of how the flow of data moves
- Dump- this is a list of data or something like a text file that is saved in the event of your program crashing. It can be very helpful in diagnosing problems (why the crash occurred)
- Floating point- basically it is a decimal number where the decimal itself is referred to as the floating point (ex 4.56)
- Function- written instructions that obtain a particular result with the ability to be used simply by writing them or "calling" them
- GIT- this is a portion of distributed software used for revision control that is used as a source code management system (also referred to by the acronym SCM)
- GUI- this acronym stands for General User Interface referring to the software you are able to see and interact with (the icons or programs you would generally think of i.e. internet browser)
- IDE- this is an acronym that stands for integrated development environment. It is slightly more advance than your basic code editor
- Interpreter- this is something that allows for translations of the computer coding (machine code) when a program is in use
- Iteration- instructions that are repeated
- JSON- this is a format used to transmit information to and from locations
- Libraries- pre written code that is used to implement common coding features which you do not have to write yourself
- Logical Operation- using Boolean logic (and, or and not are the key words when using this type of operative computer coding)
- Loop- the use of a piece of code that runs until a condition placed on it is filled
- Markup Language- a language format, like HTML, that is relatively simple and used to format pages
- Nested- an instance where one thing is contained inside of another

- Pseudo code- general description of the pathway between code and written language. This is code written in English that allows for a message to be sent to tell a given piece of code what to do
- Recursion- this occurs when something actually refers to itself
- Run time- this is the time that a program is in use, or running
- Sandbox- testing ground in which to run a program test
- SDK- an acronym that stands for software development kit. SDK is a bundle of software tools that allow you to create new applications for specific framework or platforms
- Subroutine- a portion of code or a function that can be run anywhere inside of a program
- Syntax- the set of rules in a computer language that guide usage or correct structure
- Variable- this is data stored that has the ability to be modified by the user whenever they would like to

Now that you have a small dictionary of terms to help you understand key concepts, I will now provide more thorough descriptions of how and what programming works and what it is. These next few paragraphs will give you an excellent foundation of knowledge for what and how to program. This beginner's guide to understanding will have you set on a proper knowledge and explanation of basic programming principles.

What is Programming?

Computer programming is defined as a process where one takes a computing problem and turns it into executable computer programs. The end goal of programming is to be able to discover instructions in a given programming language that will allow for solving a problem or automate performing a specific task. In order to fully understand how to execute this process, one should have good analytic skills, an understanding of how a computer works, be able to understand and generate algorithms and finally implement (or code) algorithms of a given programming language. Keep in mind that there is more than one language, so the world of programming is much, much larger than the scope of this book.

In addition to these basic principles, I want to reiterate how learning computer programming code is really laying the foundation for either the system itself or a program to be used in it. When you are programming you need to be debugging and testing while maintain the source code. Essentially, once you start doing coding for yourself of a program or system, you need to be able to problem solve and diffuse the issues with code quickly. This book should serve as an excellent beginner's guide and give you skills you may need to trouble shoot issues you may have while starting the process.

Also with programming and coding you are learning how to build and implement the system. These terms are often used interchangeably to describe programming of the

computer itself, creating source code or implementing said programming or coding. Once you get more familiar with what programming and coding can do you will learn how they really work together in order to create the virtual environment, buy also be able to identify the subtle differences.

How to Generate Algorithms

Algorithms are important because they allow for us to do things, in the computer world, like schedule tasks, design computer algorithms and even solve other optimization issues. Essentially, they are something a computer does to instruct it to complete a task in an efficient manner. Sometimes people even refer to them as recipes because they are similar in the structure and the fact that if you find a good way to carry out a certain task, it can be very rewarding (much like warm chocolate chip cookies are from a traditional recipe).

As for you and creating your own algorithms, this can be a complex task for any computer developer, even more so when you are first starting out. The good news is that you probably will not be dealing with high level algorithms for a long time. Those algorithms that you will be using are things like creating everyday functions or writing loops. Essentially, you are writing an algorithm to complete a task for you. There is a simple test list for creating an efficient algorithm design:

- You want to use language features that enable you to reduce operations
- You want to reduce iterative loop nesting as best as you can
- You want to have the ability to define variables outside of loops when it is possible
- Try to use automatic loop indexing instead of manually doing so
- Try to use clever reduction techniques (things like conquer and query optimization) in an effort to minimize the size of recursive processes

Basically, when you are creating or using an algorithm, you want to settle on something that not only completes a task and solves a problem—you also want this to be done in the most efficient way possible. One way to gain insight to what is efficient is to look at the work that has already been done by others. Within this book, I will not be able to give you a complete understanding of algorithms but I most certainly want to give you a good place to start. I am going to give you an examples and, from here, I hope that you will have a good jumping off point for your future projects.

This example is a very simple and somewhat standard example of an algorithm. It is often referred to as the “Hello World” program and the following will be an example of it. In the next chapters, I will go over how to use and run Java, this is just an example of something you may want to input and try (beginner algorithm).

Sample Algorithm

public class HelloWorld

```
{
   public static void main (String[ ] args)
    {
         System.out.println(“Helloworld!”) ;
    }
}
```

That is a very basic example of an algorithm you may use to start to understand how your programming skills can be used. If ever you are trying to get an algorithm to function and not having any luck, check something simple like the spacing or wrong character or symbol—it usually boils down to a simple error.

Understanding Computer Analysis

As the word analysis suggests, computer analysis consists of looking and troubleshooting for problems or software issues within your system or programs. Essentially, being able to understand how a computer works is a key component to be able to isolate any issue and solve it. This part of the programming is something you should have a good firm grasp on, as with any new task to learn, it is good to be very familiar with how and why your hardware or tools operate. This is the same with computer analysis. Keeping up with the newest technologies, programs and incorporating how and why you should use a method or program is important to know. Picking up this book and reading it is a great start to learning how coding and programming works. After you have mastered Java, do not fear! There are many different languages and systems to explore and master as well. The greatest thing about technology is that it is constantly being innovated into the best possible thing it can be. Knowing how to secure systems and resolve issues will get you very far in many aspects of life, particularly, when sensitive issues or data may be at stake, eventually something you will deal with as a programmer.

How to Verify you Code is Correct

One sure way to verify if your code is correct is to try and run it. However, if the code is something you are doing outside of your sandbox, you have several other methodologies to use to ensure you are getting your correct message across. There are many different online websites, serving as code editors that you can utilize in order to check code. These are websites such as Tinkerbin, Write Code Online or Codepad. Basically, if you do a google search for a code editing program for any language you are using, you will be able to find a suitable interface to do so.

Having a sandbox to play around with your coding is a very efficient way in which to check your code. By utilizing this method you are able to create a virtual environment to test out code to make sure it will complete the proper function. This is important if you are going to code something that will effect the ability of a program, software or even website that you are attempting to build.

Another great resource you can use to help insure you have proper code is something called a library. As you are probably familiar with a library of books, this is similar as it is a large virtual library that contains code. Using this method can be great if you need to make sure something is up to par and working. However, I would caution that you always be able to understand the code you are using—even check that code with a code editor. You should be able to understand the code in order to fix it if a problem arises. If you just copy and paste from a library, without understanding, it could lead to huge potential issues down the road.

Now that you have a great, general foundation for what programming is, let us move on to Java specifically. I will lay out in the next few chapters some important understand of what and how to use Java and its functions. This will all build on the information in this chapter, so hopefully you have a fairly firm understanding of some of the main parts of the world of coding and programming.

Chapter 2- What is Java?

Now that you have a better idea of what programming is, let us move into learning about exact what Java is and how it fits into the world of coding and programming. As a computer programming language, which is what Java is, it is designed to be a language that it class based, object oriented, concurrent and has the least implementation dependencies as possible. Essentially, any platform that supports Java can run Java without a need for recompilation. This is one of the reasons that Java is one of the most popular languages in use today. One phrase that developers use to describe the multi-platform use of Java is “Write once, run anywhere” also known by the acronym WORA.

As a language, you may have a learning curve when it comes to the understanding of why and how certain things are done within the language. Typically within this book, I try to give as many explanations to you as possible without getting into unnecessary detail for you at the beginner level. With that being said, as with the English language, there will probably be rules and situations you do not fully understand why they are formatted or done in a certain way. Rather than spending time and effort into understanding these things, I encourage you to just accept them as they are. Although you may want a better answer than that, unfortunately I think only the creators themselves know the reasoning behind them—and it may even just boil down to their preference or the computers ability to understand.

Java was created by a man named James Gosling in 1995. He designed Java as a programming language that has fewer low level facilities than languages like C and C++. Essentially meaning that it focuses on the high level, essentials for running computer programs rather than the more minute and micro details that these other programming languages have. There are five main goals that Gosling used to guide him in the creation of Java, These are that 1) it must be threaded, interpreted and dynamic, 2) it must be object oriented, simple and familiar, 3) it must be secure and robust, 4) it must execute with high performance and 5) it must be portable and architecture-neutral.

These five goals explain why it is such a commonly used language. These are goals that allow for users to have a mainstream, useable and effective method for building anything from software to a website. User friendly programs are something that bring the complexities of computer programming into a more relatable experience, especially for new users and those just starting out learning. This is why I personally recommend and find it important for beginners to have a firm grasp on. It is very widely used and therefore pretty important to know and understand for anyone trying to work, or even just have as a hobby, computer programming and coding.

Now that there is a little background on Java, how do you get the actual program running on your own system? Basically, one of the more difficult things to start coding Java is acquiring all of the necessary tools to run it. Java is an independent platform, which means it will run on just about any operating system. The reason for this is a program called Java Virtual Machine (also called Java Runtime Environment). In order to start coding in Java, you need to download this program in order to do so. You can actually make sure you have this program, as some computers do have it installed, or install it through the Java website. There are options for many operating systems so

you should be good to go once you are able to locate it and download it!

So now you have downloaded (or discovered you had) this program—what does it mean? It means that you can run Java on your computer, we will get to the actual program to download the coding now. That is known as Java's Software Development Kit, it can be downloaded off the Oracle website (it is the company that owns Java). The one you want to use it called Java SE (Standard Edition). For the purposes of being able to code, we are going to do so using something called NetBeans, so once you find Java SE you will want to find an option that is called JDK 8 with NetBeans. You are also going to want to go to the NetBeans website and download software there. After you have gained access to the software, we are ready to code!

Essentially once you have all the correct software, you are going to begin to write your code in a text editor. If you choose a different option than NetBeans, you will find the place where you write the source code saved under the file extension .java. There is a program known as Javac that then turns source code into Java Byte Code. This process is called compiling, just FYI. After Javac has completed its task, it creates a new file with the extension of .class instead of .java. That is, if no errors are detected in your code—otherwise you will have to go back and find that probably small error you made and fix it. Finally, at this point, you can run your code on the Java Virtual Machine! In case it is more helpful for you, here are the three basic steps you will use to get your code running—first is to create your source code with the .java extension, then you will use Javac to create a file ending in .class and third you run the complied class.

Now that you have a greater understanding of how you will create and complete actual coding within Java, you should be able to begin the process of creating your own codes! Within the suggested NetBeans software, it is a great environment to teach yourself some of the ins and outs of Java. This chapter really serves as a basic description of the software and a very general overview of a way to download and begin to use it. Throughout the remainder of this book, I will talk more about some of the technicalities that help you understand how and why you must do certain things in Java (I.E. learn to speak the language fairly fluently, with the knowledge of how things work to back up your fluency of language).

Chapter 3- Conditions and Control Flow

In order to properly be able to utilize Java, it is important to understand the rules of the language, like grammar and spelling in the English language. In order to properly tell the computer what to do, we need to understand what the terms conditions and control flow really mean within the coding and programming world. For conditional statements, we can think of these as if/then statements that are used to tell the computer different actions to perform or computations to make. As for control flow, it is the order in which instructions, statements or function calls of a program are executed or evaluated. You can think of these as sort of the rules for sentence structure and the usage of a common, semi colon or other literary symbolically represented rule. These are very important in programming because computers, unlike humans, do not have deductive reasoning skills. They carry out the functions exactly as they are told to do so.

Both the conditions and control flow work together in order to create the proper environmental flow of data. Without either of these concepts, it would be difficult to have Java work in the user friendly manner that it does. A key component of a computer is that it is reliable as well as straightforward without multiple interpretations of the same thing. Computers do not read minds, they execute tasks as their function.

Conditions

If you have taken an English class in high school or college, you probably have heard the term conditional statement within that context. Although this is obviously a different context than an English classroom, the basic functionality is roughly the same. There are two different types of programming languages, functional and imperative, Java falls into the imperative category so we will be covering the specific conditions of this type of programming.

When we look at imperative programming languages, we see the term “conditional statement” typically used. These essentially are structures having one or more conditions that need to be executed if the statement or statements have determined to be true, some are also executed if they are false, depending on the statement. In Java we see four types of statements that are made. These four statements are if/else statements, if statements, switch statements or finally nested if statements. There are if statements that are a Boolean expression (Boolean is when there are two outcomes, like true or false). Whereas if/else statement is occurs when and if statement is followed by an else statement when it is executed in the code it will return the answer of false. A nested if statement is something that is two statements with one “nestled” inside of the other; these are either if or else statements. Finally a switch statement is one that takes a variable and tests it against a list of given variables.

These statements are all made up of questions that a computer can process and execute functions that are, at each of their essences, made up of Boolean (or two outcome) expressions—except for switch statements which compare one variable against a list (for example, what is the greatest number (maximum number) in a given list. Although multiple statements may be wrapped inside of another with the if statement format of any kind, it is important to remember that at their core

they are asking simple two answer questions. As I stated before, computers are incapable of inference and this really drives this point to home, these questions demand either of two responses, or it will create the third option of an error.

Next we will address more about control flow and how these statements are executed in the programming. It will also have some overlap with these conditional statements, as they are both methods of communicating (coding) what the computer should do and how it should do it.

Control Flow

As I stated earlier, control flow is the manner in which a given program executes or evaluates statements. As with any language, there are a lot of rules you have to follow in order to communicate properly to another person (in this case, machine). It is important to understand the reasoning behind these things in order to properly speak the language of the computer. Learning about control flow, even if we are specifically talking about Java, can easily be applied to other languages, although specifics may not be exactly the same. This is another part of building the foundation for your programming and coding future so it is important to make sure you understand it as well as possible.

Control flow statements are different in each different programming language, however they can be categorized by the effect they have. There are five unique categories that these statements can fall into. These five statements are 1) a set of statements executed only if a condition is met, 2) continuation of execution at a different statement, a set of statements executed zero or more times until a specific condition is met, a set of distance statements after they are executed the flow will return and finally 5) stopping of the program, this will prevent any further execution.

Now in Java, when we are dealing with something like a common loop control statement; specifically we are looking at the change a loop statement makes from its normal sequence. Java specifically supports two different types of control statements in this scenario, break statements and continue statements. Break statements are those that terminate the loop (also known as the aforementioned switch statement I discussed above) in which the actually execution moves on to whatever immediately follows the loop/switch. A continue statement is one that tells the program to essentially skip any code following it and retest the condition.

Essentially when we look at both conditions and control flow we see things that tell the computer what it should or should not to. Through programming we are able to give the computer specific instructions on how and why it should execute statements. Although these are really very brief definitions and descriptions of what and how both of these mechanisms operate, they give you an excellent foundation for knowing how they work.

Utilizing the use of statements like these is important work to figure out while you are learning Java. The benefits of Java are that these are routine functions and there is a lot of trial and error that can be easily done when you begin to experiment with coding and programming. These are important and key features you should certainly feel comfortable with before moving on to anything more advanced. At each step of your programming journey, make sure you are practicing and making sure you are firm in your ability and knowledge of what, how and why you are doing any procedure.

Chapter 4- Object Oriented Java

In this type of an object oriented language, which is what Java can be described as, you are looking at a program paradigm that is really based on the concept of “objects”. These objects can contain data, in the form of fields (often called attributes) and code that is in the form of procedures (often referred to as methods). One of the features of an object is that the object’s procedures have the ability to access and often modify the data fields of a given object they are associated with.

As Java is an object oriented language, there are fundamental concepts that it utilizes in order to fall into this class of language. The concepts of classes and objects are of particular importance and what we will discuss and analyze in this chapter. In terms of Java, an object has both states and behaviors. When we talk about class in this programming context, we are looking at the outline that allows for a description of the state or behavior that the object supports. Essentially, class is what you use to define what the object is. Now let’s get a little clearer understanding of what each of these things are, starting with objects.

When we think about “real world” objects, we probably think about things like dogs, other people or cars. Each of these objects we think of does have a state as well as a behavior, just as it does within Java. If we want to look at a human, for example, they have different states (names, skin colors, and ethnicity) and they also have behavior (speaking, working or sleeping). Each of these states and behaviors can be thought of as different parts of the same person. In fact, every person has a different set of behaviors and states, just as with most objects we interact with on a daily basis.

In comparison with real world objects, there can be very similar characteristics in some instances. The state of software object is stored within the fields and behavior is shown in the methods. So basically, methods are internally operating both the singular object and allowing the communication to happen via a method between objects. Just as with the human example, we have internal communication happening within ourselves and we use these internally stored methods to gain access to other objects (whether that be a car, computer or another human).

So what about class? It can be defined as the blueprint that individual objects are created from. There are three different types of variables that any class can contain. These variables are local variables, instance variables and class variables. Local variables are those that are localized, or within your method. They are initialized and declared locally—or within the method. When the method is completed, the variable will them be destroyed. Instance variables are those that inside of a class but actually are outside of methods. They occur when the class itself is instantiated. The instance variables have the ability to be accessed from any blocks or constructors of the class they are a part of. The last variable type, class variables, are those that, as the name suggests, are within the class. They operate outside of any method however they can use methods in order to access values.

What do we do if we would like to create an object from a class itself? There are actually three

different steps that will allow us to do just that. These steps are declaration, initialization and instantiation. A variable declaration is when a variable has both an object type with a name for the variable. An instantiation is the new keyword that you would use to create a given object. Finally initialization is what is used to call a given constructor and has the ability to initialize the new object. For example, you could create a person (object) in the human class and then name the puppy Spot, giving name to the constructor and creating a new object. Let me give you a little more clarification on constructors in the next paragraph.

When we look more into classes, we have to also talk about constructors. Constructors are a bit of code allowing you to create a given object from a class. Constructors are a necessity in every class. If you do not include one, the Java complier actually creates a default constructor for the class at hand. You can always be certain that if you create an object, one or more constructors will be invoked for that object. Technically constructors should really have a name that is the same as the class they are in; however, classes can have more than one given constructor. For example, if you had a class known as puppy, the constructors could be lab or bulldog—or like the example in the previous paragraph of a name like Spot. Now let us talk a little bit about some rules that are needed to be followed when you are declaring classes.

Source File Declaration Rules

These source file declaration rules are actually essential when you are doing things with your different source file statements or import statements. They are also important when you are declaring given classes (which is a Java class file with the .class extension).

- You can only have one public class per source file
- The package statement should always appear first inside of a package
- One source file is able to have multiple nonpublic classes
- When dealing with import statements, you always need to remember that statements are written sandwiched between both the class declaration and the package statement. If you do not have any package statement the very first line in the given source file should be the import statement
- If you have both source files and public class files you should have them both be the same name. They should also be appended by .java at the end of the file. Example-administrator.java
- When you have both import and package statements, the classes they are associate with will be applicable fully in any source file that you are using. You cannot have source files that contain package and import statements associated with different classes within the source file

Chapter 5- Data Structures

Within Java there are certain types of data that are used and defined by the program. I am sure that, in some form or another, you have heard of data—it could be the gigabytes of megabytes of data used by your cell phone or maybe you have a backup for your computer that stores data in terabytes. Each of these different terms tells you how much data your device or backup can support and use.

When it comes to data itself, there are many different ways we use this terminology both inside the virtual world and in the real world. When we look at data, I personally think of it as something that gives me quantifiable means to look into or solve a problem or question I may have. Within the world of Java, data is just defined in its own computer programming and coding language manner.

In terms of the type of data Java can support, it is much vaster than just the typical “byte” data you have heard in the past. Java actually can support eight different primitive data types. These data types are byte, short, integer (referred to as int), long, float, double, Boolean and char. Now that you have heard these types of data, let us break them down and tell you what they mean and what they can do for a user.

Byte

A byte is what can be considered a unit of memory size that is actually a group of bits, or binary units, typically eight that is operated on as a unit. What does this mean for your usage of Java? Well, within Java it has a minimum value or -128 and a maximum, inclusive value of 127, as well as has a default value of zero. This type of byte data is actually used in Java to save space in large arrays—most of the time in the place of integers as a byte is actually four times smaller than an integer. Typically you will see byte represented in a similar fashion to this—ex -100 byte.

Short

Where byte data requires full integers, you can use short data to contain integer values that do not actually need or require the entire width of integer. Short data can, in some cases, pack your short variables closely together in an effort to save memory consumption. Within Java, short data type is 16 bit data with a minimum values of -32.768 or maximum value of 32,767 also with a default value of 0. These amounts are actually two simple equations—(-2^15 and 2^15 -1 respectively). This type of data is also good for saving memory as a byte type data we spoke about in the previous byte section. That being said, short data can be used in an effort to save memory as the byte type data as it is two times smaller than an integer. A couple of examples of how you may see short data are short s= 20000 or short r = -100000.

Int

This type of data, in long, unabbreviated form, is actually referred to ask integer data. The definition of integer data is that it is a very small subset of mathematical integers that can be seen as a group of what is known as binary digits (often referred to as bits). Computer hardware

always wants to have a way for the user to represent data (like memory address or processor regester) as an integer In terms of its uses in Java, integer or int data is a 32 bit signed two's complement integer with a minimum value of -2,147,483,648 and a maximum, inclusive value of 2,147,483,647 and has a default value of 0.These can be represented by the equations -2^31 and 2^31 -1 respectively. Integer data is the most commonly used data type for integer values, in fact it is often used as the default type of data. If there is a concern about memory, however, you may not want to use this method (as you can see the minimums and maximums are substantially larger than the previous two types of data). You will also see this data represented in a format such as int a= 200000 and int b = -100000.

Long

As you would gather from its namesake, long data is in fact very long in the minimum and maximum values it utilizes. The minimum data value for long data is -9,223,372,036,854,775,808 and the maximum is 9,223,372,036,854,775,807 and is inclusive with a default value of 0L. These are represented by the equations -2^63 and 2^63 -1. Long data is used when a much wider range than integer data is need. You will see long form data like these examples—long a = 200000L or long b=-100000L.

Float

With Float data we are looking at a single prevision 32 bit IEEE 754 floating point. When you want to save memory, you may want to look into this option to be able to save larger data amounts. The default value for this data is 0.0f and this type of data is never used for values that are considered precise, such as currency. You will see it represents like float f1= 345.6f

Double

Double data is a type of double precision, 64 but IEEE 754 floating point data. This type of data is the default used for storing decimal values. Always remember that these types of data should not be used when you are dealing with precise values, or whole integers. We have a default value of 0.0d for this type of data, example—double d1= 567.8.

Boolean

We talked about Boolean data briefly before. Boolean data is the type of data that only has two different outcomes, true or false. It is used in order to find and track both true and false conditions. Always remember that, unlike you may be inclined to think, the actual default value in this data set, rather than some form of zero, is false. This makes sense when you look at true and false as the only possible outcomes. You would see this type of data represented in an example like Boolean one =true.

Char

This char, or character, type of data is a single, 16 bit Unicode character. This data is represented by a minimum value of '/u0000 (or 0) and the maximum value is '/uffff (or 65,535) and the maximum value is inclusive. This type of data is used to store any character. An example of how you would see this data is char letterB ="B"

Now at the beginning of talking about data, I wrote that these are primitive types of data. Something that may be helpful to you when thinking of data is referring to our previous content above about classes that is actually non primitive data. However, they are both ways of representing something in our coding and programming that are ways in which we categorize and store materials—or data. Data in any sense is something that we gather and organize in order to properly explain or store for future use. Although these various types of data may seem somewhat obscure by just reading these definitions, once you start experimenting and seeing how various algorithms or coding is developed in your own usage, you will find reasons for each of their uses.

Reference Data

Within the world of Java data, we will also see what is referred to as reference datatypes. We can view reference variables as those that are created when actually utilizing constructors of the given class or classes to define them. These types of data are used as a method in which to actually access objects. Reference variables are those that are not able to be changed as a point of definiton (for example—puppy or employee). There are some variables and class objects that actually have a reference variable without value (or null). Using these you can refer to objects that can be compared with each other. An example of this is Animal = new animal(“cat”).

Any primitive type data can be seen as a literal. Source code that is literal is something that has a fixed value (or unchangeable). Basically, you cannot have anything within the value other than the code, having no computation. These literals can be assigned to any primitive type of variable, example char d = “D”. When you think of literal, it somewhat means a taking at word at the base sense (most literal value). There are also what is called String and char literals that support few special escape sequences that are good to note. These are things like /n, which means newline, /b or backspace, /s or space, // which is backlash, /t or tab, /’ which is single quote, /” or double quote, /r or carriage return, /f or formfeed and finally /ddd or octal character. These special escape sequences may come in handy for you in the future. Take reference of them for now and there will be times in your programming that they may be useful.

The beginning of this chapter really helped to explain data types, now we are going to switch gears and talk about variables.

Variables

As with in any math based activity, you can manipulate the variables in Java in order to name specific variables in a helpful way. Each variable does have different sets of “rules” that are associated with it that can be applied to the variable at hand. Within Java, you must declare a variable before it is used. There are the three different types of variables we can use in order to give proper categories of organization for the variables themselves. In the previous chapter, we discussed variables in some detail. The reason for addition an additional section in this chapter is to bring to your attention how they are related to the data and data structures. It is important to remember that there are other forms of data than the memory we assign to them to let us know how much room we need for them.

Chapter 6- Helpful Hints and Resources

Helpful Hints

When it comes to hints, I am going to refer to those that would be used within the NetBeans software I recommended you download in order to run Java in the beginning of the book. There are 29 helpful hints in using the program—however, if you have chosen a different software than NetBeans, you can access more helpful hints that I will go into more detail in the resources section following this. I will break the 29 hints down into short categories that can be used as a helpful reference guide if you run into trouble.

These different hints can be found in greater detail on the NetBeans website, in case any of the terms or suggestions are confusion I suggest you head there for more clarification—or even a quick Google search on a term could help as well! There are a good deal of terms and explanations below, be sure to not be overwhelmed by the vast amount of information, start coding and programming within NetBeans to understand more about how these tips can be of assistance in your programming and coding. It can be difficult to fully grasp them without the software working in front of you!

Before we continue with the hints and resources I wanted to point out one last philosophy that has served me well—which is try to aim for the middle ground and not allow yourself to get too caught up in details or sloppily rush through and miss potentially important ones. As you learn and grow within your skill set, you will begin to understand the somewhat subtle differences that can make your coding and programming far more understandable to both you and the computer.

Abstraction

Sometimes when you are dealing with type cast it may actually be too strict and if you cast to a more general type that would be sufficient for the actual expression. There are instances casted to a specific subtype— however methods invoked or the fields used from the casted instance that is defined by a super type. This too specific vast introduces really unnecessary coupling to the code, limiting its extensibility.

APIs

The API technologies will be made option in Java EE 7, they may be available now but in the future the options may not be present—essentially remember that technology is ever changing and never rely too heavily on newly introduced options. Also make sure that return and parameter types of all public methods on any type of public fields are available to anyone to use. Basically, many people may be using these same things so just keep that in mind for security purposes—not something you need to worry about if you are just building and coding in a sandbox environment.

Assignment Issues

When you assign operations, always remember that you could code using operator-assignment. Theoretically, it may be more clear if you do use the operator assignment. You can also assign to a variable declared as a catch block parameter, however this may be confusing when just starting

out. Loop parameters reports to a given and declared variable for a statement within the statement (as we spoke of briefly earlier). This could be confusing when it occurs accidently, which often happens as a result of a typo. It is good to get familiar with what and how that operates and looks like in case you should have an issue in the future. As with the other nested assignments, they can also be confusing as they include a separate statement within a statement. Make sure you are aware of the expressions and how they work as they could violate some general design principles of your coding—as most constructs should do just one thing.

Bitwise Operations
Within bitwise operations we see things like incompatible masks that allow for you to know that you have create a bitwise expression that is guaranteed to be evaluated false. There are also inspections that report that there are instances of pointless bitwise expressions that have to deal with scoring by zero. Also we will see an inspect report that states values have shifted and it is both constant and outside the reasonable range.

Braces
When to use the braces symbols are for things like do while loops and for loops. They are the proper time to use braces. Along with if else statements, you should always use braces within these types of statements.

Class Structure
Class structure is something that is a data method we will certainly always see and has the potential need for organization and firm definitions. Within this section it can be important to remember that a class may be an interface, and that is ok. Also there will be reports on things like the final class, method and the final private method. The initializer may actually not access instance variables or methods and could just execute once—not during each individual instance creation—remember that to ensure you redo it if needed. You also have the ability to do things like organize member or even protect them in their final class. We also look at constructors that are both nonpublic and public as a factor.

Code Maturity Issues
We see code maturity issues when the code we use may be outdated or obsolete. You may see something like obsolete collection or system out errors and other efforts to debug and remove outdated and unusable code.

Empty Statements
These empty statement type of programs run and check for empty statements of all kinds, including after words like do, for, if/else or even while. This allows for another check to make sure that you will not have an error in your coding due to a simple error.

Encapsulation
This is a large field that is designed to find out more about what you have incorporated or encapsulated and protected within your code. This is going to be things such as warnings about assignments of different data, warnings about package fields or protecting inner class. Also it will

look at protected and public data to warn you of anything publically seen within these categories.

Enterprise Java Beans
All of these types of tips have to do with keeping everything in line with the Enterprise Java Beans through their own system of alerting you to things like uncomplete sessions or method issues. Or even things like local and remote interface together or just a lack of usage of certain constructs. It can be very helpful, as all of these tips are, to be aware of to check that everything is running smoothly.

Error Fixes
These are things like finding missing on project class paths of Jave EE API. Or that you can access things like creating a field or local variable. These all help you to find and fix errors that you may not be aware of.

Finalization
These finalization tips have to do with getting warnings about the finalization tasks your computer may need to go through. These are going to be things that say thing like finalize declared, does not call super.finalize, not declared protected finalizations or even an explicit calling of a finalize. It is just as important to know how to end things as it is to be able to start them. If you have holes in your finalization, potential malicious things could happen to your system (as with holes in any part of your coding or programming structure).

General
These general functions will assist you with removing redundant content and making sure you are using functional operations. These general functions will also warn you of a general wrong package or other fairly easy to spot issue as well as easy to fix. These are probably some of the more beginner friendly type of tips and tricks to deal with in the beginning stages of your Java learning experience.

Imports
With imports you have quite a few options that can be utilized within the import genre of instructions. You can import from the same package, or from java language package or even from classified or excluded packages. You can also use a tool to organize your imports as well as add things like starring or unused imports to help categorize them.

Initialization
Initialization tips and tricks are to be aware of certain key variables that could possibly get in the way of not fully initialized things. This couple be passing suspicious parameters within the constructor, or even a problematic call in the constructor itself and finally a static non find variable that is used during initialization. All of these could potentially cause problems or issues for you if not addressed.

JDK Migration Support
Within this category, we are looking at ways in which you will get warnings of potential issues

that you should be aware of. This s things like not using annotation as a super interface, converting different resources like finally block to try with in an attempt to figure out a given problem. There is an @Override annotation to warn you of an override situation. There are many similar type of warnings and issues that come up within all of the programming tools. In this category, the terms are very explanatory and will essentially tell you how to fix the problem within the message itself.

Java Code Metrics
Java code metrics are a series of potential alerts to things that are going wrong within your code. This could be anything from classes having too many instructors, fields or methods or even that the class itself is too complex or coupled. Remember that you can get overly complex with your coding at any point and the system may not be able to properly process your data because of it. You will also see words about anonymous classes having too many methods or complexities as well as arithmetic expressions.

Java Persistence API
You will use Java Persistence API really to check and validate different classes and such. There are lots of phrases that, in this particular section, will start with verify and have you figure out things like finding location or subclass and verifying access levels or other attributes. You also can do things with the entity table names or serializable variables that an entity has. There is also a need for instructions on what classes need ot have default public/protected constructors.

JavaDoc
Two good terms to know for JavaDoc are create JavaDoc or error in JavaDoc. These are the basic two phrases that allow you to create something in JavaDoc or know and be aware of an error you need to look into.

JavaServer Faces
Instead of using JavaServer faces, as it will be depreciated in the next version, it is recommended that you use CDI and Java EE instead.

Logging
You have probably logged in and out through various instances on computers. Within Java, you will see phrases like multiple loggers, no loggers or logger declaration is not static final. These are all class based things you are alerted to just either for your own information, or to change something about the logging process you have created.

Maven
This is really used to resolce a missing class dependency. It alerts you to add to projects.

NetBeans Development
There are terms you may see in NetBeans that allow you to see errors or issues with the work you are doing. These may be phrases like empty cancel for cancelable tasks or HelpCtx issues (these warn of misuses or .org.openide.util.HelpCtx) or instances of illegal uses of instanceOf operator.

Using the phrase @NbBundle.Messages to bundle properties. All of these can and should be explored further within NetBeans itself. As we did not cover the ins and outs of NetBeans, there are lots of opportunities to learn more in their libraries.

Performance
There are lots of performance tips that can assist you with you improving your coding methods. There are things like creating new Boolean phrases and boxing an already boxed value that will decrease your performance to some extent and are really not cost effective, time wise. There are also methods that do not allow for optimal performance when you begin to call things like String.toString. There is no use for reinventing the wheel most of the time, and you can get yourself into trouble. It is usually most efficient to just utilize String and not try to overcomplicated it—especially when you are learning.

Probable Bugs
There is a very long list or probable bugs, so many in fact I am just going to list what they are, as you can infer from that what their purpose is. There are finally blocks that suppressed exceptions, you can also throw inside a finally block or even @CheckReturnValue. There are also .equals on array, incompatible types and null. There are annotations without runtime retention, assertions that have side effects or what is called a boxed value identify comparison. You also may run into clone able class that does not implement said clone or a compare that does not use its parameter. If you have confusing indentation (spacing) this could also present a problem for you. There is also sometimes confusing primitive data array passed on to what is called a varargs method or if you have a hint of something throw able (instead of thrown) it will be discarded. There are also a series of bugs that have to do with malformed data types that can cause a red flag to these to get you to address, and hopefully fix them. This is a pretty good list of potential bugs you may run into. If there is something else, or you need further clarifications on what these mean to fix them, I encourage you to look them up in another resource, like Google.

Standard Javac Warnings
These are general warnings you may see when you are using Java. These are things like deprecated (warning when code uses depreciated or old API), you have had a division by zero, empty if statement or things like a fallthrough. There are also warnings of when a finally block interrupts the flow or an override warning that is not properly annotated by the @Overrrides expression. There are also what is called raw types, serialization, unchecked or unnecessary casts that warn you when different things are occurring that need your attention.

Suggestions
There are lots of suggestions I think you should be able to try out in order to see what and how you can manipulate language in Java. These are quite a few things, and span a lot of different areas. I encourage you to look into how to assign and return a new variable instead, or to assign unused constructor parameters to a given field. You also have options to combine tow nested if statements, if this is something you find appropriate to your causes. There are also ways to convert some anonymous to a member, covert what's called a lambda body to use a block or express to anonymous interclass or even member reference. You can also convert lambda

expressions to member reference or use it to explicit parameter types. There are also options to convert integer constants to a different base or even create sub classes or test classes for a selected source.

You could also look into how to do things like fill missing cases to switch or even expand enhanced for loops or a declaration of instance of. You can also tell Java to invert if… and join ifs into if else. These are all helpful tools and suggestions that are able to assist you with developing your knowledge of just what Java is capable of.

Testing
With testing you want to look out for inconvertible parameters of Assert.assertEquals, incorrect order of parameters of Assert.assertEquals or assertEquals for any other parameter. Basically these are functions you can use to discover things about the parameters of what you are doing.

Threading
Here are some different types of statements you could use if you would like to thread in any particular circumstance. These can be things you may need to use in the future like invoking a condition, invoking thread, locking, and nested synchronized blocks, double checked locking, notify or wait. Basically all of these .notify or .wait etc. type of extensions do different things within your programming that allow you to so some more complex functions.

I want to reiterate again that all of these categories may not be something you are ready to have complete knowledge of at this time. You are learning and definitely take your own comfort level into consideration with what you feel is a reasonable scope. I encourage you to really study the material in the first few chapters and gain a great understanding as to the basic ins and outs of Java.

Resources

There are literally thousands of options when it comes to resources to assist in your learning process, which can be overwhelming at first. Sometimes it is hard to know exactly what will be helpful to you. The first piece of advice I would like to give is that you should aim to find resources that address your specific problem. It is important to see the problem, or answer, in the title of the book or article you are reading. If you use this simple strategy, I guarantee that you will have a much easier time answering the questions you may have. This isn’t a fool proof strategy but it is a good place to start.

There are a lot of step by step tutorials that you may want to look at if you get caught up in a programming rut. These tutorials can provide you with steps that breakdown how and why you should use certain terminology, spacing or even data types. They can easily be found by searching for something like a basic Java walkthrough for beginners. In case any of the content in this book is not quite as step by step as you need, there is no shame in needing more help and finding

something that will help clarify or cement ideas for you.

We also see lots of people getting help through forums. These are great places to access because you literally have thousands of people putting their input into whatever topic you are researching. Most of the time, within the programming and coding community, people generally want to help newbies because they know what it was like to once be in your shoes. Remember that not all advice or coding examples may be helpful to you, stick to beginner forums at this point in order to not overwhelm yourself with the amount of possibilities out there.

There are also a ton of articles or magazines you could look to or subscribe to in order to assist you with building your knowledge. I think a lot of people disregard actual printed media, because we are in such a digital age, however there are still many popular magazines out there that can provide up to data, useful tips and suggestions for you. Among the most popular of these magazines are Wired and PC World that I have found to be helpful in my quest for knowledge. The generally write at a level that any beginner can understand and, if they do not, they will provide resources for you to understand. Which is a great thing as media is widespread and these magazines are meant to be informative to those of all walks of life interested in computers and coding and programming. Do not discount online articles from major publishers either, you never know where and when there will be useful information.

Also, never discount the power of things like libraries or code editors. Each of these types of tools can assist you with finding out code you may want to use and want to make sure will not be riddled fill of errors. These resources provide you with a plethora of useable code that can assist you with any potential trouble source you may run into with your coding and programming. I do caution, however, that you fully understand the code, in case there are any issues with it you can figure out where and how to address them.

Another great place to find resources in YouTube videos. These videos are ways for you to watch programming examples in more of a real life scenario that may be of great assistance to you. Although it is great to read and have the information, the beauty of a video is that you can visually see what is expected of you and how another person does it. No matter how hard I try in this book, I am unable to give you that kind of demonstration, so that is a great place to utilize YouTube. When it comes to the quality of videos, just like with finding any resource, it can be a bit of sifting through unhelpful videos in order to find ones that you can actually use. My advice to you is similar to that of a search on Google or something, make your search description as close to the content you would like to see. This enables the results to, hopefully, contain those words together. Think of the search function in terms of the statements we have talked about, you need to specify the data you want in order for the computer to return the results you want. Once again, remember that computers do not infer meaning like humans can—they are straightforward to a fault.

Remember also to use your both your own brain and perhaps friends that also are interested in computer programming and coding in order to brainstorm solutions to potential issues. Never discount yourself and your own ability to look for answers in simply trying out different solutions within your own sandbox to play with coding and programming in. If at first you don’t succeed, try again!

Conclusion

The aim of this book was to give you a how to guide for what and how Java programming is and how you can learn to apply it in your own computer programming skill tool box. Although this is a beginner's guide, I have given you a great general set of terms, rules and examples of how Java works. Through the pages of this book, I hope you have been able to get easy to understand terminology, ideas and methodologies you will be able to take with you as your continue to learn more.

From the first chapter where I outlined the basics of computer programming, the goal of my book was to make the material something anyone could understand. The aim of my writing was to ensure that even a total novice could understand the basics and be able to (even if you never use them) understand a little more about how computers actually function. As technology, and computers in particular, are a huge part of our daily lives, I feel strongly that we should understand them more to make our own lives easier. When it comes to programming and coding, it may not be something everyone has an affinity to as a hobby—but it really is quite enjoyable once you begin to speak the language.

As with learning anything new, particularly a language, there is always a learning curve. If you have read this book and still feel weary of the programming world, maybe take a few days and just reflect on the content and come back and read it again. I always use this tactic and it typically makes more sense if you let the content marinate for a while. For those of you that have fully grasped the content on the first go round, more power to you!

It is always important to remember that you cannot remember everything, all the time, and there are great resources out there for you to refer to when you may not be able to recollect something. I hope that this will be a book you will come back to in the future to get a greater understanding of the beginner concepts we can sometimes forget as we get into more advanced programming techniques or situations.