Intro to Computer Science & Programming

A Cranes Club Initiative

Cranes Club

To create opportunities for professionals of Unificationist background to network and apply their expertise to better serve their communities and the greater society.

Overview

First Half: Fundamentals of CS using Java

- What is computer science?
- Object-oriented programming
- input/output
- Data structures and algorithms
- multithreading and GUI
- network programming
- Android app

Computer Science

The study of using computers to solve problems. How to solve problems better- more and faster?

Computer

A machine that can perform calculations and operations to accomplish various tasks. Fetch, decode, execute.

Program

A set of instructions that tell the computer what to do.

Programming Language

A specific way to write instructions for the computer, with its own syntax and grammar.

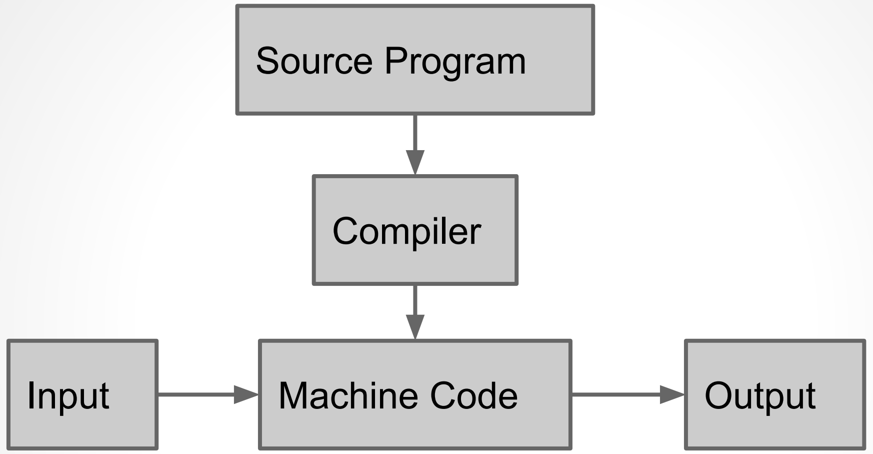
- Compiled
- Interpreted
- Imperative
- Declarative

Compiled vs Interpreted

Compiled: broken down into machine language before execution.

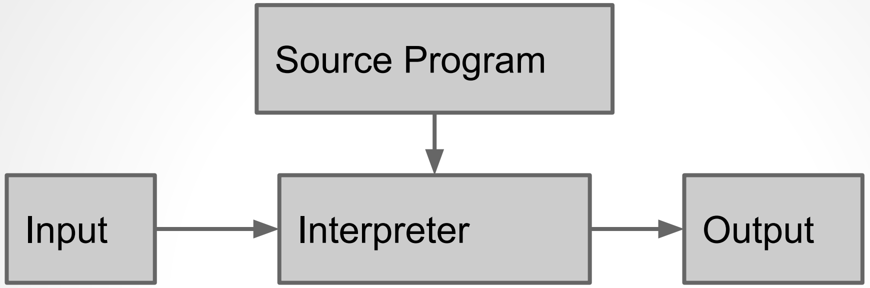
Interpreted: executed by an interpreter, which itself is a running program.

Compiler



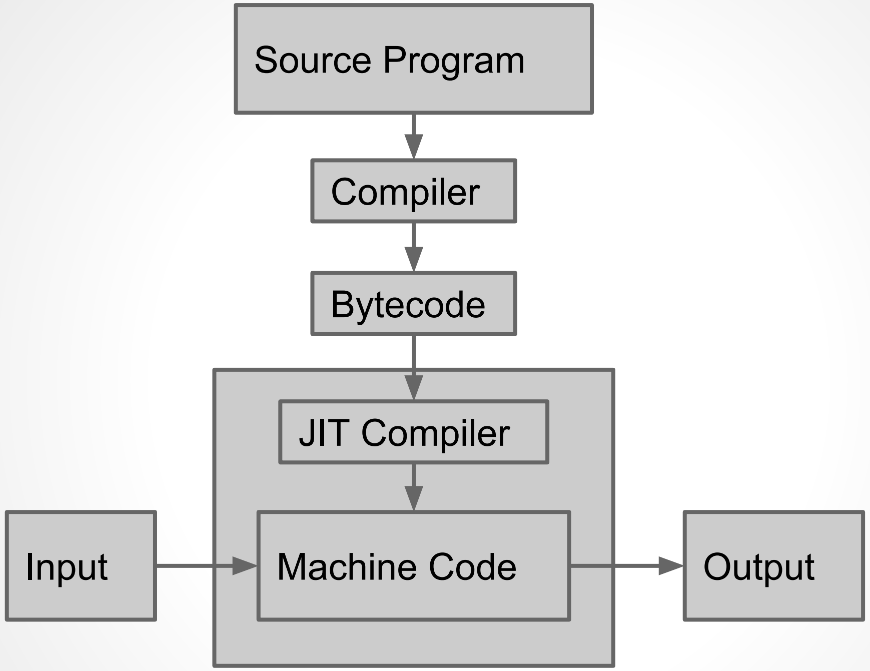
- Pros: performs fast.
- Cons: hardware-specific, compile time.

Interpreter



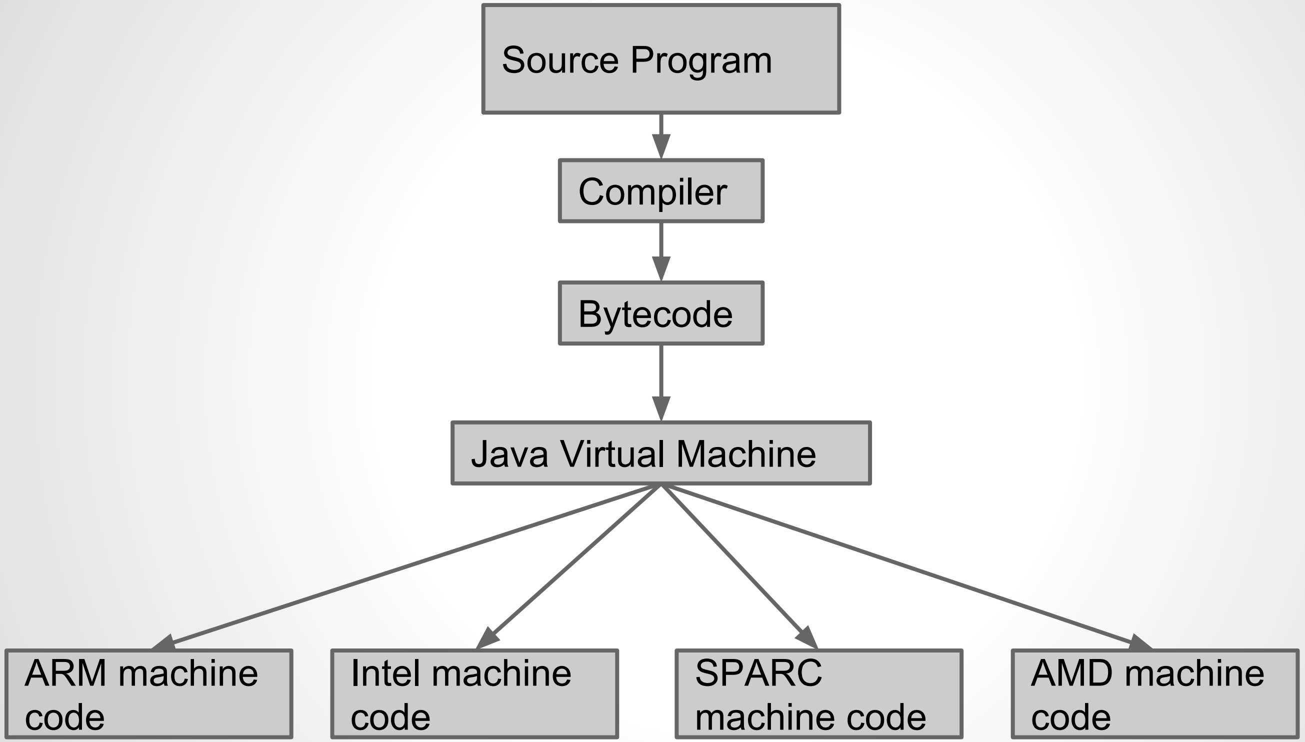
- Pros: dynamic, no compile time.
- Cons: lots of overhead, slow performance.

Just-In-Time Compiler

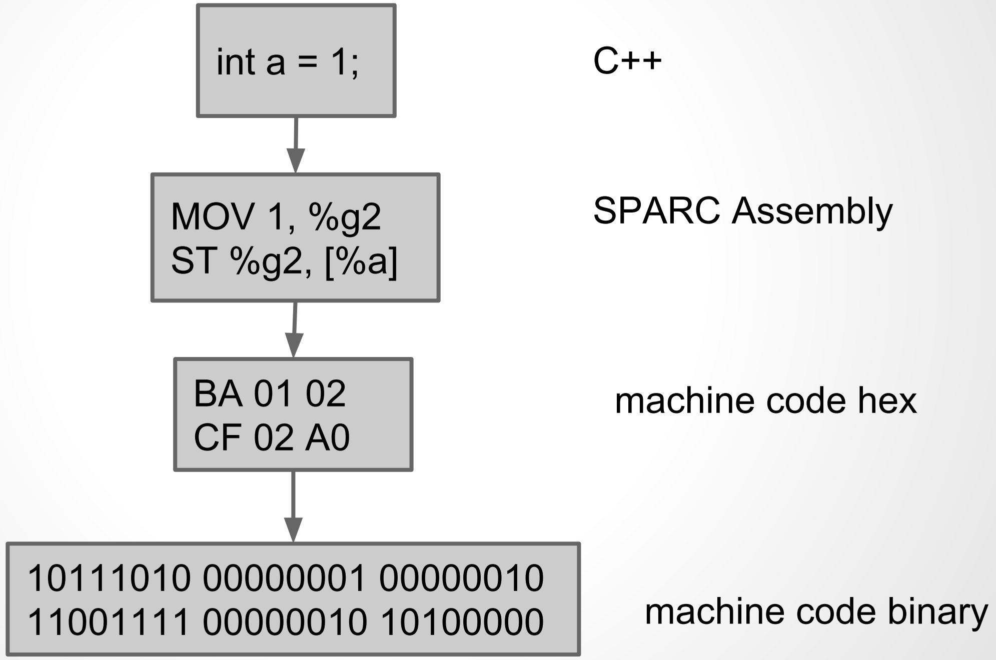


- Pros: optimizations can be done on the fly.
- Cons: overall still slower than compiled.

Bytecode


Source Program
Compiler
Bytecode
Java Virtual Machine
ARM machine code
Intel machine code
SPARC machine code
AMD machine code

Breaking down a Program



Declarative vs Imperative

- Declarative describes **what:** statements.
- Imperative tells **how**: control flow, state.
- Ex: get the min, max, and average price of a stock over the past week.

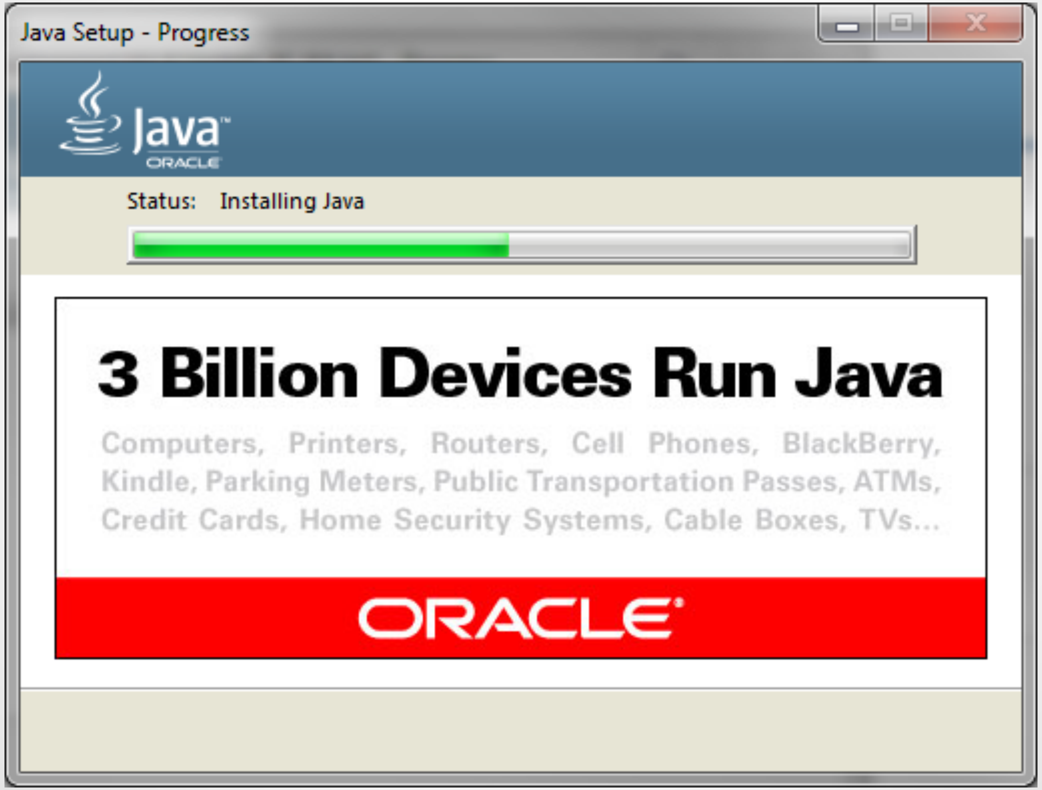
```
//Declarative implementation

double[] prices;

prices = (results.Selects(item => item.price).ToArray());

double min = prices.Min();
double max = prices.Max();
double avg = prices.Average();
```

```
//Imperative implementation

double min = results[0].price;
double max = results[0].price;
double avg;
double sum = 0;

foreach (TradeRecord record in results)
{
        if (record.price < min) min = record.price;
        if (record.price > max) max = record.price;
        sum += record.price;
}

avg = sum / results.Count;
```

Why Java?

- it is used everywhere- Android, web servers, enterprise systems, desktop applications, etc.
- Most popular language for jobs.
- Will help you learn other languages more easily (C#, Python, etc.)
- But, it’s quite verbose.

Why Java?


Java Setup - Progress

Status: Installing Java

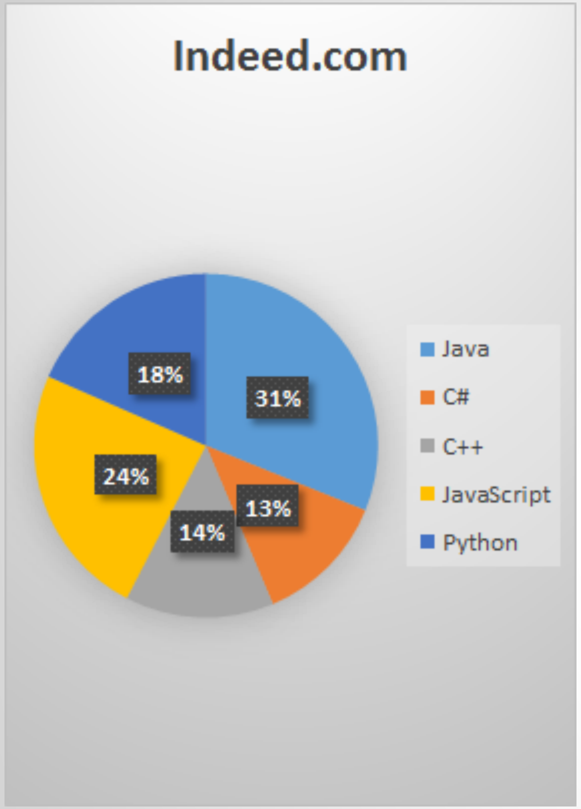
3 Billion Devices Run Java

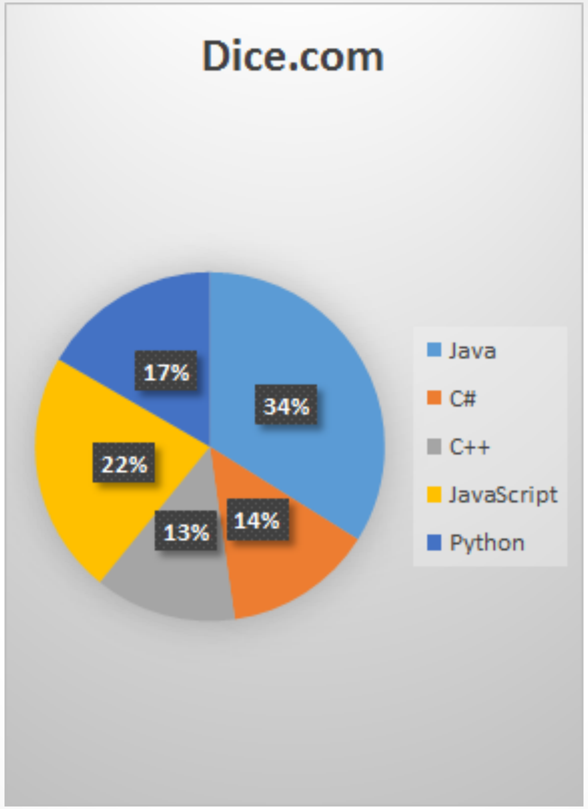
Computers, Printers, Routers, Cell Phones, BlackBerry, Kindle, Parking Meters, Public Transportation Passes, ATMs, Credit Cards, Home Security Systems, Cable Boxes, TVs...

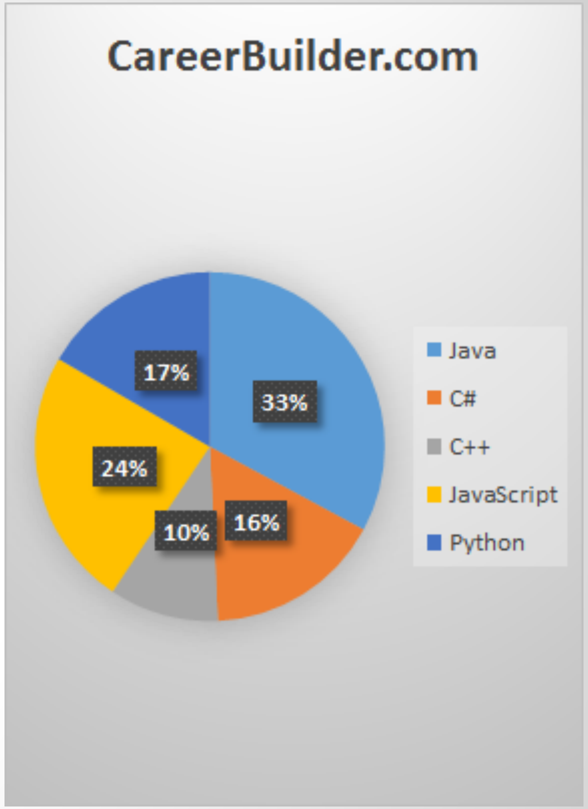
ORACLE

Developer Jobs per Language in NY

Job market by Language	Java	C#	C++	JavaScript	Python
Indeed.com	7964	3197	3573	6126	4711
Dice.com	1876	763	726	1245	921
CareerBuilder.com	273	136	84	199	138


Indeed.com
31%
13%
14%
24%
18%
Java
C#
C++
JavaScript
Python


Dice.com
34%
14%
13%
22%
17%
Java
C#
C++
JavaScript
Python

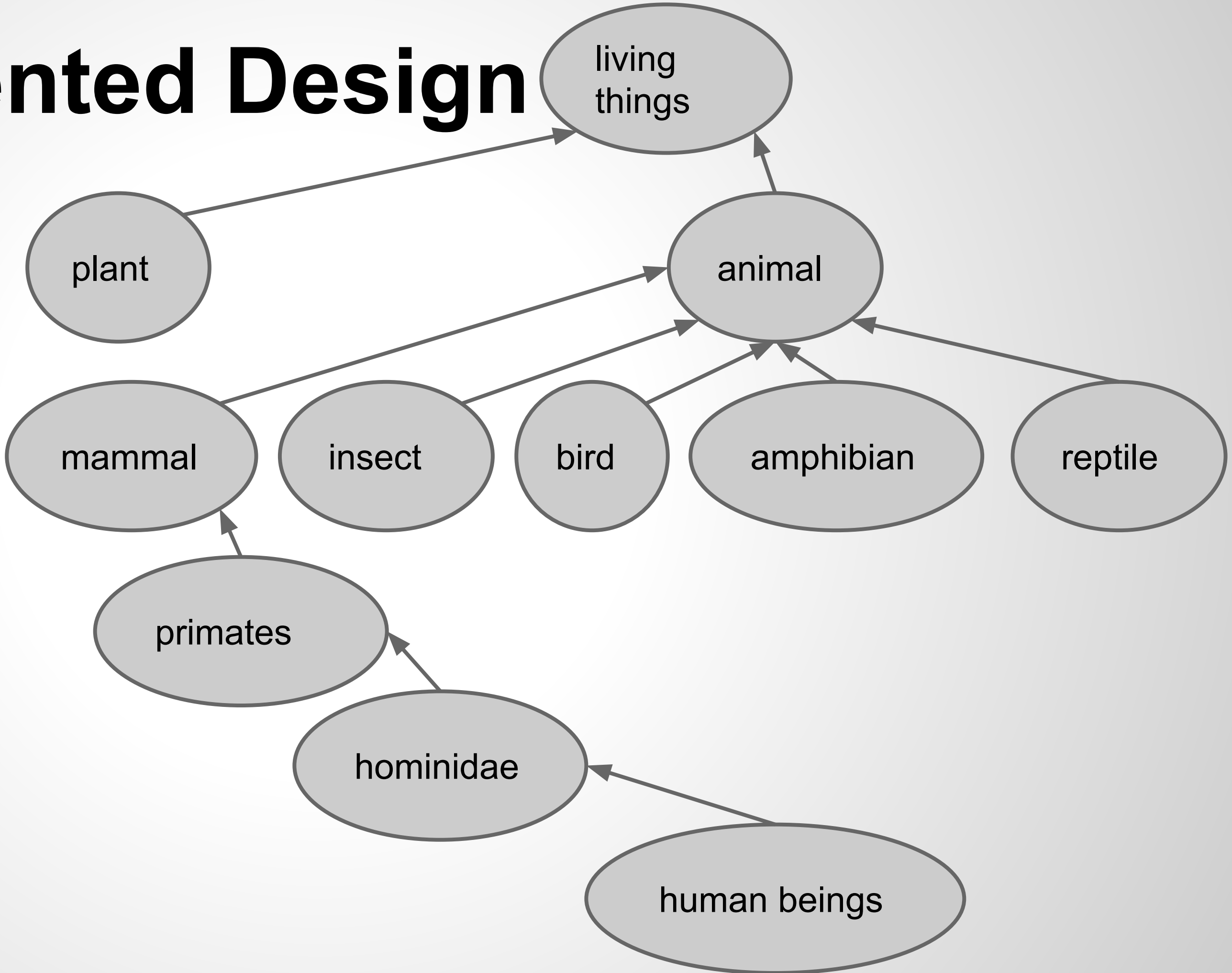

CareerBuilder.com
33%
16%
10%
24%
17%
Java
C#
C++
JavaScript
Python

Average Salary (source: Glassdoor)

Associate Software Engineer	$69,305
Software Engineer	$90,374
Senior Software Engineer	$106,575
Staff Software Engineer	$124,324
Software Developer	$86,226
Senior Software Developer	$122,296
Embedded Software Engineer	$82,739

Object Oriented Design

- objects
- classes
- behaviors
- interfaces
- inheritance
- encapsulation
- composition
- polymorphism



Overview of Objects and Classes

- **Class**: blueprint defining a set of behaviors (functions, methods) and states (fields, properties). aka Type.
- **Object**: an instance of a class.
- All Java programs are made up of classes and objects.
- Furthermore, all classes descend from the Object class.

The Object Class

Object behaviors (methods):

- clone()
- equals(Object obj)
- getClass()
- hashCode()
- notify()
- notifyAll()
- toString()
- wait()

The Object Class

- The Progenitor Class (the God class!)
- All classes in Java… past, present, and future… can do anything the Object class can do.

Example: Let’s Create a Dog Class

What are some dog behaviors?

- Bark
- Growl
- Whine
- Eat
- Sleep
- Fetch

What are some dog properties?

- Height
- Weight
- Coat
- Color
- Sex
- Temperament

Dog Class in Java

```java
public class Dog {
    int height;
    int weight;
    String coat;
    String color;
    Boolean sex;
    Enum temperament;

    void bark(){}
    void growl(){}
    void whine(){}
    void eat(String meal){}
    void sleep(int hours){}
    void fetch(String object){}
}
```

A Dog Class Object

- Recall an object is an **instance** of a class.
- Fido, AirBud, Lassie, RinTinTin are all instances of the Dog class (i.e. they are all **Dog objects**):
- They are also Object Class objects.

```
Dog Fido = new Dog();
Dog AirBud = new Dog();
Dog Lassie = new Dog();
Dog RinTinTin = new Dog();

Fido.bark();
AirBud.fetch("basketball");
Lassie.eat("dog food");
RinTinTin.color = "black";
Fido.toString();
```

Classic Hello World Example

```java
public class Main {

   public static void main(String[] args) {
  // write your code here
       System.out.println("Hello Cheon Il Guk!");
   }
}
```

```
Hello Cheon Il Guk!

Process finished with exit code 0
```

Syntax and Semantics

- Syntax: the **grammatical rules** of a language.
- Semantics: the **meanings** of a language.
- English syntax error:
  Bear honey the likes eat to.
- English semantic error:
  Honey likes to eat the bear

Java Syntax Overview

- Reference Types vs Primitive Types
- Variables
- Arithmetic Operators
- Console I/O
- Control Statements
- Comments
- Keywords

Reference Type vs Primitive Type

- **Reference Type**: descendant of Object
- **Primitive Type**: Simple numerical types. Not descendant of Object.

Reference Type

- **Reference:** the object's name.
- The `new` keyword is required to actually create the object in memory (in the **heap**).

```
Dog Fido;              //object reference only.  No Dog object yet.
Fido.bark();           //this will not work!

Fido = new Dog();      //Dog object is now created.
Fido.bark();           //this will work!
```

Primitive Type

- The `new` keyword is not used.
- These objects go on the **stack**.
- Wrapper classes are Reference Types

```
int x = 10;                                 //primitive type int object value of 10 on stack
Integer x_wrapper = new Integer(x);         //wrapper class for primitive type int in heap
x.toString();                               //this won't work!
x_wrapper.toString();                       //this will work!
```

Java’s Primitive Types

Primitive type	Size	Minimum	Maximum
boolean	—	—	—
char	16-bit	Unicode 0	Unicode 2^{16}- 1
byte	8-bit	-128	+127
short	16-bit	-2^{15}	$+2^{15}—1$
int	32-bit	-2^{31}	$+2^{31}—1$
long	64-bit	-2^{63}	$+2^{63}—1$
float	32-bit	IEEE754	IEEE754
double	64-bit	IEEE754	IEEE754
void	—	—	—

Variables

- Named value.
- Left side = name (identifier)
- Right side = value (literals, constants, expressions)

```
int h = 13;
int i = h * 2 / 4;
h++;
String word = "Hello";
Dog Pluto = new Dog();
Integer holla = new Integer(h);
```

Arithmetic Operators

Operator	Description	Example	Associativity

[] . ()	array access, method call	Fido.bark();
		dalmations[100] = “Pongo”;
			Left to Right

++ -- - !	increment, decrement, negative, NOT	weight++; degrees--; degrees = -40; verdict = !GUILTY;	Left to Right, Right to Left

/ % *	divide, modulo, multiply	gdppc = gdp / pop;
		odd = number % 2;
		e = m * c * c;
			Left to Right

+ -	add, subtract	sum = 1 + 1; diff = sum - 1;	Left to Right
< <= > >=	LT, LTE, GT, GTE	if (Fido.weight < RinTinTin.weight)	Left to Right
== !=	equality	if (AirBud.height == Lassie.height)	Left to Right
&& ||	conditional AND, conditional OR	if (Fido.weight < 10 && Fido.height > 20) { System.out.println(“Fido is too skinny!”); }	Right to Left
? :	ternary, conditional operator	(a ? b) result = true : result = false;	Right to Left
= += -= *= /= %=	assignment	a += 4; google_stock -= 34.0;	Right to Left

Console I/O

- Output: use System.out object.
- Input: use Scanner object and System.in object.

```java
System.out.println("Hello Cheon Il Guk!");
final double PI = 22.0 / 7.0;
System.out.println(PI);
System.out.format("%.5f\n", PI);
Scanner scanner = new Scanner(System.in);
String sentence = scanner.nextLine();
System.out.println(sentence);
int number = scanner.nextInt();
System.out.println(number);
```

```
Hello Cheon Il Guk!
3.142857142857143
3.14286
welcome to Intro to Comp Sci and Programming
welcome to Intro to Comp Sci and Programming
1234
1234

Process finished with exit code 0
```

Conditional Statements

- if/else conditional logic
- Can be nested
- Curly braces { } are important.

```
if (5 > 6)
   System.out.println("5 is greater than 6");
else
   System.out.println("6 is greater than 5");

if (PI > 3)  {
   if (PI < 2){
       System.out.println("PI is greater than 3");
       System.out.println("PI is less than 4");
   }
}

if (10 > 3)
   if (4 > 10)
       System.out.println("10 is greater than 3");
       System.out.println("4 is greater than 10");
```

```
6 is greater than 5
4 is greater than 10

Process finished with exit code
0
```

Comments

- for documentation purposes
- // for single line comment
- /* */ for multi-line comment

```
// this is a single line comment

/* this is
   a multi-line
   comment!
*/
```

Java Keywords

- reserved by the Java language
- cannot be used for naming things

```
Dog short = new Dog();      //can't do that!
```

Java Keywords

abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

Homework 1

- Tip Calculator
- Due by next week’s class
- For more Java practice, go to http://codingbat.com/
- Now it’s quiz time!!!!!