OOPS + Java Part 2

JDBC: Java Database Connectivity

It is an API for middle layer interface between java applications and database.

It acts as a middle layer interface between java applications and database.

It defines how a client may access any kind of tabular data, especially relational database.

Manages three programming activites:

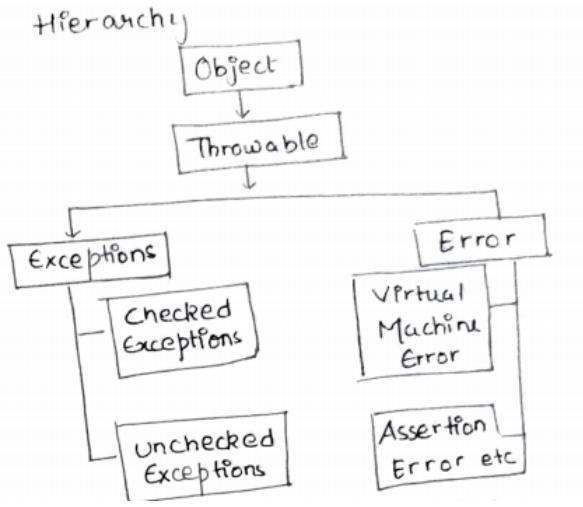
- connect to a data source, like database
- Send queries and update statements to it
- Retrieve and process the results received from the database.

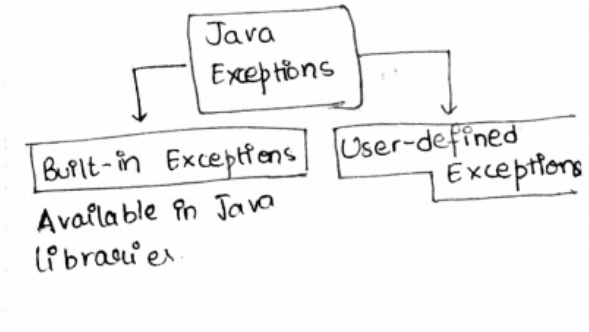
Exception

An exception is an unwnated or unexpected event, occurs at runtime.

Error: it is a serious problem that a resonable application should not tryy to catch.

Exception: conditions that a reasonable application might try to catch.





Checked Exceptions: Exceptions that are checked at compile time.

If some code within a method throws a checked exception, then the method must either handle the exception or it must be specified using throws keyword.

Unchecked Exceptions: These are not checked during compile time.

In Java: Exceptions under error and runtime exceptions classes are unchecked exceptions, everything else under throwable is checked.

If the exception is recoverable, then make it checked exception

If the exception isn't recoverable make it unchecked exception.

throw: Used to explicitly throw an exception from a method or any block of code. We can throw either checked or unchecked exception.

Throws: used in the signature of method to indicate that this method might throw one of listed type exceptions.

Syntax: type method_name(param) throws exceptions_list

- used for checked exceptions
- usage of throws keyword does not prevent abnormal terminal of program.

How JVM handles an Excaption?

If an exception occurs, method creates an object known as exception object known as Exception object and hands it off to the run-time system(JVM)

Run-time system searches the call stack to find the method that contains blocks of code that can handle the exception

Call-stack: ordered list of methods

Exception handler: block of code to handle the exception

Run-time system goes in reverse order of the methods called to find where the exception occured.

If the runtime-system searches all the methods on call stack and couldn't find it, then it hand the exception object to default exception handler, part of run-time system. It prints the exception info and terminates the program abnormally.

Constructor Chaining

Calling one constructor from another constructor with respect to current object.

- within same class: using this()
- from base class: using super () (because of inheritance)

A subclass constructor's task is to call superclass's constructor first and ensure creation of sub class objects starts with the initialization of the data members of the superclass

Why do we need constructor chaining?

When we want to perform multiple tasks in a single constructor rather than creating a code for each task in a constructor, we create a constructor for each task and chain them. Improves program readability.

Serialization and Deserialization

Serialization: mechanism of converting the state of an object into a byte stream

Deserialization: Reverse process of the above where we recreate the java object using byte stream.

The use of the above two is that, byte stream created is platform independent. Object serialized on one platform and deserialized on another platform.

Objects of these classes need to implement java.io.serializable interface

Serializable is a marker interface (it has no method and no data member)

It marks these classes so that they get certain capability.

Only non-static data members are saved via serialization process.

Constructor of an obj is never called when an object is deserialized.

Serialization

```
Demo object= new Demo(1,'hey');
FileOutputStream file = new FileOutputStream(file)
ObjectOutputStream out= new ObjectOtputStream(file)
out.writeObject(object)
out.close();
file.close();
```

Deserialization

```
FileInputStream file= new FileInputStream(file_name)
ObjectInputStream in= new ObjectInputStream(file)
object1= (Demo)in.readObject();
in.close();
file.close();
//Always put each block of code in try catch block
```

SerialVersionUD: associates a version no. with each serializable class called serialversionUID, which is used to verify that the sender and receiver of serialized object have loaded classes for that object which are compatible w.r.t serialization.

Serailver: The serialver is a tool that comes with JDK, it is used to get serialVersionUID number for Javaclasses.

enum in Java

They are used to represent a group of named constants in Java

```
enum Suit
{ DIAMOND, HEARTS, CLUB, SPADE
  private Name()
  { System.out.print("called for:" +this.toString()) }
}
public class Test
{ public static void main (String[] args)
  { Suit s1= Suit.DIAMOND;
    System.out.println(s1);
  }
}
```

enum can contain constructors and it is executed seperately for watch enum constant.

can create concrete methods only i.e. not any abstract method.

enum cannot extend anything else

every enum constant is always implicitly public static final, can access using name and can't create child enums(final)

can write main() method inside enum.

each constant in enum is an object

enum type can be passes as an argument to switch statement

Static keyword in Java

non access modifier used for

- blocks
- variables
- methods
- nested classes

when a member is declared static, it can be accessed without any reference to any object.

Static block: used for static initialization. The code in the block is executed only once, first time an object of that class is made or first time you access a static member of that class.

```
// syntax
static { //code
}
```

Note: static blocks are executes before constructors.

Static variables: a single copy of variables is created and shared among all objects at class level.

- just like global variables
- can be created at class level only
- static block and static variables are executed in their order they are present.

Static method: this method could be accessed before any object of its class are created and without any reference to any object. Few restrictions:

- can call other static methods only

- can access static data only
- cannot use this or super keyword

Static classes: use can have static class in java. In java, we cannot make top level class static. Only nested classes can be static.

Nested static class doesn't need to reference outer class, but non static nested classes will have to

non-static nested class can access both static and non-static members of outer class whereas static class can access only static members.

Final keyword in Java

- final keyword with variable value cannot be changes once initialized.
- final with class= this class cannot be subclasses, that class cannot be extended
- final with method: cannot be overridden by a subclass or any other way

Note: if a class is declared as final then by default all of the methods present in that class are automatically final but variables are not

Finally keyword

used in association with try- catch block, and guarantees that the section of code will be executed no matter what.

Application: used for resource deallocation

all resources which are opened in try block are needed to be closed, so that we won't loose our resources.

JDK1.7: fianlly block is optional because resources opened in try block will automatically get deallocated when the flow of program reaches end of try block.

Finalize method in Java

Not a reserved key word

It is a method that the garbage collector always calls just before the deletion/ destroying the object with is eligible for garbage collection.

Part of clan- up activity,closing resources allocated

once finalize method completes, immediately garbage collector destroys that object.

If we call a finalize method, it will be executed like any other method but the object would be destroyed only after system.gc() is executed

If an exception occurs while the user explicityly calls a finalize mehtod, JVM terminates the program by rising exception. Termination: abnormal

If garbage collector calls finalize method, while executing finalize method some unchecked exception rises.

JVM ignores the exception and rest of the program will be continued normally, termination Normal

not all exceptions, only unchecked exceptions are ignored

If a catch block is present, it is executed.

It is not necessary that system.gc() is executed at the very instant, it's up to JVM to call garbage collector or not. It is called when there is not enough space available in Heap area or when memory is low.

Wildcards in Java

Can be used for parameters, represent unknown type, fields or local variables.

Type of wildcards:

- upperbound wildcards:

```
public add (List <?extends Number> list_name)
// integer and Double are subclass of Number thus a Integer or
//Double list would be passed.
```

- lower bound wildcards: This method below will take Integer or its superclass objects

```
public add(List <?SuperInteger> list_name)
```

- unbounded wildcard: unknown type useful when writing a method which can be employed using functionality provided by object class.

- When code is using methods in the generic class that don't depend on the type parameter.
-

```
private void printList(List<?> list_name)
```

Java

It was developed by James Gosling in 1991. High level, object oriented, high performance, robust, secure, platform-independent, multi-threaded and portable programming language.

Other- Important features

Interpreted: used Just-in Time(JIT) interpreter along with compiler.

Dynamic: Supports dynamic loading of classes.

- Neutral Architecture
- Secured: because it doesn't use explicit pointers. Also provides concept of byte-code, exception handling etc
- Robust: Uses strong memory management, automatic garbage collection, exception handling etc
- JIT Compiler: it is used to improve the performance .It is compiles pieces of byte code which have the same functionality at the same time, thus reducing the total time required for compilation
- It is basically a translator from instruction set of JVM to instruction set of CPU
- Order of specifiers doesn't matter , public static void == static public void
- You can't use an explicit return type with the constructor, but it implicitly return the current instance of the class

Constructor

- name same as class name
- no explicit return type, returns instance of the class
- invoked implicityly
- used to initialize the state of object
- default constructor is provided if there is no constructor

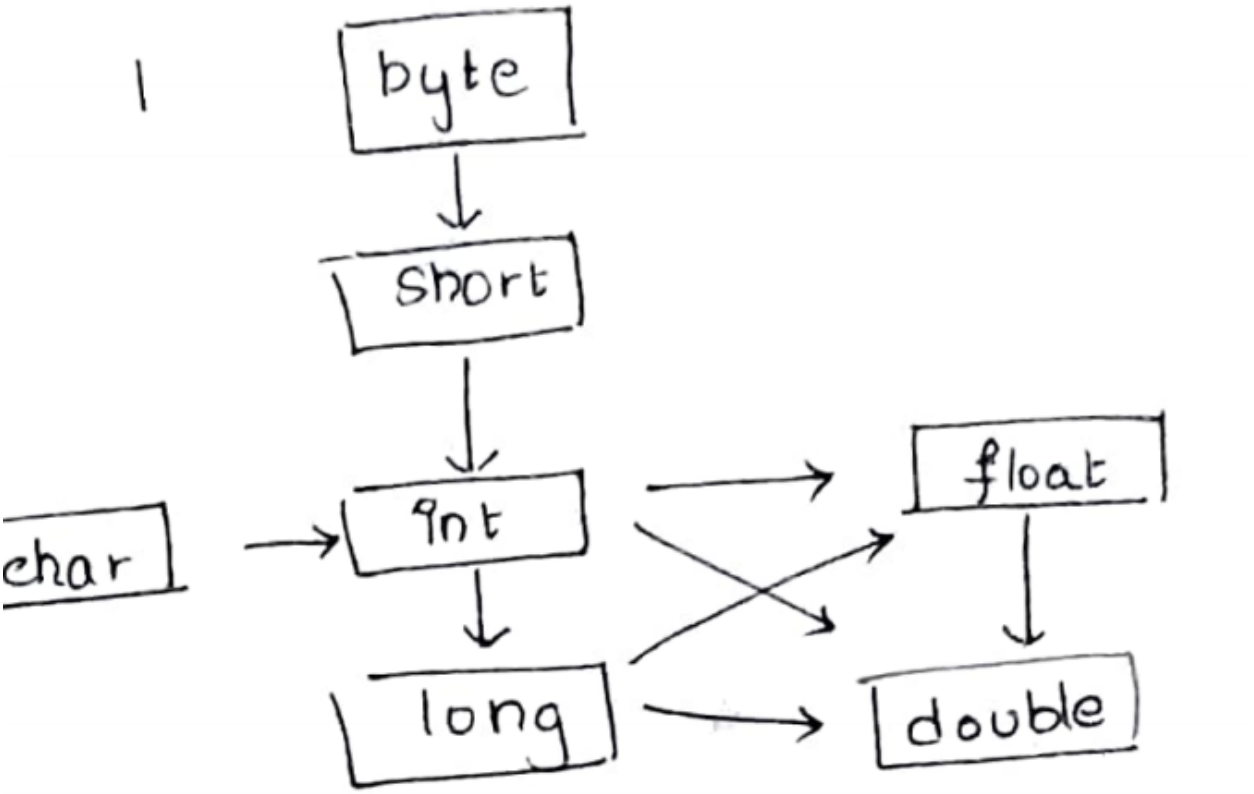
Method

- may or may not have the name same as class name
- explicit return type is there if required
- invoked explicitly
- tells about the behavior of the object
- no default method provided by compiler

Why is main method static?

To call a static method, object is not required. If the main method is not static, JVM will have to create the object first and then call main() method which will lead to extra memory allocation

Method overloading with type promotion



Why cannot we override static method?

Because static method is part of the class, and it is bound with the class, whereas instance method is bound with object.

Note:

Static gets memory in class area

Instance gets memory in heap

- we can change the scope of the overridden method in subclass. but we cannot decrease the accessibility of the method.
- All functions in java are virtual by default
- Instance of: instance of operator, returns boolean, its true if the object is of that class.

```
simple s= new simple();
System.out.println(s instanceof Simple1);//true
```

- Marker Interface: an interface that has no data member and member functions Example: serializable, cloneable etc
- Throwable class is the base class for error and exception
- A read only class can be made in Java by making fields private and having only getter methods which return values.
- String Pool: space reserved in head memory that could be used to store things. Main goal and advantage is that it saves memory.
- Whenever we create a new string literal, the JVM checks the "string constant pool" first. If it already exists then its reference is returned. If not, a new string instance is created and placed in pool
- Why are objects immutable in JAVA?

  Java uses the concept of string literals, suppose 3 reference variables are pointing one object. If one variable changes the obj's value then all 3 are affected
- "==" checks references, a.equals(b) checks the value of a and b. Strings created using constructor and string literal do not point to the same location.
- How can we create an immutable class in Java?

  By defining a final class having all of its members as final
- Metacharacters: those characters which have special meaning to the regular expression engine. Ex: ^, $, *, _ etc, to treat them normally, use backslash
- Nested class can access all the data members of the outer class within its (static implicitly), a class can have a interface in it.
- Garbage collection: process of reclaiming the unused runtime objects, performs memory management. A process of removing unused objects from memory to free up space and make this available for JVM
- Types of Inner classes(non-static nested classes)
  - Member Inner class: created within class outisde method
  - Anonymous inner class: created for implementing and interface or extending class.

- Local Inner class: a class created withing the method:

```
interface Eatable{ void eat();}
Eatable e = new Eatable(){
public void eat(){
sout("Hey");}
e.eat()}
```

- nested interface: static by default, the external interface or class must refer to the nested interface. Can't be accessed directly. It must be public if declared in interface, can be any modifier if declared in class.

Difference between Abstract class and Interface

Abstract class:

- can have abs or non-absolute methods
- can have instance variables
- can have constructor
- can have static methods
- can extend 1 abstract class
- can provide implementation of interface
- can have class members like private, protected etc.

Interface:

- has only abstract methods
- cannot have instance varaibles
- cannot have constructor
- cannot have static methods
- can implement multiple interfaces
- cannot provide implementation of abstract class
- members of a java interface are public by default.