These are some notes on the java programing language.

<http://docs.oracle.com/javase/tutorial/>

<http://docs.oracle.com/javase/7/docs/api/>

I have not double checked all the example code yet. Might have slight errors.

**Table of Contents**

Variables.

Arrays.

Statements.

Operators.

Loops.

Hot Keys.

Modifiers.

**Variables**

<http://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

( I have never used byte or short so I'm not exactly sure their use. The java docs give a technical explanation.)

byte

short

int integers are whole numbers ( -1, 0, 1, 2)

long longs are whole numbers but can be a lot larger than integers

float floats are numbers with decimals (1.04f) (145.2f) (.0054f) Floats are often declared with 'f ' after the number.

double doubles are just floats that can be a lot larger, and have more sig figs i think.

boolean Can be equal to (true) or (false)

char letters. ('a') ('d') ('!') surrounded by ' (apostrophe?)

string collection of letters and/or numbers. (“apples”) (“Jacob is majestic”) (“gr8b8m8”) they are surrounded by quotation marks when in code.

Declaring variables: In order for a computer to use a variable you need to tell the computer to distribute(set aside) memory for it. When you declare a variable you tell the computer what type of variable it is, what its name is, and you can optionally tell what the variable is equal to. If you don't tell what it is equal to then its equal to the variable types default value.

type of variable

name of variable

what the variable is equal to

All java statements are closed with a semicolon ;

This is a line of code that creates an int variable named “playerX” that is equal to the number 12.

int playerX = 12;

This is a line of code that creates an int variable named “playerY” but since it is not set equal to anything it is equal to int's default value.

int playerY;

PROBLEM The line above is wrong. playerY is null until defined. Only arrays set the variable to its default value.

|  |  |
| --- | --- |
| Data Type | Default Value (for fields) |
| byte | 0 |
| short | 0 |
| int | 0 |
| long | 0L |
| float | 0.0f |
| double | 0.0d |
| char | '\u0000' |
| String (or any object) | null |
| boolean | false |

(obtained from java tutorial, “[datatypes](http://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html)” link above)

**Arrays**

Arrays allow for chains of a single variable type with similar names. They are just ways of saving variables that are convenient to keep track of and work well for loops and other parts of coding.

They are declared by putting square brackets [ ] after the variable type when declaring it.

This line of code declares an int array.

int[] goldChains;

But before you can use the array you need to tell how many variable your array list will contain. If you want to change this you will need to delete the variable and declare a new one.

This line of code tells the computer that there will be 5 different variable in the goldChains array, because the variable is int they are all set to 0.

goldChains = new int[5];

in order to interact with the variable you use square brackets and their number. Arrays, as with many numbering systems in java starts, with 0.

goldChains[0] = 8;

goldChains[1] = 7;

goldChains[2] = 5;

goldChains[3] = 4;

goldChains[4] = 2;

You can get the length (how many variables are set aside for it) of arrays using .length after the variables name.

System.out.println(goldChains.length);

This line of code would output an int of 5.

You can use a for loop to easy display all of the arrays variables.

for(int I = 0 ; I < goldChains.length ; I ++ ){

System.out.println(“Gold Chains “ + I + “: “ + goldChains[i];

}

ArrayLists are open containers of variable that you can easy add more to and take some away. But they are more computer intensive, I don't know to much about the pros and cons.

**Operators**

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/operators.html

compares the two variable to meet the operator, if it does it will be equal to true, else it will be false.

Since booleans are either true or false then to compare variables for statements you need to learn the syntax.

What it looks like What it checks

( x == y) if the variables are equal

( x>= y) if x is greater than or equal to y

( x > y) if x is greater than y

( x <= y) if x is less than or equal to y

( x < y ) if x is less than y

( x != y) if x does no equal y

( x == y || x == z ) if x is equal to y OR x is equal to z (either need to be true for loop to happen)

( x == y && x == z ) if x is equal to y AND x is equal to z (both need to be true for loop to happen)

**Statements**

**If. If-Else**

Any booleans in this section can be replace by an operator as seen below.

If the boolean/operator is true it will run the body (code in the brackets).

If:

if ( boolean ){

}

If-Else:

if( boolean ){

} else {

}

If the boolean was false it will run the "else" statement once it's done.

Can also look like

This has another if check that runs in the first one was false.

if ( boolean0 ){

} else if ( boolean1 ){

}

**try, try- finally**

Try statements are used to suppress and handle java errors/ exceptions.

try{

} catch (Exception ex){

}

**Loops**

For

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/for.html

first section

second section

third section

for( int i = 0 ; i < x ; i ++ ){

}

for (initialization; termination; increment) {

statement(s)

}

And every new time it gets called it immediately does the first section only once per call. this will run until the second section is false. Every time it runs it does the third section once its done.

I think it un-declares the variable after the for loop is done running. ( 'i' variable in the above example )

While

these loops normally use operators or booleans that change when called upon by other code, if it was a variable that didn't change the loop would either never run or stop any other code from running again.

while ( boolean ){

}

This will run indefinitely until the the boolean is false.

Infinite loops will crash program or just prevent it from doing anything but that loop essential crashing it.

(Modifiers?)Purple Names:

Static: Can only be one of this variable, multiple objects of the class share the same variable.

Final: The value of the field can not change.

Public: this int or method can be called into other classes.

( className.varName / className.methName() )

Void: The method does not return any variable with and return statement. As opposed to...

Variables: (int, boolean, string and such) The method return a variable when called with a return statement, the method will normally be called differently here is an example.

int nextTick = 0;

public void setNextTick(){

nextTick = getTime() + 40;

}

public long getTime(){

long time = curTime - startTime;

return time;

}

the method getTime return a long, “time” when called the method setNextTick is called.

Eclipse

Explain eclipse workbench and ui.

**Hot Keys**

(eclipse)

ctrl + 's' Save open class

ctrl + shft + 's' Saves all classes

ctrl + shft + 'f' Organize open code

ctrl + space Suggests available code, probably has another name, very useful

“syso” (typed in workspace) + ctrl + space auto completes “System.out.println();”

ctrl + 'c' Copy highlighted text

ctrl + 'x' Cut the highlighted text

ctrl + 'v' Paste copied text

ctrl + 'd' Deletes selected line

shift + arrow keys highlight text passed over

ctrl + arrow keys jump to next capitol or blank space

shift + ctrl + arrow keys does both dummy

Organization of Code:

There are

Classes

Methods

Declarations / statements / loops

Coding has a lot to do with algebra, the main method may simply call one method, but that method can represent almost infinite lines of code and any number of other methods.

Get simple code to show people and give them a basic idea.

To Explain

How code is read/organized

Threads (only one thing can happen at once per thread)

Objects

methods

main method

classes

arrays

ArrayLists

Random shit

“\t” used in a string inserts a tab. I have only done this in the console

“\n” is a new line, also only tried in console.