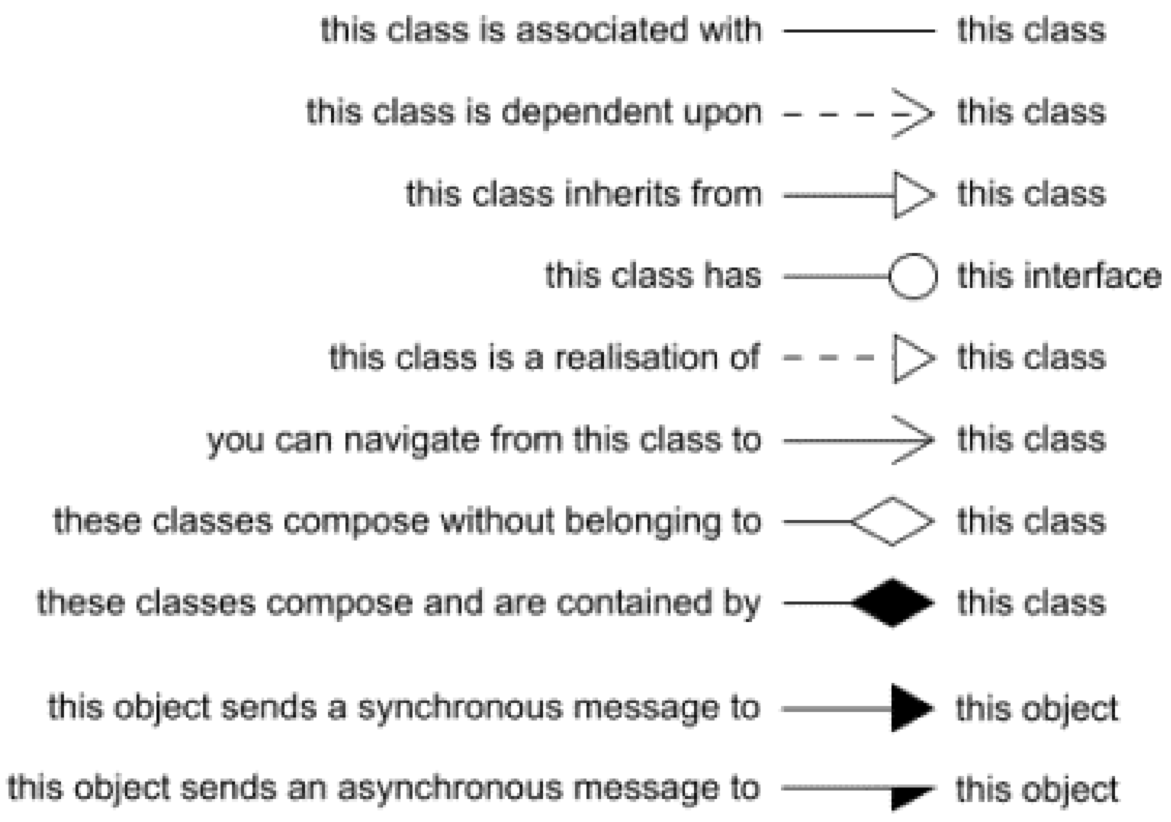
Table of contents

UML - What is UML

- ✓ Unified modeling language (UML) for visualizing, specifying, constructing, documenting of artifact of a software system
- ✓ The blueprint of a system is written in it
- ✓ UML is also used for modeling non-software system
- ✓ It is standard for building object oriented and component based software system
- ✓ UML is a notation system though which we can visualize a model of a system
- ✓ It describe only design or structure of system

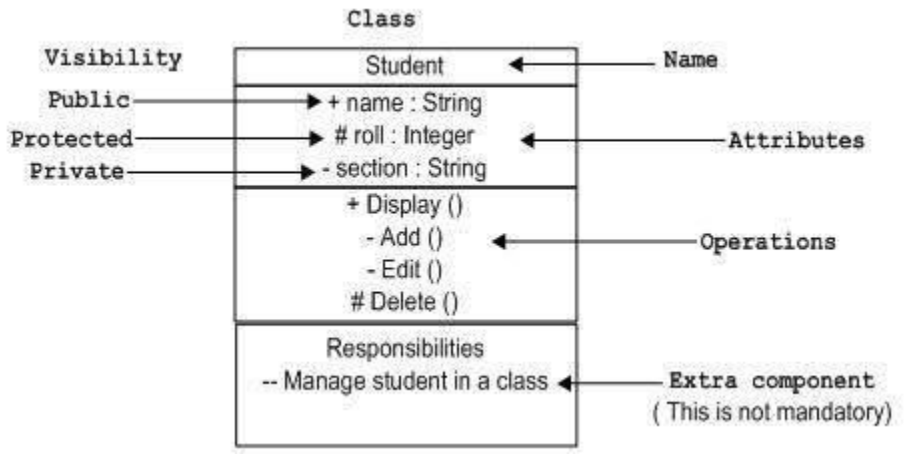
Additional information to be read regarding UML diagrams –
http://www.tutorialspoint.com/uml/uml_class_diagram.htm
http://www.codeproject.com/Articles/22769/Introduction-to-Object-Oriented-Programming-Concep

UML - notations

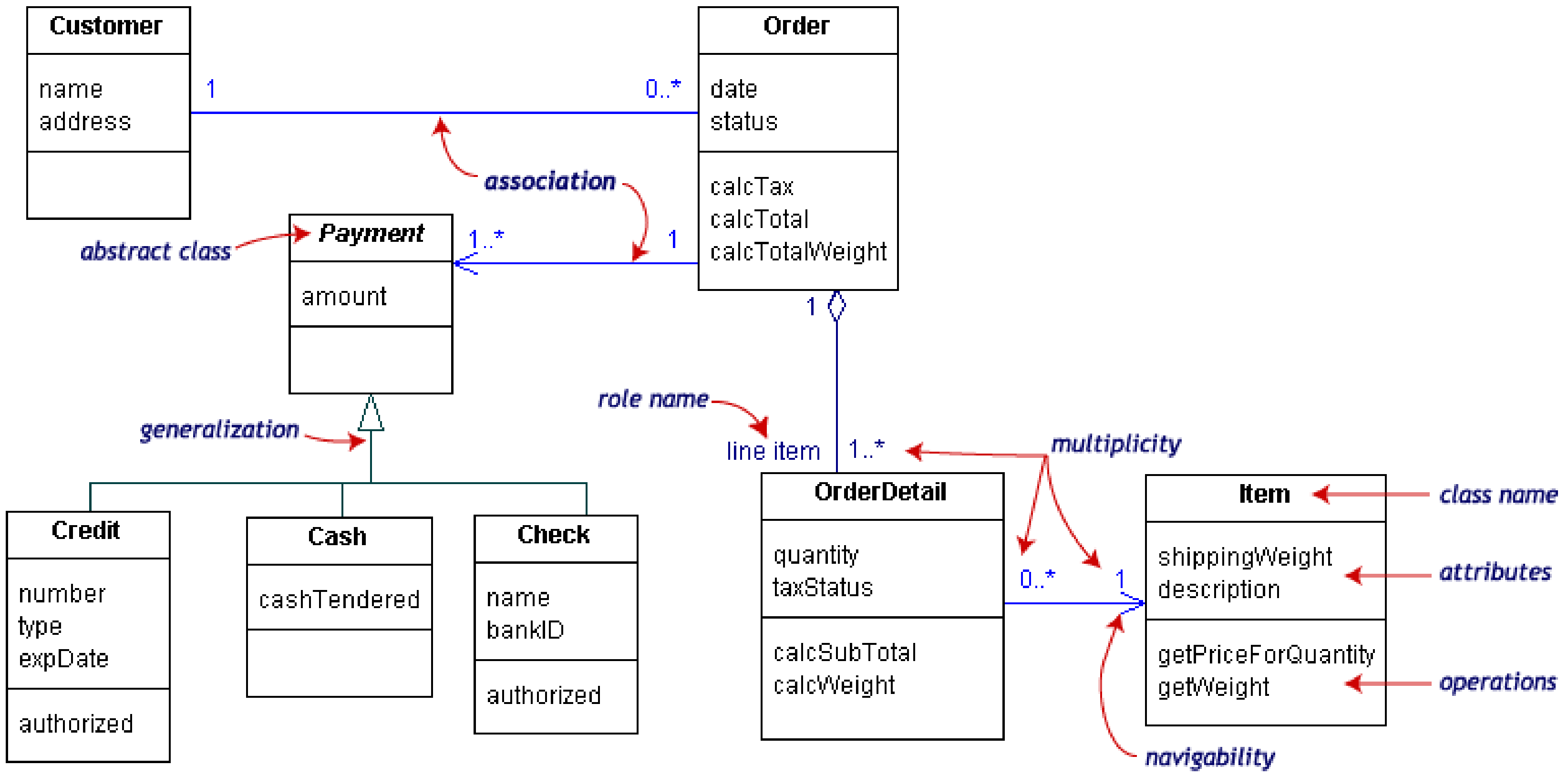

this class is associated with
this class
this class is dependent upon
this class
this class inherits from
this class
this class has
this interface
this class is a realisation of
this class
you can navigate from this class to
this class
these classes compose without belonging to
this class
these classes compose and are contained by
this class
this object sends a synchronous message to
this object
this object sends an asynchronous message to
this object

UML - Class diagram

- ✓ It depicts the static view of a model
- ✓ It is the basic building block of the object oriented system
- ✓ It illustrate the relation ship between classes in the system
- ✓ + sign with attributes or methods shows class member are public
- ✓ - sign with attribute or methods shows class members are private
- ✓ It shows relationship between class
- ✓ Generalizations are used to indicate inheritance.
- ✓ Aggregations are used to depict elements which are made up of smaller components



UML - Example of diagram

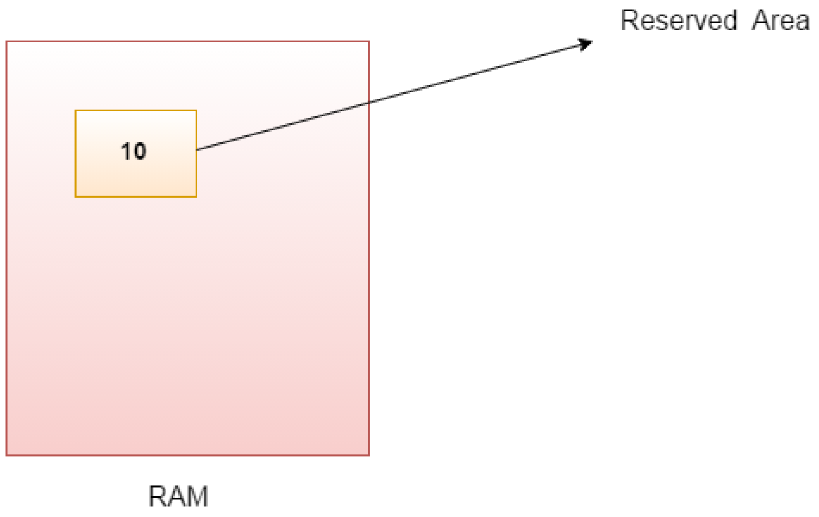


Basic data types

Variables and Data Types in Java

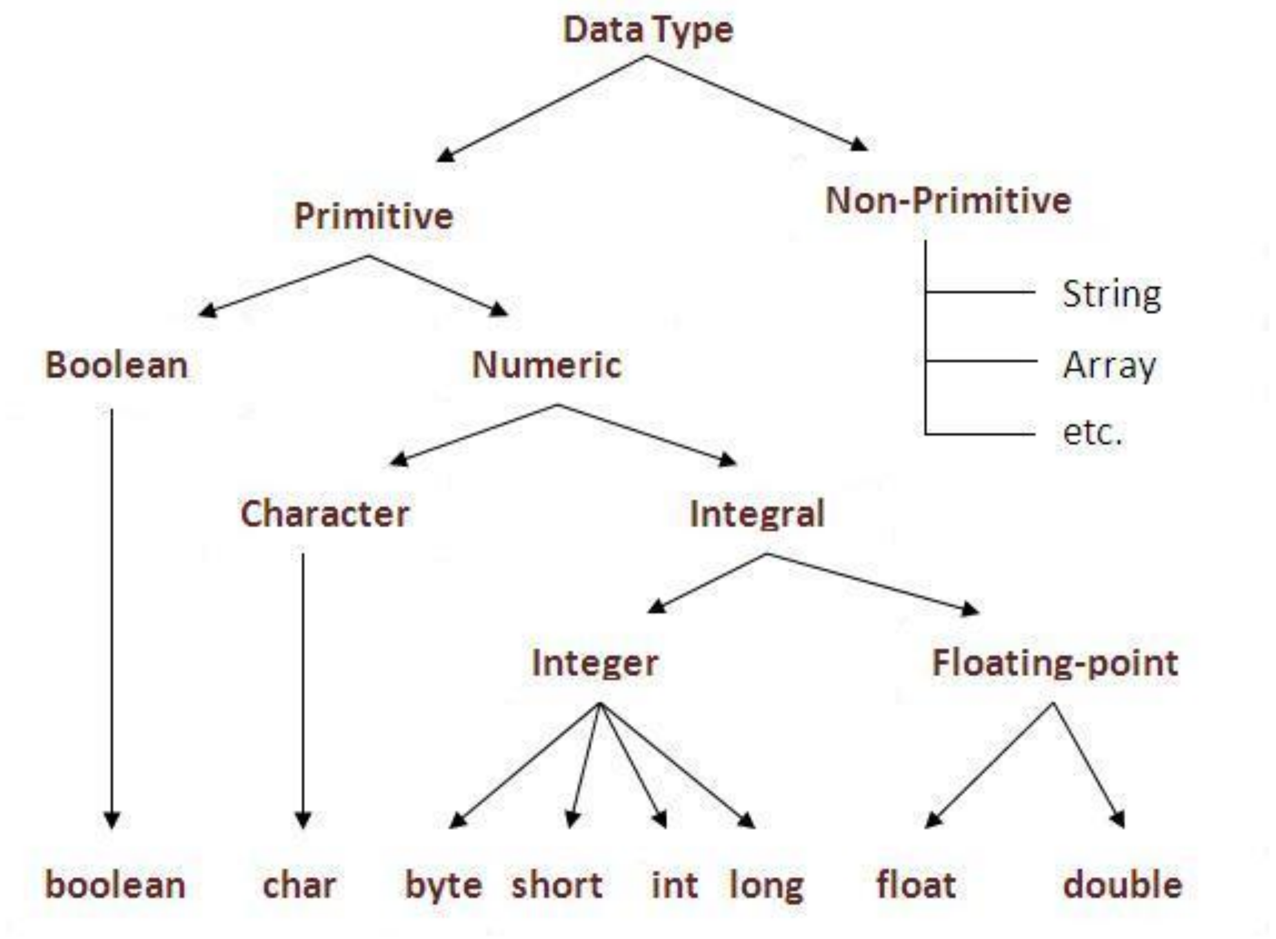
- ✓ Variable is a name of memory location. There are three types of variables in java: local, instance and static.
- ✓ There are two types of data types in java: primitive and non-primitive.
- ✓ **Variable** is name of *reserved area allocated in memory*. In other words, it is a *name of memory location*. It is a combination of "vary + able" that means its value can be changed.

```
int data=10;//Here data is variable
```



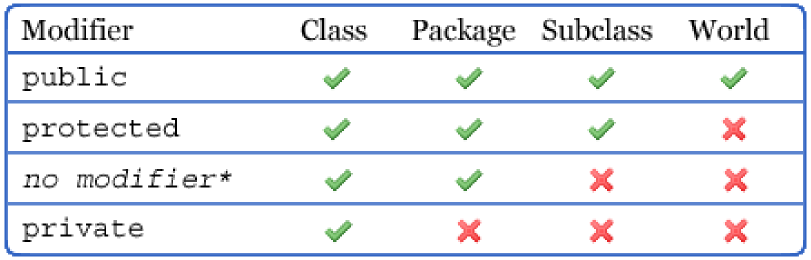
Additional information to be read regarding data types
http://www.tutorialspoint.com/java/java_basic_datatypes.htm

Basic data types



Java Access Modifiers

- ✓ **default** - visible to the package, the default. No modifiers are needed.
- ✓ **rivate** - visible to the class only
- ✓ **public** - visible to the world
- ✓ **protected** - visible to the package and all subclasses



Modifier	Class	Package	Subclass	World
`public`	✓	✓	✓	✓
`protected`	✓	✓	✓	✗
`no modifier*`	✓	✓	✗	✗
`private`	✓	✗	✗	✗

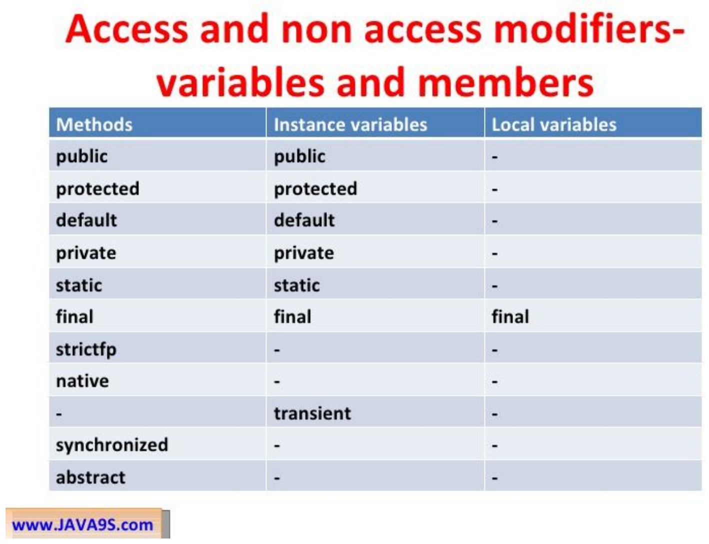
Read about access modifiers here:

http://www.tutorialspoint.com/java/java_access_modifiers.htm

http://www.slideshare.net/srinivasreddy_java/java-access-modifiers

Java Non Access Modifiers

- ✓ **final -** *(classes, methods, instance variables, local variables)*
- ✓ **static -** *(methods and variables)*
- ✓ **abstract -** *(classes and methods only)*
- ✓ **strictfp -** *(methods and classes)*
- ✓ **native -** *(methods)*
- ✓ **synchronized -** *(methods)*
- ✓ **transient -** *(instance variables)*
- ✓ **volatile -** *(instance variables)*



Access and non access modifiers-variables and members

Methods	Instance variables	Local variables
public	public	-
protected	protected	-
default	default	-
private	private	-
static	static	-
final	final	final
strictfp	-	-
native	-	-
-	transient	-
synchronized	-	-
abstract	-	-

www.JAVA9S.com

Read about non-access modifiers here
http://www.tutorialspoint.com/java/java_nonaccess_modifiers.htm
http://www.slideshare.net/srinivasreddy_java/java-non-access-modifiers

Java Constructors

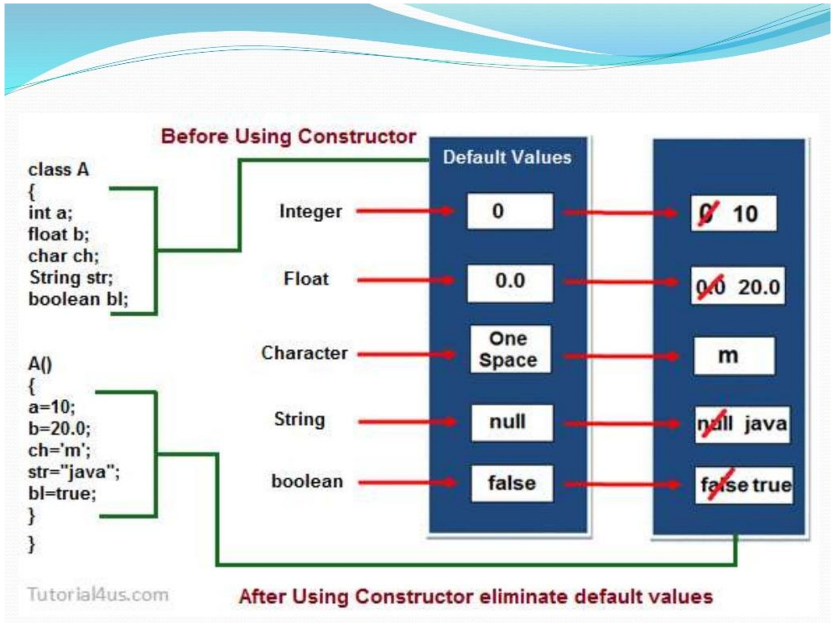
- Constructor is a special member method which will be called automatically when you create an object of any class.
- The main purpose of using constructor is to initialize an object.
- **Properties of constructor**
- ✓ Constructor name must be same as class name
- ✓ Constructor will be called automatically
- ✓ Constructor can’t return any value even void.
- ✓ Constructor should not be static
- ✓ Constructor can’t inherit

Java Constructors

How Constructor eliminate default values ?

Constructor are mainly **used for** eliminate default value, whenever you create object of any class then it allocate memory of variable and store or initialized default value.

Using constructor we initialized our own value in variable.



```
class A
{
int a;
float b;
char ch;
String str;
boolean bl;

A()
{
a=10;
b=20.0;
ch='m';
str="java";
bl=true;
}
}
```

Read about Java constructor:
http://www.javabeginner.com/learn-java/java-constructors/
http://www.tutorial4us.com/java/java-constructor

Java **this** keyword in java

- ✓ This keyword is a reference variable that refers the current object in java.
- ✓ This keyword can be used for call current class constructor.
- ✓ Why use this keyword ?

The main uses of this keyword is to differentiate the formal parameter and data members of class

Read about this keyword in Java
https://docs.oracle.com/javase/tutorial/java/javaOO/thiskey.html
http://www.sitesbay.com/java/java-this-keyword.php

Java **void** keyword

- ✓ The void type is the type for the result of a function that returns normally, but does not provide a result value to its caller.

- ✓ Void is the Java keyword that tells the compiler that a function will not be returning any value after it is executed.

- ✓ For instance, you make two functions, square() and calcSquare(), which both calculate the square of a number. The difference is that square() will find the square of a specific variable call "input", while calcSquare() returns the square of the number passed as an argument to the function. In this example, square() would be void since it does not return anything, but instead directly alters a known variable. However, calcSquare() will not be void because it does return a value.

Read about void keyword in Java:
https://ru.wikipedia.org/wiki/Void
https://teamtreehouse.com/community/what-is-void-and-return-in-java

Java **return** keyword

- A method returns to the code that invoked it when it
  - ✓ completes all the statements in the method,
  - ✓ **reaches a return statement,** or
  - ✓ throws an exception
- The return statement is used to perform an explicit exit from the method. The operator can be used anywhere in the method to return control to the object that caused this method. Thus, return stops the execution of the method in the place where it is located.
- Return is the Java keyword that tells the compiler what will be returned when a function is finished.

Example:

```
// a method for computing the area of the rectangle
public int getArea() {
    return width * height;
}
```

Read more about return keyword in Java:
http://www.tutorialspoint.com/java/java_access_modifiers.htm
http://pr0java.blogspot.com/2015/05/return.html

Java Object and Classes

Class − A class can be defined as a template/blueprint that describes the behavior/state that the object of its type support. Main components: variables, methods, constructors

```
public class Dog {}
```

Object − Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behaviors – wagging the tail, barking, eating. An object is an instance of a class.

```
Dog myDog = new  Dog();
```

Read more about creation of class instances (objects) -

http://www.tutorialspoint.com/java/java_object_classes.htm