JAVA

STEP BY STEP
PROGRAMMING
GUIDE

A DETAILED APPROACH TO PRACTICAL CODING

NATHAN CLARK

Java

A Detailed Approach to Practical Coding

Nathan Clark

Table of Contents

Important Notice

Introduction

Getting Started with Java

The Data Types of Java

Reference versus Primitive Types

Making Some Classes in Java

Control Flow and Conditional Statements

Using the If and Switch Statements

Working with Inheritances

Some Other Things You Can Do with Java

Conclusion

About the Author

This document is presented with the desire to provide reliable, quality information about the topic in question and the facts discussed within. This eBook is sold under the assumption that neither the author nor the publisher should be asked to provide the services discussed within. If any discussion, professional or legal, is otherwise required a proper professional should be consulted.

This Declaration was held acceptable and equally approved by the Committee of Publishers and Associations as well as the American Bar Association.

Any information contained in the following pages is considered accurate and truthful and that any liability through inattention or by any use or misuse of the topics discussed within falls solely on the reader. There are no cases in which the Publisher of this work can be held responsible or be asked to provide reparations for any loss of monetary gain or other damages which may be caused by following the presented information in any way shape or form.

The following information is presented purely for informative purposes and is therefore considered universal. The information presented within is done so without a contract or any other type of assurance as to its quality or validity.

Any trademarks which are used are done so without consent, and any use of the same does not imply consent or permission was gained from the owner. Any trademarks or brands found within are purely used for clarification purposes, and no owners are in anyway affiliated with this work.

Important Notice

Please note that this is the 2nd book in the Step-By-Step Java Series. This book can be read as a standalone book, but to get the most value out of this series it is highly recommend to first read

Java: Programming Basics for Absolute Beginners (Nathan Clark)

Introduction

Congratulations on purchasing *Java: A Detailed Approach to Practical Coding* and thank you for doing so.

The following chapters will discuss everything that you need to know to get started on your journey to using Java as your coding language. There are many different languages that you will be able to choose, but Java is one of the best to use when you are trying to put together a webpage, no matter what kind of page you are putting together. It will help you to make headers, write paragraphs, and can be used to add in links and pictures when needed.

This guidebook is going to show you how to design some of your own pages, as well as some of the cool things that you can do when you are using this programming language. We assume that you have a bit of experience with Java in the past, perhaps just getting to use it to write some basic codes, and then will move into doing some of the more complex codes and options inside this program.

This guidebook will help you to write some of your very own code and can even make it possible to write out your own webpage if you choose. While there are so many great things that you will be able to add into your coding, this guidebook is going to show you how to do some of the best things. Such as writing down the information, adding in links to other pages you write or other websites, adding in forms, and so much more.

Whether you are just getting started with writing out some of the code that you will need for this, or you have experience with coding but want to get into the world of using Java to write out your own websites, this guidebook is going to be the tool that you need to get started.

There are plenty of books on this subject on the market, thanks again for choosing this one! Every effort was made to ensure it is full of as much useful information as possible, please enjoy!

Getting Started with Java

While there are so many things that you can do with the Java programming language, the first thing that you should understand is that it is an object-oriented programming language. This basically means that it is easier to understand; many of the newer programming languages switched over to OOP because it is easier to use and can prevent mix-ups and other issues that occur when you are working in your coding. Java is also general purpose because software developers wanted a programming language that they could create once and then have run on many different platforms.

This program uses some of the same features of C++ and C#, but they have been changed a bit to make it easier to use without having to compromise on the power and speed of your programming language. Java is all about keeping things simple and being able to be used on a variety of different platforms, making it the perfect coding language to use.

Some of the Basics of Java

First, let's take a look at what Java is all about. Java is a programming language that was developed to help create various applications on your computer. It is not uncommon for a user to download programs that will require them to use Java runtime and you probably already have this on your computer. It is also possible to use this as a plug-in on the computer so that you can run these apps in the browser.

It is important to realize that Java and JavaScript are not the same things and outside of the names, they aren't that similar. For example, JavaScript is going to be used with web pages that will work with HTML documents, rather than for standalone apps that can work inside of the web pages. This is sometimes confusing since Java is still able to run inside the browser, but there are some differences. First, you will notice that many types of websites are going to use JavaScript while Java is a bit more limited, even though it is easy to use on many websites.

It is free to download the Java program on your computer and it doesn't take too long. In fact, there are a number of systems that already have this downloaded on them to make things easier. But there are a few simple steps that you should take to get Java downloaded to a computer that doesn't have this coding language on it already. Some of the things that you will need to do to install this program include:

- Install the JDK—you will need to have this in order to use the tools that will develop your application in Java. You will just need to install this at the same time as Java, and it is quick and easy.
- Choose your IDE—this is where you are going to be working with Java. It will allow you to build your codes and see how they will show up. There are a few that you are able to choose, but for beginners, NetBeans is a great option to help you out.
- Once all of these are done, you will be able to get started with writing your programs. These are easy to install, will just take a few minutes, and you will not have to worry about them taking up too much space within your program either. Starting with a Hello World program is one of the best to use if you want to get a feel for the language as well as for the environment that you have chosen.

And that is all that you need to work with when it comes to setting up your Java program. Once the IDE is set up, and you have Java plus all of the tools that will make it work well, you will be able to work with any of the codes that you want to make your web pages amazing.

What are the Benefits?

There are quite a few benefits that you can get when you go with Java as your programming language including:

It is portable

The developers of Java wanted to make sure that Java would work on as many of the platforms as it could. This would make it easier for the developers who used Java because they were able to write the code once and then it would work on any platform that reads the Java code.

It is safe and secure

In our modern world, safety and security are important. You need to make sure that while creating a new program that the user is going to know their information is safe and won't be intercepted. Java is encrypted to ensure that all information is safe.

People can connect quickly

Java is designed to use simple wording and phrases so that the developer is able to connect with it quickly. This makes it easier for them to learn the language, even though it may look difficult in the beginning, and they will be writing their codes in no time.

There are a lot of features

You will find that the Java code is going to have a lot of the features that you are looking for. It is easy to make modifications to when you need, it can work with threads, and it is one of the dynamic programming languages.

It should work well

No one wants to work hard on a code and then find out that it is slow or that when you try to execute it, there is something wrong. Java is a really precise code that is easy to use, and you will get the results that you want each time.

Because of all these great benefits, many developers, both those who had been working in coding for some time and those who just wanted to get started for the first time, choose to go with Java for their coding needs. It is easy to use, it is precise, and the coding language is not one that is difficult to learn and understand.

Drawbacks of Java

There are a few drawbacks that do come with Java, which is why some people will choose to go with another coding language in order to get their work done. Some of the issues that you are going to find include:

Applet limitations

These are there because of the current security model that is placed on Java. You are going to have trouble getting local data. You are only able to get the data from the site where the applet is located, and you can't make system calls. You can bring in other applications to make this a bit easier to handle.

Slower

This program is a bit slower than some of the other coding languages, but there have been some changes done to the later versions that help to speed this up a bit. It is a little less optimized and will take more time to compile some of the larger programs compared to other options.

Takes up more memory

Unlike some of the other programming languages such as C++ and C, which can take a little bit more time to get the apps to download.

It is just a single paradigm language

This is mainly a single paradigm language, although some of the newer versions are changing over to the procedural paradigm to make this issue go away a bit.

Overall, Java is a great program that you are going to be able to use in order to get the code written. You are going to love how easy it is to use this program and all the great codes that you are going to be able to write to make your website look amazing!

Performance

In the beginning, Java was a slower program, mostly because the programs that were created to go with it needed more memory than those that were developed for the other programming languages, such as C+. However, the makers of Java have worked on this in order to improve the speed and with it becoming an open sourced language, there are more improvements being made to this programming language all the time.

Syntax

The syntax is Java is important to making sure that the interpreter is going to understand what you want it to work on. The first thing to keep in mind is that when you write out a file in this language, you will need to save it as .java and the child pages should be saved as .class. If you would like to make sure that the child page or any of the classes are private, you will simply need to save that in the same Java folder and it is all good. If you save it in its own folder, other will have some access to it.

So how do you decide if you want the child page to be accessed? If the program is still in the developmental stages, you will want to keep it as private so that you are able to keep working on it, you will probably need to keep it private. But once you are ready for the program to be used by other people and users, you should consider moving the child page out of private so that the users are able to access it.

Some examples of the lines that you would use with this Java programming language include:

Program HiWorld (

Public static.void (

System out PrintIn.HiWorld

)

)

Memory

In the beginning, the memory in Java was quite a bit slower, especially when it was compared to some of the other programming languages that were there. Issues with memory, in the beginning, made the program take longer to load for the user and it was not that user-friendly on some platforms. This is because the Java language used memory a bit differently than some of the others, but this has been changed in the later versions to make it easier to get the speed and the memory function that you need.

Over time, the developers of Java made some changes so that Java was able to work faster. This is due in part because it uses the Automatic Memory Management. This basically allows the programmer to choose which changes they are going to make to the code and their runtime is not going to be affected at all.

This kind of memory changes things so that it is no longer referencing a certain object. That object is now not a part of the program, and this can save a lot of the memory that you need to make the program easier to manage and a lot faster. You may not need to know all the specifications about the memory on Java, but it is nice to know that you will be able to work on your code and get it to come out nice and fast.

Now that you know a little bit more about Java and how the programming language works, it is time to learn more about some of the things that you are able to do with this language to make the right code that you are going to love and enjoy.

The Data Types of Java

When it comes to working in Java, there are a few different data types that you are going to need to learn. These are going to ensure that the program is going to work properly, that the interpreter is able to take a look at the code and be able to execute it properly, and knowing how each of the data types works ensures that you are writing the code right. Let’s take a look at the original data types that are in the Java programming.

Characters

The first data type that we are going to discuss is characters. These are just simply the words that you will use when you create your code for others to use. These can include a lot of different things, but some of the ones that you may want to learn include:

- If—this is going to work on a true and false basis. You can also use the if…else and the else if statements based on what you want to get done with your block of code.
- While—this is often used for true conditions or to execute out blocks of code.
- For—this is used for blocks that are attached.
- Try—this is handled by except and will make it easier to raise exceptions for your code blocks.
- With—this will use your block managers and can be used to help out with opening and closing files.
- Def—this means that your function or your method will be defined in the code.
- Class—this is going to be used in object-oriented programming and will execute the blocks of code.
- Print—you can use this when you want the phrase to be printed on the screen. For example, after someone puts in their age or a number, you can print out some kind of phrase to tell them they are right or wrong.
- Import—if you are trying to import some modules from your other functions, the import can be nice.
- Yield—this is going to be used as an operator and it is used to generate function values.
- Assert—this is often used to check out the statements that you want to have applied.
- Pass—this will allow you to create a code block that is empty.

If you are using these words, you will not be able to change them into variables. They are reserved to be used as commands to the program so that it knows what you want it to accomplish when the code is executed.

Lines and Indentations

The spaces and the words that you are using in your program will determine whether the code is going to show up the way that you would like. This is a program where the spaces are going to matter so you will need to take some time to make sure that everything is lined up the way that it should be. Let's take a look at two examples, the first one which is right in the Java language and the second that is not right.

Example 1:

```
if True:
  print "True"
else:
  print "False"
```

Example 2:

```
if True:
  print "Answer"
  print "True"
else:
  print "Answer
  print "False"
```

Quotations

Quotations can be used within the Java language if you would like and all types of quotations are allowed within this language. You will be able to choose between using a single, a double, and a triple quotation mark depending on what you want to write within the code.

Keep in mind that while any of them can be used, make sure that you are using the same type of quotation marks at the beginning of the phrase as you do at the end. This means that if you used the double quotes at the beginning of your phrase, you would need to use those same double quotes at the end of your phrase.

Multiple Line Statements

When you are using Java, when you start out a new line, you are at the end of a command. But what are you going to do if the line is going to take over more than one line? Is the program going to recognize that you are trying to get started with a new line or is it going to realize that you want to make this into just one command?

If you want to make a command that is going to take over more than one line, you will simply need to add in the backslash (\) to make it happen. If you are using brackets, this is not going to be expected to use this symbol, but if there are no brackets being used, you will need to make sure that the same command that is going over a few lines is going to use that backslash to help the program know what is going on.

Comments

There are times when you will want to leave a comment in your code. You may want to explain in more depth what is going on in the code or tell the user what they are supposed to do with the code to make it work. If you do these with the right signs, you will be able to get the comment into the code without having the interpreter read the comment.

The comment is not really adding anything to the code. It shouldn't be read in the code or you may end up with problems, so you need to make sure that you are using the right symbols to ensure that the interpreter will just skip over the comment. This allows the user to see what is going on in the code but keeps the comment from making issues within your code.

When it comes to doing Java, you will need to use the hashtag symbol when you want to leave a comment. Simply place the hashtag right at the beginning of your comment and then write out what you want to have happen. It is that simple. A good example here would be:

#input your age below

Then the user would know that they need to add in their age in the field that they see on the program. This is simple, and you will be able to use as many of the comments as you want in each of your codes. Just make sure that you are keeping them just to the amount that you need and that you start each of them on different lines to keep the code looking nice.

Boolean Expressions

The next thing that you are going to need to work on with Java is your Boolean Logic. While there may be a lot of options that are available when you are writing your code, Java, as well as some of the other programming languages, are going to simplify it down to statements that are either false or true. And they will also go with the Binary Numbering System. This is going to work to ensure that the program is going to make some of its own decisions based on the information that the user gives it.

The Boolean expressions are going to be based on true or false. For example, if you only want someone who enters their age to be above 21, you would have a true show up when the person enters any number above 21, but a false any time they enter a number that is 20 or below. If the true expression comes up, you will have a certain statement come up, such as “Congratulations, you qualify to participate.”

Now, depending on which way you do this, there are a few things that can happen. In the simplest of forms, you will have the program just end if the statement ends up false, or the person enters an age that is 20 or below. But you can go through and make some changes so that instead of having the program just shut down when a false occurs, another message comes up. For example, when the person enters a number that is 20 or below, you can make it so that the program will show a statement such as “Sorry, you are not able to participate at this time.”

Remember that you are the one who is in charge when it comes to the code that you are writing. There are a lot of expressions that you are able to use that fit into the Boolean expression, such as the “if”, “if else” and “else if” statements that allow you to create the exact code that you are looking for no matter what program you are working with. You will be able to determine a few different true and false statements that will work the best with the code that you are making.

Floating Point and Integer Numbers

When you are working in your code, you can expect that floating point and integer numbers are going to be used in order to store numerical values in your code. They can also be grouped together. There are a few different numerical types that are recognized in Java to make things a bit easier including:

- Int (signed integers)
- Complex (complex numbers)
- Float (floating real point values
- Long (long integers; these could be either octal or hexadecimal

The thing that you should keep in mind with these is to utilize some of the uppercase letters when you are combining them with the numbers. For example, if you use the lowercase l, the computer can easily confuse it with the number 1. So make sure to use the L when you are writing the program, and if you are unsure about how the letter is going to look inside a code with a lot of numbers and letters mixed together, it is best to go with the uppercase choice to make things look good.

Variables

Variables are really important no matter what kind of programming language you are going with, and this is no exception when you are working with Java. First, we need to understand that the variables are the means used in order t store the data. You are going to use them to determine where to keep all of the statements and other things that you are creating within your code. You will need to use a lot of different character, integers, and decimals in order to make the variables that you need.

The first thing that you will need to do is value assignation. The equal sign is the one that you will need to use the most because it really shows what everything is going to do together. You can show what the age equals or what the color equals or something else similar. You don't necessarily need to go through and make declarations that are explicit here, but you do need to make sure that the variable is on the left side of the equal sign and that the thing that will happen to the variable ends up on the right. A good example of how this works includes:

```
#!/usr/bin/java

counter = 100       # an integer assignment
miles = 1000.0      # a floating point
name ="John"    # a string

print counter
print miles
print name
```

When you put this into your system, you are going to get the output of:

```
100
1000.0
John
```

This is just one of the examples that you are able to use when you are putting together the variables in your language. You can have as many of these as you need to make the code work, just make sure that you are naming them properly and that they all line up the properly way.

Operators

Now let's take some time to look at the operators that are present with your Java Language. Operators are very interesting because they will allow you to manipulate the operand values in some way and there are so many different categories that come along with these. Some of the categories of operators that you may need to use within your code include:

- Arithmetic
- Identity
- Assignment
- Bitwise
- Membership
- Comparison or relational
- Logical

Arithmetic operators

There are a number of arithmetic operators that you are able to use. These are mostly going to be able to help you to do various mathematical equations in your coding, so they are good to know if you are making that kind of code, or just to have on hand for other uses. Some of the arithmetic operators that you will find handy include:

- (+)—this is the addition operator. It is going to add together two or more of your operators. For example, a + b = 10.
- (-)—this is the subtraction operator and it is going to take the right operand and subtract it from the operand that is on the left. For example, a – b =5
- (/)—this is the division operator. It is going to divide the operand that is on the left from the one that is on the right.
- (*)—this is the multiplication operator and it is going to multiply the two or more numbers that you want.
- (//)—this one is going to be one you want to use when you are looking for quotient results, such as when the decimals have been removed.
- (**)—this is the exponent. If you want to use "the power of" or another option similar, the exponent is the best choice for you.

Assignment operators

Next on the list are the assignment operators. These are good when you want to make sure that the two values are assigned together. This can be shown within your code, or you can use it in order to save the function that you are working on. Some examples of the assignment operators that you may want to work with in your code include:

- (=.)—this is going to give the value of what's on the right side back to the operand on the left. For example, if you have c = a + b, the value that you get from a + b, is going to turn into c.
- (-=)—subtract AND—this is going to take the operand that is on the right and subtract it from the left operand. It will then assign the results to the left. For example, c- = a is the same thing as c = c-a.
- (+=)—add AND. This is going to work the same as the one above, but it is going to add the operand that is on the right to the left, and then gives the left operand result.
- (/=)—divide AND. This will divide the left with the right operand and then will assign the value over to the left.
- (*=)—Multiple AND. This is going to multiply the right operand with the left operand before assigning the results to the left operand.
- (**=)—exponent AND. This is going to derive the exponential notation. C ** = a is going to be the same as c = c** a.

Identity operators

These can be really helpful any times that you want to compare the memory of two different objects, especially if they are going to be in different locations. Some of the identity operators that you may want to use within your code to make it stick together include:

- Is. This is going to determine whether the variables on both sides are going to be true. If they are not, the operator is going to point to the same subject.
- Is not—if the variables on either side of your code are false, the operator is going to point to the subjects and will say true-wise.

Membership operator

These are going to be used within the code to check if there is coherence, or a membership, within the specific sequence you are working on. To do this, try out these options:

- Is. This is going to check if the variables on both sides are true. For example, x is y, here is the result in 1 if id(x) equals id(y).
- No in—this is going to check if the variable is not inside of a specified sequence and can then evaluate whether this is false or true.

Bitwise operators

The bitwise operators are going to go through the operator bit by bit. A good example of this is:

a = 0011 1100

b = 0000 1101

---

a&b = 0000 1100

a/b = 0011 1101

a^b = 0011 0001

Logical operators

These are pretty basic ones that you can choose to go with. They are going to be basically items that will be able to hold 10 to 20 variables between them.

Comparison operators

These are going to make different comparisons to figure out if the statements are true and what the program should do if the statements are true or if they are false. Some of the comparison operators that you may use within your Java program include:

- (==.)—these are where the condition equates to true when the operands are equal.
- (<>)—if the values are not equal, the condition will become true.
- (!=)—the condition will become true if the values of two operands are not equal.
- (<=)—the condition becomes true if the left operand has a value that is less than the right operand.
- (>=)—the condition will become true if the value on the left side is greater than the right

operand.

- (<)—if the left operand is less than the right operand, the condition will become true.
- (>)—if the left operand is greater than the right operand, the condition is true.

As you can see, there are a lot of different things that you are able to do with your operators when they are set up properly. You will be able to do basic arithmetic equations, figure out if a statement is true or false, and so much more. You should take some time to go through and practice doing some of these options to see how they work and to get some experience!

There is just so much that you are able to do when creating a new code using Java. You will be able to combine some of these basic classes in Java to make the exact code that you want, adding in more things, taking things out, and just making it into your very own.

Reference versus Primitive Types

When it comes to working with Java, there are a few different kinds of data types. We have talked a bit about the independent or the primitive data types in Java, but there are also reference types, and these can go into the different categories as well. While they are slightly different, you will also need to have them supported by the primitive data.

While the primitive types are able to stand on their own, there are some things that the reference data types are able to do that will make the program work better and you will see a big difference when it comes to how well your program is going to work. There are a few different types of these you are able to use so let’s take a look at them to learn how they work.

User Datagram Protocol

This is a really important member of the IPS. It is going to provide transmissions that are connectionless so that they are reliable and they are not going to cause any issues within the protocol of your network in any way. If you have time-sensitive applications that need to drop the pockets quickly, this is the type of protocol that you should use. Some of the other things that you are able to see with the user diagram protocols include:

- This is able to provide datagrams to the network
- It is considered transaction oriented, so it will work for the Domain Name Systems and the Network Time Protocols
- It is going to work with unidirectional communication, which is good for broadcast information and service discoveries.
- It works for applications that need to happen in real time, such as Snapchat and Twitter, because it will work with transmission delays. It often works for some games as well as various VOIP applications, such as Skype.
- It is good when you are dealing with a lot of different clients and for streaming applications.

Transmission Control Protocol

Another option that you can go with is the transmission control protocol. This is a reliable stream of octets that will happen between the IP Network and the various networks that it contains. It is orderly and error-checked, so it is good for most of the applications that you will use online, but is especially used for email. This protocol is also connectionless and latent, and it will work to process data that has already been sent through.

There are a few things that you will need to keep in mind when using this, including that it is reserved and that you need to have the urgent pointers, checksum, and window size all reserved at the same time. Some of the things that you should keep in mind include:

- 4 bits specify the data header in 32-bit words
- 16 bits will identify the source port as well as the receiving port.
- 32 bits identify the acknowledgment of numbers
- 9 bits will contain the 9-bit flags
- 3 bits define the data that is reserved for future use.

Address Resolution Protocol

As you can get from the name, this is a telecommunications protocol that will make it easy for websites and apps to communicate. It is used to help map network addresses, and it will make video calls and other messages possible. There are many great technologies that are going to use this kind of protocol, including IPv4, Xerox PARC Universal Packet, ChaosNet, and DEC Net. Some of the things that you should keep in mind with this include:

0 – Hardware Type (HTYPE)

2 – Protocol Type (PTYPE)

4 – Hardware address (HLEN) or the Protocol Address (PLEN)

6 – Operation (OPER)

8 – Sender Hardware Address (SHA)

10 – Next 2 succeeding bytes

12 – Last 2 succeeding bytes

14 – Sender Address Protocol (SAP)

16 – Last 2 succeeding bytes

18 – Target Hardware Address (THA)

20 – Next 2 succeeding bytes

22 – Last 2 succeeding bytes

24 – Target Protocol Address (TPA)

26 – Last 2 succeeding bytes

Internet Protocol

And the one that is the best to use with Java, because this program was created to go along with it, is the Internet Protocol. This one is all about providing the right boundaries to the network, as well as relaying out datagrams that will allow all the networking on the internet to happen. The construction is going to involve a payload and a header where the header is going to be known as the main IP address, and the interfaces are going to be connected together with the help of specific parameters.

You will also need to use the routing prefixes and the network designation to get everything to work together. There are a lot of things that you will be able to add into this, but the basic framework that you will need to work with when it comes to Java and the Internet Protocol include:

Data Application

UDP Header | UDP DATA Transport

IP Header | IP Data Internet

Frame Header | Frame Data | Frame Footer Link

Making Some Classes in Java

Now that we have spent some time looking at the basic things that you are able to make with Java code, it is time to get out there and make some of your own classes in the code. The classes are really important when it comes to your Java program because they are responsible for determining what will happen to the program.

There are quite a few elements that are going to be considered when making a class. These are all important to make sure that the code is actually going to work. We are going to start out with the framework of the class and will move on from there!

Working on the framework

The things that you will need to include in the framework of the Java class that you want to make include:

- The program type. You will need to use the UNICODE declaration in order to describe the program type.
- Program information is going to be provided within the text. You should use the <HEAD> and </HEAD> symbols to surround this program information.
- The Java Program part is going to be described similarly, but the symbols are going to be <Java> and </Java>
- The title should be at the beginning and it is going to be found within the text <TITLE> and </TITILE>
- The text that you will see between the <H1> and </H1> are going to be the headings. If you want to have more than one heading show up, you would just need to change the heading number, such as having H2, H3, H4, and so on.
- The content of the program or the information that you will see on the screen is going to go between the symbols <BODY> and </BODY>
- And if you would like to create a paragraph, but write out the text between and

Your Cheat Sheet for Tags

When it comes to writing out tags, you need to remember that for the most part, they are going to come in pairs. You have seen this a bit above, where all the information that you need for the tags is going to fall in between two of the same tags on either side. What is on the left side is going to be considered the start tag and then the end tag will be almost the same, but you will need to use the slash right before it to indicate that this particular part of the code is done. Some of the tags that you will likely use in your Java program includes:

Unicode

- NAME –this is going to pertain to one of the sections on the page
- HREF—this is a good one to use when you want to hyperlink within the document or when you want to add in the name of a particular URL that you are referring to.
- onCLICK—this is the script that is going to run so that the user will be able to click n the anchor. This is going to work so that the user can click on your links and the link is actually going to work when they get there.

<A> Anchor

- TITLE—this is the name of the program that you are trying to open up.
- onMOUSEOUT—this is the script that is going to determine that the mouse is no longer the anchor text.
- TARGET—this is going to show the window where the program has to go into.
- onMOUSEOVER—this shows that the mouse is right on top of the anchor text.

<App…>

- CODEBASE—this is the path that would take you to the applet class
- CODE—this is going to determine which app is currently running.
- HEIGHT—this is the height of the applet
- WIDTH—this will determine the width of your applet
- ALIGN—this is going to determine how much of the text that you are writing is going to show up on the screen.

<APPLET…>

- HSPACE—this is going to talk about the horizontal space between the applet and the

surrounding text.

- VSPACE—this is all the vertical space that is between the applet and the surrounding text.
- NAME—this is the name that you will give to the applet and it is also the name that is used by other applets that are referencing it.
- BORDER—this is the empty space that will be seen surrounding the aplite.
- ARCHIVE—if the applet has taken some of the components and compressed them into one, this will be the collection of those components.
- MAYSCRIPT—this is going to show whether the JAVA program is going to be able to use JavaScript or not.
- SHAPE—this is going to show the shape of the area that is shown onscreen.
- ALT—this is the alternate text that will be used when the image is not able to display properly on the screen.

<AREA…>

- TARGET—this is going to determine the frame that you are going to go to.
- TITLE—this is a small description of the area that you are trying to use.
- NOHREF—this means that there are no links that are contained within the specific area.

These are just some of the different tags that you are going to be able to use when you are working with Java on your coding needs. There are many others and they will all have their place in the code that you are making.

Turning Your Classes into a Page

We could spend all day talking about the different tags that you can use in your classes and even what classes are, but in reality, you want to learn how to make these classes and even to turn them into pages. You want to be able to write out your new pages, such as a blog or a website, and get it to work properly for your needs. Here we will give some examples of how to make this happen. Remember that you will need to start out your code, no matter what you are writing, with <!UNICODE Java> so that it works properly. Here is an example for you to use!

<!UNICODE Java>

<Java>

<body>

<h1> Welcome to the Web! </h1>

* My first page *

</body>

</Java>

This is a simple example of what you would be able to do when writing a page. You would simply have a header that says “Welcome to the Web!” and then the first paragraph is going to say “My first page.” It is that simple. Of course, you will most likely want to write a lot of other things in this place, but this is a good place to get started on your first time.

The Headings

When you are writing online, you will need to make your page look attractive. You can't just have one big block of words on the page, not because the coding won't let you, but because this is hard to concentrate on and is not that easy to read. The headings can be something as simple as the top of the page telling what it is all about or little subheadings that help to break up the information and help it to make more sense.

The headings that you are able to use will range from <H1> to <H6>. They are really easy to put in. An example of how you would be able to do this includes:

<h1> Look at my heading </h1>

<h2> Today is a beautiful day </h2>

<h3> Tomorrow I have no plans </h3>

And so on. You will notice on your page that each of these are going to show up on their own on the page, showing that these are their own little sections. You will be able to add in some more words, as many paragraphs as you would like, to make it fill out and to get all the information that you need onto the page.

Paragraphs

Of course, when you are writing out the code, you do not want to just have a few headings and the title of the page. People come to the page in order to learn something, to figure out what is so important for you to tell them. Underneath all of the headings that you are writing, you will want to make sure to have at least one or two lines to help explain them more.

To write out a paragraph in your coding, you would simply need to use the and symbols to indicate when you are starting and when you are finishing. You will be able to write as much in between these symbols as you would like. Sometimes it is just going to be a few lines or a short paragraph, and other times it is going to be longer, like a page. There are no limitations to how long your paragraphs can be, just make sure that they have the right symbols surrounding them.

Images

There are times when you want to add in some images. These can be nice to enhance the articles that you are writing or if you are selling something, you will find that it is helpful to have quite a few pictures to show what items you have for sale. It is important to put the pictures into your code properly because they are important. With the right pictures you can make your website look more appealing, but the wrong pictures or the wrong placement of pictures can make your website hard to use.

To use the image tag, you will just need to use <img> around the pictures that you want to add in. Make sure that the attributes, including alternate text (alt), image source (src) and size are all in the right order.

Links

There are times when you will want to add some links in your website. If you are selling an item, you will want to link right back to the item that you are trying to sell. Or you might want to give an example of another website that is similar to yours or show where you got your information from. You will just need to use the <a> tag in order to make things easier. This is great because it keeps things simple on the page. A good example of how a link would like on your page would include:

See this link

How to Quote In Your Page

There may be some times when you are going to need to quote some parts from the page. For example, if you are trying to attribute some of your information to another website, it is sometimes best to get the word exactly the same so that the reader is able to get the most value out of the words, rather than having them all twisted around.

You will need to make sure that you are linking back to the original source if you do this, and quotes will help. Some of the things that you will need to remember when you are trying to quote in Java include:

- <q> --this is going to tell the program that you are adding a short quotation.
- <cite> --this is going to be attributed to the title of the topic that is on the page.
- <blockquote> --this means that you have gone and quoted something that is going to come from outside sources.
- <bdo> --this states that the program should do a text dictation.
- <address> --this is the contact information quote
- <abbr> --this shows that there is an abbreviation or acronym that is used.

You will be able to use these in order to write out a shorter quote, such as one small line, or to quote out something that is much longer, depending on what you would like to have on your page.

It is important to get your classes all organized from the beginning so that you can make the page look nice and neat on the website. The Java programming language keeps things pretty simple, allowing you to add the right title, heading, words, pictures, and even links that you need to ensure that the page looks nice to the end user.

Control Flow and Conditional Statements

There are times when you want to take a program and edit it to make it a bit more personal. If you want to do this with the code that you are writing, the conditional statements are going to make this a bit easier. They are going to work to make sure that the program you are developing is going to be completely your own, whether you want to change the color, the font, or something else. This chapter is going to take some time to talk about the different things that you can add to your page to make it look amazing.

Java Forms

There are many times when you will find that having a form is really nice within the Java pages. You may want to get some information from the customer, such as with a survey or with a poll. If you are putting up your business website, you may need a form that is going to help with contacting your business or one for the customer to fill out when they are placing an order. There are just so many times when the Java form is going to be nice. The elements that you can use in order to create your Java Form include:

- The text area elements
- The button elements
- The select element
- The input elements

A good example of what you will want to do to make this happen includes:

```
<div ng-app = “myFavoriteApp” ng-controller = “formCtrl”>
  <form novalidate>
   First Name: <br>
    <input type = “text” ng-model = “user.firstName”><br>
   Last Name:<br>
    <input type =”text” ng-model = “user.lastName”>
   <br><br>
    <button ng-click=”reset()”>RESET</button>
  </form>
  <p>form = {{user}}</p>
  <p>master={{master}}</p>
</div>
```

This will allow the person to place in their name to the form that comes up and then the computer will be able to recognize it if it is one of the options. Another form that you may want to consider is the Select Boxes. These are when you want to offer a few different options for the customer to use and they can make a selection. For the example that we are going to use below, you will put in

the names of some animal and let the customer, or the user, pick their favorite one.

```
<form>
Select an animal
<select ng-model=”myVar”>
   <option value-=””>
   <option value=”dogs”>Dogs
   <option value=”cats”>Cats
    <option value=”horses”>horses
</select>
</form>
```

You will be able to put in as many of the options as you would like on their form, anywhere from just two all the way to twenty or more, although most of these forms stay pretty small so that they are easier for the user to get through when needed. And with the example that was further up, if you need more information than just the first and last names, such as their mailing address and billing information, you will be able to add that in as well.

Control Flow Statements

The control flow statements are going to help to make sure that this program is going to run smoothly. It is going to do this by making use of the controllers that are present in the Java system. You will need to use the syntax JavaConnector::connect in order to get this to work for you. You will use this in order to get the string of IP Addresses that are available along with the clients and the servers that will work well together. When doing this, keep track of the following controllers to know which one is the right one for you.

Controllers 17 to 20

These are known as the connect::passing() because they are going to point the user to the right structure that is needed. It is going to help you to know how long your structure is, making it easier not to get your function confused with some of the others ones inside the program, as well as inside the network.

Controller 16

This is the first argument socket() so that you will be able to select all of the specifics and protocol families that you want to work with inside the communication networks. If you are unsure of how to get started, you should try PF_INET and SOCK_STREAM together because they work well when done together.

Controllers 13 and 15

These are the controllers that are known as JavaConnector::resolvehost. They have the ability to convert the DNS Host and name string over to an IP address so that all of the assumptions can be converted to different network addresses.

Controllers 5 to 10

These are going to be used in order to resolve the host and they will work under the syntax of JavaConnector::resolvehostname() and they work in order to convert the host names and IP Addresses with the getaddridinfo() function. This is one of the safer methods to use to get this process done, but there are some other choices.

Now if you want to connect to one of the servers, you will need to use the following code:

```
#include <string.h>
#include <netdb.h>
```

```
#include <arpa/inet.h>
#include “Javaconnector.h”
My JavaStream*( MyJavaConnector::connect(const char* server, int port)
    {
        Struct sockaddr_inaddress;
        Memset (&address, 0, size of(address));
        Address.sin_family = AF_INET;
        Address.sin_port = htons(port);
        if (resolveHostName(server, &(address.sin)addr)) !=0) [
        inet_pton(PF_INET, server, &(address.sin_addr));
        }
        int sd = socket(AF_INET, SOCK_STREAM, 0);
        if(::connect(sd, (struct, sockaddr*)&address, sizeof(address)) !=0) {
        return NULL;
    }
        Return new MYJavaStream(sd, &address);
    }
```

In order to resolve the host name, you will need to input the following information.

```
int JavaConnector::resoloveHostName(const char* hostname, struct in_addr* addr)
{
    Struct addrinfo *res;
    Int result = getaddrinfo (hostname, NULL, NULL, &res);
    If (result ==0) {
    Mempy(addr, &((struct sockaddr_in *)
Res->ai_addr)->sin_addr;
    Sizeof(struct in_addr));
    Freeaddrinfo(res);
```

}

Return result:

}

While this may seem like a lot of information to get started with, it is going to ensure that you are getting the right information sent to the server. This will also ensure that you get the right stuff on to your website and that your code, and your words, will show up nice and neat on the page in the way that you want them to. Make sure to try out this code a few times to see how it works, mess around to make changes, and see how it will look when you bring it up to your page.

Using the If and Switch Statements

These statements are going to be unique to your code because of all the things that you are going to be able to do with this code. Here we are going to talk about how to use the if statements as well as the if…else statements to show the different options that your user will be able to pick from and the switch statements that can make a difference in how your code looks as well.

The If Statement

The if statement is going to work on the true and false idea. It is going to decide whether the input that the user is putting in is true or false and then give the right response that you put in if the answer ends up being true. You will need to determine which statement that you would like to have come into the program if your user puts the true statement. On the other hand, if the user puts in a number that ends up being false based on the criteria that you put into the program, the screen is going to end.

While you may not like that the program is going to end if the user puts in the wrong number, this is how the if statement is going to work. We will explore a bit later what you are able to do if you would like there to be a few options available in case the user puts in a number that is not originally accepted by your code.

So say that you want to do a code for people who are voting. You may write out the code so that it has a few varieties. The question that the user is going to see is "What is your age?" They are able to put in any number that they want because it is just their age. But you will need to set up the program to respond in a certain way.

You may choose just to have a message come up if the person is older than 18. You would need to use the >=18 sign in the code and then add in the message that you would like. For example, if the person placed their age as 35, you could have the program provide the statement "Congratulations! You are able to participate!" and then it would go on to the next part of your code. On the other hand if your user places their age as 16, the program would either end or it would move on to the next thing that you want to show up in your program.

The if…else Statement

Now, with the option above, you are only going to have one answer that is true or false. If the answer comes out true based on what you wrote into the program, you are going to get the statement that you place inside of the code. But if the answer is false based on what you put into the code, you are going to get nothing.

This is not going to work that well for most people. You don't want your code just to give up after the answer ends up being false. In most cases, you will want to have a message that comes up whether the person answers inside a certain age range or not.

Let's take a look at the example that we did above. Let's say that the person did answer that they were 16. You still want to have something else come up in the code for someone who answers this age. You will be able to use the if…else clause to make sure that when the first statement doesn't end up true (the person isn't 18 or above that first one is false), the code is then going to move on to the second statement that you had available. You may choose something like "Sorry, you are not eligible at this time to participate. Please try again later."

So with the if…else statement, if the person is 18 or above, they will be able to get the original message that you wrote. But if they place in a number that is below 18, they will get the second message as we just discussed. This is still going to work off the true and false idea, but it is going to allow the program to show up examples no matter what answer you are given.

You are able to use the if…else statement as much as you would like. It is possible to do as many of these statements as you would like. This can be helpful if you want to split the people into several different groups. Let's say that you want to have three different groups, one that likes cats, one that likes dogs, and one that likes both. You will be able to list all of these choices on the screen and allow the user to pick them out. You can go through and add in the statements that you would like to show u based on the answer that you are given.

There are no limits on the amount of if…else statements that you are able to use when it comes to your code in JavaScript. You get to be in control so pick as many options as you would like to make it work for the ideas that you have with this particular code.

The switch statement

Another option that you will be able to use is the switch statement. This is going to allow you the option of testing out any variable to see the equality of it against a list of known values. A singular value in this option is going to be called a case, and the variable that is switched on is going to be checked for each of the existing cases. An example of doing an enhanced loop that has a switch statement inside of it will include:

```
switch(expression){
    case value:
        //Statements
        Break; //optional
        Case value :
        //Statements
        Break; //optional
        //You can have any number of case statements.
        Default : //Optional
        //Statements
}
```

There are many times when you will want to use the switch statements and the if statements in order to add more definition to your programs. You will be able to test them out a few times to get a feel for how they work and what all you are going to be able to do with them.

Working with Inheritances

Inheritances can be really useful when you are writing your code. They are going to allow you to take the features from one of your older classes and transfer it over to a new class as you need. You will then be able to make changes as well. You will also be able to keep the first class intact, which is helpful if you want to still use the first class, but also want to change some of the features to go in with the new class.

For example, if you transfer all of the elements from class A over to class B, you are going to have both of these still in the program. None of them are going to disappear. Right at the beginning you will simply have two classes with the same information, although most people who use inheritances will want to make changes to class B, while still maintaining some of the original features of class A. This makes things easier since you will be able to make those changes without completely getting rid of the first class.

When working in this kind of thing, remember that class A is going to be considered the original class and that class B is going to be known as the subclass, just to keep things in order. The elements of that first class may now be found under another class once you do the inheritance, but you will be able to trace the origins back to class A.

Concepts behind Inheritance

For the most part, when you look at programming books, they are just going to talk about how inheritance is the ability of elements to have common qualities. It will also highlight the fact that these elements are able to have some unique features, these features are going to be what allows the new class to stand out.

The inheritance is going to allow the elements of the first class to be transferred to another class, making it have a lot of the same features. But it is nice because you will be able to make some changes to this second class, without losing what you did in the first class at all. This is really useful because it will save a lot of time when trying to write the code and allows the programmer to “borrow’ elements from different programs in order to get the functions that they desire.

What Does the Transitive Relationship Mean?

When you are talking about the relationship between the class and the subclass, you are talking about a transitive relationship. In both mathematics and programming, this means that the relationship is going to be traced to a partial order, as well as to an equivalence order. If the group is introduced within another group, it is said that all the elements of those groups are going to be associated with each other. In addition, the transitive nature of your inheritance will point out that it is inclusive and extensive in characteristics.

There are two concepts that are going to be followed with inheritances. The first is that if class 1 extends into class 2, the elements of class 1 are going to be included inside of class 2. And second, when class 2 then extends into class 3, the elements of both class 1 and class 2 are then going to be included in class 3.

What this basically means is that you are going to be able to make as many inheritances down the line as you want. If you want to make a new code from class A, you can do that and then you can later go down the line and make a Class C that has the features of both Class A and Class B. You can't skip down and just go from Class B to Class C, you will need to have some of the features that are found in Class A in there as well, but you can keep going all the way down to Class Z or further if you choose in your code.

This helps to prevent waste in several ways. First, you are not wasting your time rewriting the same features of code over and over again, even if you want to reuse them within the code. Second, you will not waste as much space inside the code because the new subclasses that you are making will be able to link back to the original code, rather than having it rewritten a whole bunch of times within the code.

And of course, once you make the new subclass, you are going to be able to make some changes if needed. If you want that part of the code to do something a bit differently than the original one, that is fine. You will be able to make changes to that subclass which will not make any changes to the other classes or subclasses within your inheritance.

Java Shadowing

Another thing that you are able to do within Java that is connected to inheritances is shadowing. This is a technique that will take two variables that are named similarly and which have overlapping scopes. It is going to need one that is a higher variable and one that is a lower variable because one of the variables is going to overlap the other. The lower variable is going to have elements that will get hidden in the process.

For example, your lower variable can be called theta and the higher one is called beta. When the theta is able to overlap the higher variable, it is going to hide all the elements that are in beta. But just because the elements of the shadowed variable are hidden, doesn't mean that they are not able to work. They are going to work out just fine. They are often hidden for the purpose of not having to display the length.

You may want to do this for a few reasons. Some programmers like to use it when they want to combine a few elements together before making them into a new subclass. This can save a lot of time since you will not need to rewrite each of them in the process. The elements of both are going to be fully functional so you will be able to use them both, but you can combine them and make changes in the new subclass that you are trying to create.

It is also nice to sometimes hide the length of the higher variable. When you are working on the computer language, you need to be careful with what you are doing in terms of length or it can slow down the whole system. The shadowing technique is going to help prevent some of these issues and can make your whole program run a bit faster.

As you can see, there are so many things that you are going to be able to do when it comes to writing your own program. While we have discussed a lot of different things in this book, the inheritances can be some of the most useful. They allow you to mix and match things while still getting all the power that you need from the original code. You can make changes, keep carrying down the inheritance and so much more. Give it a try and see just what you are able to do when you introduce inheritance with your code.

Some Other Things You Can Do with Java

We have spent quite a bit of time discussing what you are able to do when you begin working with Java, but there are still so many other things that you will want to learn to do to make your program work well really. Some of the things that we are going to discuss in this chapter that will set your project apart from others when used properly are finally, objects, classes, and methods.

Classes, Objects, and Methods

When you are working in the Java language, these are often going to be grouped together because they are a collection of statements that will perform an operation within this language. For example, you are able to use the command System.out.printIn method and it is going to bring out a lot of ordered statements that will display on the screen.

If you want to write out a method, there is a basic syntax that will make the right things come up. This syntax includes:

```
Modifier returnValueType methodName(list of parameters) {
  // Method body;
}
```

Let's take a look at what all of these parts mean so that you have a good idea of what will fit into each part to make the code work properly.

Method Name: Modifiers

These are the optional modifiers that will tell the compiler how you want it to call up the method. The access type of the method should be defined in this way.

Return type

This is going to show some of the return values for the method. The ReturnValueType is going to be defined as the data type of the value the method returns. There are some operations that may be performed without returning a value if you choose. If there is to be no returning value, you need to make sure that you are using the keyword "void".

Method name

This is going to be the actual name of the method. The perimeter list and the method name will be the two items that make up the method signature.

Just Working with Classes and Objects

When you are working inside of Java, the object is going to be the component that will have behaviors and state. You can have a ball and notice that the states would be the color, how it bounces, and how big it is while the behaviors would be things like rolling and bouncing. When you are working with classes, you are working with the plan or the template that will describe the behaviors as well as the states of the object it is supporting. A good example of the syntax of a class includes:

```
public class Horse {
   String color:
   int age;
   String breed;

   void lame (){
   }

   void happy(){
   }
   void lazy(){
   }
}
```

There are a number of variable types that you are able to put within your classes to make some differences to the code including:

Local variables

These are the variables that are defined within the constructors, blocks, or methods. The variable is going to be initialized and then declared inside the method. When the preferred method is complete, the variable is going to be destroyed.

Instance variables

These are the variables that can be found within the class, but they will be outside the method that you established. These will be instantiated when the class is loaded and they can be accessed within any of the constructors, blocks, or methods that are in that defined class.

Class variables

The class variables are going to be declared within a specific class, outside of a method, with the static keyword you use.

Exception Handling

There are many times that you will need to use exception handling in your code in order to keep things safe. You must use the catch and try keywords in order to deal with the exception handling that goes on within your code. You can place these around the code that you want to generate the exception, or you can put this around any code that you want to place under protection.

The Throws/Throw Keywords

Any time that you have a checked exception that is not being handled in some way by the method, you will need to declare it with the throws keyword. This is going to appear right at the end of the method's signature.

On the other hand, if you have a new instantiated, or also an exception that you just caught, you will be able to throw it using the "throw" keyword. This is going to help you to make some differences in your code. Experiment a bit with both of these keywords so that you can learn how they are going to be a little bit different when used in the code.

Finally Keyword

And next, we are going to take a look at the finally keyword. When you have a block of code that will follow a try block, you will need to use the finally keyword. This is true even in cases where the exception has not occurred. This is a good one to learn about because the finally block is going to allow your cleanup statements to run to make the code look better and end on time, even if you have some protected code around. A syntax that you may want to use for the finally keyword includes:

```
try
{
    //Protected code
}catch(ExceptionType1.ei)
{
    //Catch block
}catch(ExceptionType2.e2)
{
    //Catch block
}catch(ExceptionType3.e3)
{
    //Catch block
{finally
{
    //The finally block always executes
}
```

Learning how to use the objects, classes, and methods in your code will help you to define certain behaviors and functions within your code on Java. Sometimes you will find that you are trying to put too much into the code though. More is not always better when it comes to using the Java program, so it is a good idea to test out the code a few times and make adjustments when they are needed. But overall, you will find that working with the methods, classes, and objects in Java can add some greatness to your code and can open up a lot of new things that you are able to create.

Conclusion

Thank for making it through to the end of *Java: A Detailed Approach to Practical Coding*, let's hope it was informative and able to provide you with all of the tools you need to achieve your goals whatever it may be.

The next step is to get started with your new programming language. We spent a lot of time taking a look at this programming language and how you will be able to create some of your very own websites and other great programs, whether you are brand new to coding or you have used some of the other coding languages and want to get started with Java.

While there are a lot of great coding languages that you are able to go with, Java is one of the best ones that you can go with if you are creating your own websites. It is not going to take too long to get it all set up and once you write out the information that you want on your page, as well as pick out the connection that is going to work the best for you, you are going to find that this is one of the easiest languages to use for writing your web page.

You don't have to be scared of programming if you are brand new to the whole experience. You simply need to know how to make it work and learn some of the syntaxes. Java is simple enough for even a beginner to master and you are going to enjoy how easy it is to get your website and other web content up and running in no time.

Finally, if you found this book useful in any way, a review on Amazon is always appreciated!

About the Author

Nathan Clark is an expert programmer with nearly 20 years of experience in the software industry.

With a master's degree from MIT, he has worked for some of the leading software companies in the United States and built up extensive knowledge of software design and development.

Nathan and his wife, Sarah, started their own development firm in 2009 to be able to take on more challenging and creative projects. Today they assist high-caliber clients from all over the world.

Nathan enjoys sharing his programming knowledge through his book series, developing innovative software solutions for their clients and watching classic sci-fi movies in his free time.

To learn programming from an expert, look out for more of Nathan's books in store and online.