# Cheatography

# CSC201 Cheat Sheet

by NoxLupus (NoxLupus) via cheatography.com/46432/cs/14862/

## Hello World

```
/* Noc_Luppus */
import java.u  ti l.Date;
public class Hello
 {
      public static void
main(S tring[] args)
      {
            System.ou t.p rin -
tln ("100 * 100 = 10,000 & 1000
* 100 = 100,00 0");
     }
 } // dont forget me
```

## Data Types

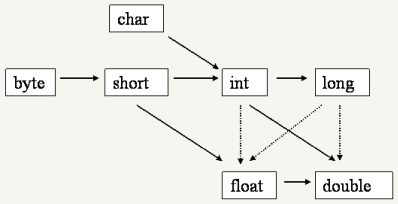
| Primitive types | size | Reference types | size |
|---|---|---|---|
| byte | 8 | Byte | |
| short | 16 | Short | |
| int | 32 | Intager | |
| long | 64 | Long | |
| float | 32 | Float | |
| double | 64 | Double | |
| char | 16 | Charter | |
| bool | 8 | Boolean | |
| | | All classes are reference type like: Scanner, Random, Die, int[], String[] | |

## Data Types (cont)

| holds value | holds a **pointer** to a value |
|---|---|

- *Term primitive is used with several meanings:*
- Provided as part of language
- Not composite (ie no component parts)
- Variable stores value
- Variable is not a pointer to the value
- Autoboxing is the automatic creation of a wrapper object from its corresponding primitive type

## data convertoin




- Solid lines lose no information
- Dotted lines may lose precision (but not magnitude)
- Chars are unsigned 16 bit values, shorts are signed 16 bit values
- A narrowing conversion of a value of an integer type simply truncates the value

## DecimalFormat

The advantages of the DecimalFormat class compared with the NumberFormat class include precise control over the number of digits to be displayed

Java.text's NumberFormat dose not truncate their display during the formatting process

```
Decima lFormat fmt=new Decima lF
o rma t("0.## ####);
```

## strings

String objects' lengths never chang and the shortest string has zero length

- Strings: have a .length() method (vs args.length field)
- Strings are dynamic and immutable
  - Each version of s is newly allocated
  - Mutable strings do exist
- Strings are not arrays of characters
  - Although a String does contain an array of characters

## Class Math

•Clients of Math can access its members using the class rather than objects

•Static members can be accessed using either the class or an object

•sqrt and PI are declared as static

## enumeration

Consider the following enumeration

```
enum Speed { FAST, MEDIUM, SLOW };
```

`Speed.c ha rAt(0)` = FAST

`Speed.c ha rAt(1)` = MEDIUM

`Speed.c ha rAt(2)` = SLOW

enumerations are like a sting of things, but they are unchangeable and, can be referenced by number

## javafx comands

| Rectangle name= new Rectangle(x, y, <>x, <>y); | rectangle upper-left corner is at coordinates (x, y) and its dimensions are <>x X <>y |
|---|---|




Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

# CSC201 Cheat Sheet

by NoxLupus (NoxLupus) via cheatography.com/46432/cs/14862/

## javafx comands (cont)

| | |
|---|---|
| Circle name= new Circle(x, y, r); | circle, centered at coordinates (x, y) and ridus of r |
| name.setStrokeWidth(2); | sets the size of the lines |
| name.setFill(Color.GREEN); | fills the shape with a color |

## JavaFX Qs

The individual items held within the JavaFX scene graph are known as nodes. root = first, branch = parent, leef = child

Parent, group, and stackPane nodes can be used as a root node in a JavaFX

The javafx.scene.shape packages includes classes that represent shapes in JavaFX

Event, control, and event handler are a kind of object that is used to create a **g**raphical **u**ser **i**nterface

You should override the start method in a JavaFX Application

In a development environment that fully supports JavaFX, Since the launch method is called automatically, you do not need to write the main method

A color image is broken down into individual pixels (points), each of which is represented by RGB

The coordinate of the upper-left corner of a stage is 0,0

## arrays

| | |
|---|---|
| DataType[] name = new DataType[x]; | crates a array, lanth x, DataType = int, string, char, object...ect, index = 0 - x-1 |
| **a**(int arrays) = **b**(int array); | will create an alias of b |
| "off-by-one" error | a loop gose +- too many times |
| ArrayIndexOutOfBoundsException | when the input is out of bounds like: arr[-1] = 0; |

an int array is passed as a parameter to a method, (int[ ] a) would adequately define the parameter list for the method header

## sorting algorithms

We compare sorting algorithms by examining the number of instructions

the amount of memory required by selection sort and insertion sort, neither method requires additional memory

selection sort: O(n2) time complexity

insertion sort: dose one item at a time. It is much less efficient on large

binary search: O(log2 n) efficiency

## classes

• Everything in Java is declared inside a class
• Packages are collections of classes
• Multiple classes per file allowed. But only one public class per file

## classes (cont)

• If a file contains a public class, the file must have the same name as the public class
• Careful: What do we mean when we refer to the class Hello?
• Classes are the only structured or user-defined types

In addition to their usage providing a mechanism to convert (to box) primitive data into objects, the wrapper classes provide static constants

All classes must have 1 parent (other than the object class with has no parent) but may have any number of children (derived or extended) classes

The relationship between a class and an object is best described as objects are instances of classes witch is created by the reserved word **new**

## Java Classes' Purposes

•Template for creating objects
•Type for (reference) variables
•Encapsulation mechanism (eg visibility control)
•Library of routines and constants

## Objects

**Objects serve two purposes:**
1. Objects model entities in real world (eg students, books)
2. Objects are instances of abstract data types (eg stacks and queues)

**Objects typically described as having 3 characteristics:**
1. State: Data



Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

## Objects (cont)

2. Behavior: Actions the object can take, perhaps modifying data

3. Identity: Objects are distinct (even if data is the same) and can be distinguished

**Programming with objects involves**

Creating objects

Sending them messages

Consider modeling a library's books

1. State: Author, title, status, ...

2. Behavior: getTitle, hold, checkout, return

3. Identity: Object of each book

Hmmm, multiple copies???

## Methods

• **Important rule:** *Static methods can call static methods only*

• **Instance methods** are declared without keyword static

• **Private** methods are not visible outside the class

• **Method** declarations methods are declared inside classes, not other methods (i.e., no nested methods)

• Having multiple class methods of the same name but differing types or numbers of variables **method overloading**

•**Static** methods can not reference instance data

•methods define the object's behavior

## Abstract

Abstract methods are used when defining: abstract classes, derived classes

## variable

In Java a variable may contain a value or a reference not a class, method, or package

If two variables contain aliases of the same object, then: the object may be modified using either alias, the object will become an "orphan" if both variables are set to null

a Java identifier can contain only A-Z, a-z, 0-9, _ and $, and start only with one of A-Z, a-z, or $.

A Java variable is the name of a data value stored in memory that can change its value but cannot change its type during the program's execution

## Literals

```
(2 > 3) ? true : false;
" tex t".s ubs tri ng(2);
System.ou t.p rin tln ("Di splay a hard coded float: " + 37.19f);
```

The code section 3.62 contains two number literals followed by two boolean literals at line 1, one string literal followed by one number literal at line 2, and one string literal followed by one real number literal at line 3:

## Instance Variables

Variables length and width are called instance variables

Each instance of the class gets a copy

Instance Variables are also called fields

Instance data for a Java class may be primitive types or objects

Scope: Entire class (and beyond, if not private)

Declared inside class but not inside any method

Instance variables should always be declared private

```
public class Box{
int foo = 5;
int bar = 10;  }
```

**default values**

integer type: 0

floating point type: 0.0

type boolean: false

type char: null character (\u0000)

reference type: null

array references: null

## Program Development

| | |
|---|---|
| establishing the requirements | **what** |
| creating a design | **how** determine the classes and objects needed |
| implementing the code | least creative step |
| testing the implementation | goal: to find logical and run-time errors |



Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

# CSC201 Cheat Sheet

by NoxLupus (NoxLupus) via cheatography.com/46432/cs/14862/

## simbles

| | |
|---|---|
| comments | . |
| // one line | method(args)<br>sum(int1, int2)... |
| /* multi */ | ++<br>adds one |
| /** javadoc */ | +<br>adding |
| | / + %<br>divition + remainder |
| | ^= \|= &= >> = >>>= |

## Terms

| | |
|---|---|
| Packages | are collections of classes |
| class | everthing is writen with in a class, blue print of object |
| object | an instance of a class |
| Method | main is a method (ie procedure or function) |
| Member | Methods are one of the kinds of members that can be in a class |
| Access modifier | specifies visibility |
| Kernighan and Ritchie (K & R) style | stase nl.starts nl.code end |
| Allmann style | nl.stase nl.starts nl.code end |

## Terms (cont)

| | |
|---|---|
| camel style | (eg doSomethingBig())<br>Variables, methods, packages |
| capita-lized | (eg Hello) classes |
| all_up-percase | CM_PER_INCH = 2.54;<br>Constants |
| instan-tiation | means creating a new object of the class or a new alias of a object |
| alias | two different numeric variables refer to the same physical object |
| API | Application Programming Interface |
| flow of control | the idea that code runsright though onless told other wise |
| arrays | are objects |

## Precedence

| | |
|---|---|
| postfix | . [ ] method(args) |
| unary | ++ -- + - new (type) |
| multiplic-ative | * / % |
| additive | + - |
| assignment | = += -= *= /= %= ^= &= \|= <<= >>= >>>= |

## Constructors

- Constructor has same name as class
- No return type for constructors
- Possible to have method and constructor with same name!
- Default no parameter constructor provided if no other constructors are provided
- Default constructor is NOT provided if others are (which can cause problems in some circumstances )
- Constructors override initializations in declarations

## Declare

| | |
|---|---|
| public | everyone can see |
| privet | only child can inderectly interact |
| protected | only child can see |
| static | static means it belongs to the class not an instance |
| final | cannot be changed |

**Access** modifiers: public, protected, and private

Modifier **requiring override**: abstract

Modifier restricting to **one instance**: static

Modifier **prohibiting** value **modification**: final

final double CM_PER_INCH = 2.54;

---

final double CM_PER_INCH;
CM_PER_INCH = 2.54;

---

public static final double PI = 3.1415;

By **NoxLupus** (NoxLupus)
cheatography.com/noxlupus/

Not published yet.
Last updated 15th April, 2018.


Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

## Object Assignment and Garbage

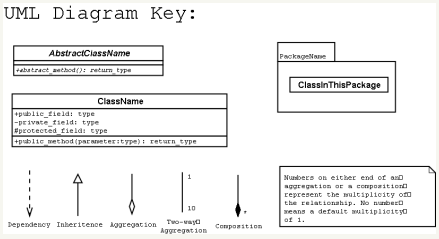
- Object variables are always references
- Assignment assigns references
- Java has automatic garbage collection

## Inheritance

- Inheritance through an extended (derived) class supports code reuse
- All classes in Java are directly or indirectly subclasses of the “Object” class
- “the default constructor”, “equals” and, “toString” are a method of the Object class
- Using the reserved word, **super**, one can access a parent class' constructor(s) , methods and instance data
- An object that refers to part of itself within its own methods can use **this** reserved words to denote this relationship
- The expressions that are passed to a method in an **invocation** are actual parameters

| | |
|---|---|
| parent class | child class |
| base class | driver class |
| super class | extended class |

## Unified Modeling Language




## Loops

all three loop (do, while, for) statements are functionally equivalent

| | |
|---|---|
| while( x==y) { stuff} | as long as teh conditoin is true it will run |
| do{ stuff }while (x!=y) | the do loop is the same as the while exsept it will exacte at lest once |
| for(x;-y;z){ stuff} | x = inst; y = condistoin; z = chang; it checks the condishtoin then dose the chang |

Loop Sitet

## object-oriented programming

*encapsulation*, *inheritance*, *polymorphism* are the main programming mechanisms that constitute object-oriented programming

Polymorphism: achieved by overriding

inheritance:

polymorphism:



Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com