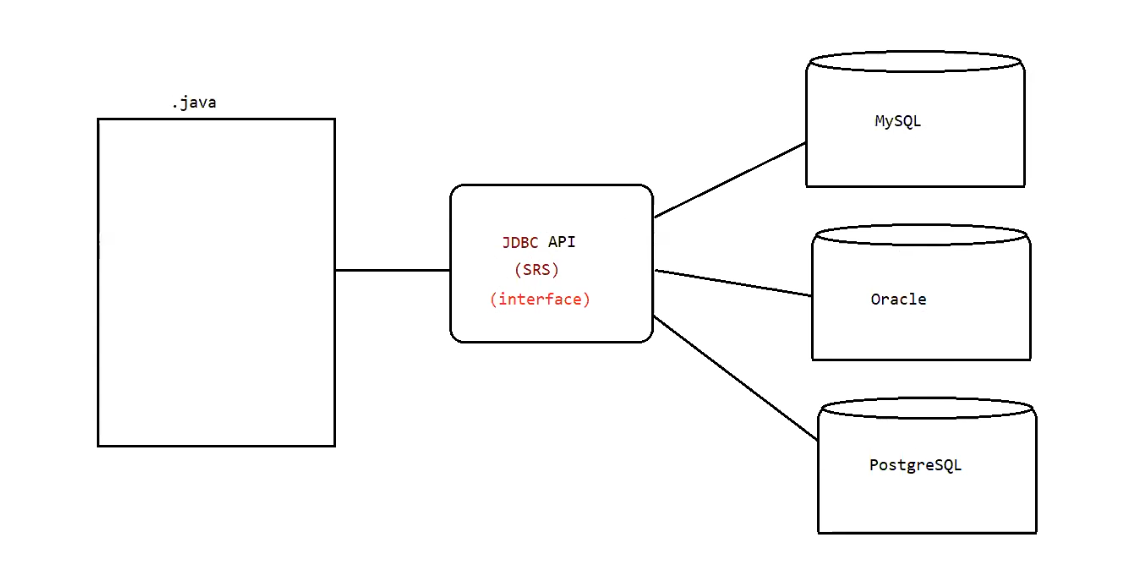
Interface :

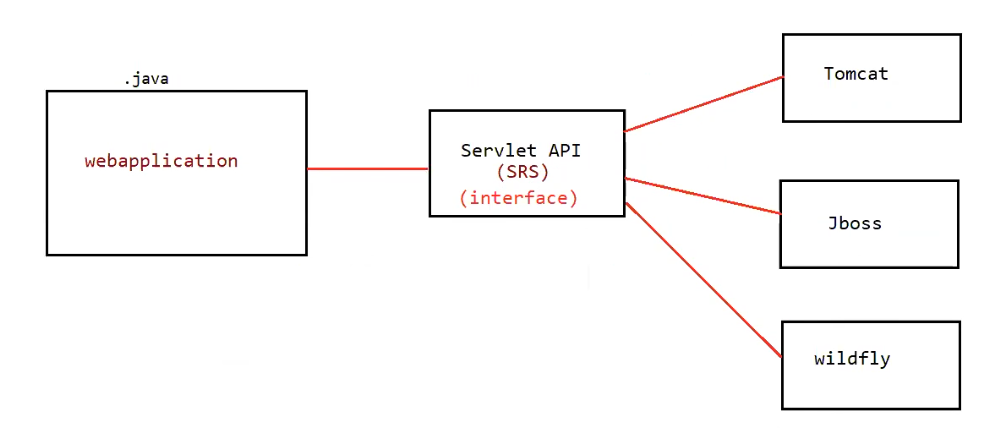
Any SRS (Software Requirement Specification ) is called an interface



Explanation :

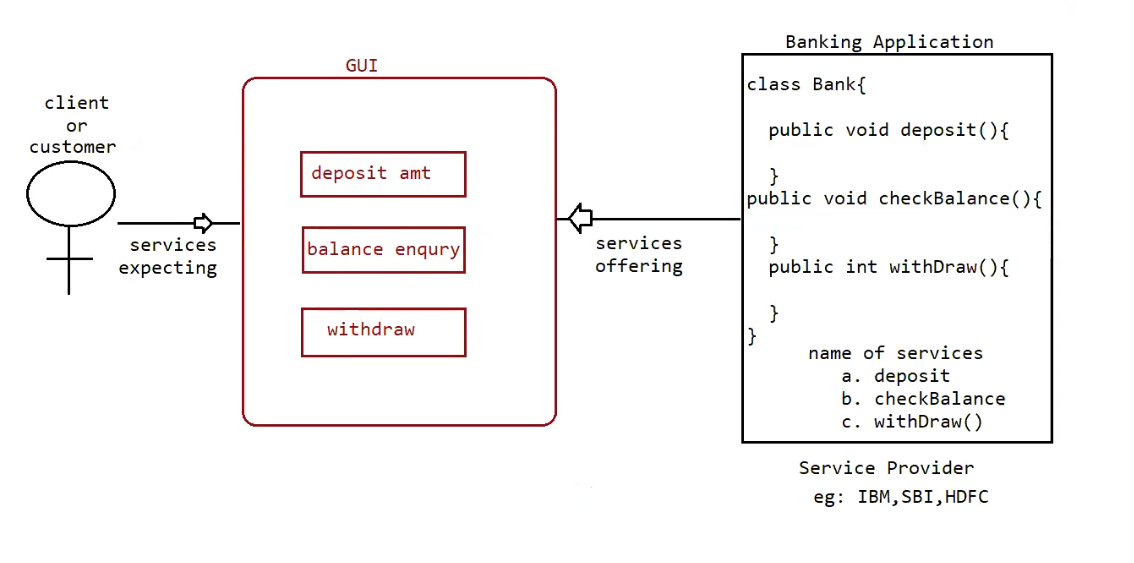
* We write a java program ( .java file ) to connect with the database
* There are different database’s in the market some of them are MySQL , Oracle , PlSQL . To connect with these 3 databases we do not need 3 different java programs , only a single java program is enough to connect with the 3 databases
* To promote this type of specification sun micro system team had given one API( Application Programming Interface , API is collection of .class files , API means someone would write the code we will use it ) JDBC .
* So this can be said as SRS (service requirement specification) .
* In java to represent this srs we use Interface .

Example : 2



* Here this .java file is a web application , it does not run on jvm , this application runs on the server. There are no of servers , some of them are Tomcat , Jboss , Wildfly .
* To promote this type of specification java team had given one API called Servlet API
* So this can be said as an SRS
* In java to represent srs we use interface.

Example : 3



Def:

From the client point of view an interface define the set of services what is expecting.

From the service provider point of view an interface define the set of services what is offering

So interface acts as a contract ( for that contract only we will write the code ) between the client and service provider

Eg:: GUI screen of ATM defines the set of services what the customer is expecting,

Bank offered the same set of services what customer is expecting.

Through class we can have abstract methods and concrete methods ( methods that have logic) , so 100% abstraction is not possible through class .

Inside Interface every method is always abstract whether we are declaring with abstract keyword or not.

But through Interface 100% abstraction is possible.

Eg: Interface Account{

// it is 100% abstract class

// the methods are by default ( no need to specify manually) “ abstract and public “

void withDraw();

void deposit();

void checkBalance();

Note : In abstract class if we write void withDraw() its access modifier is default.

Interface corresponds to Service Requirement Specification (SRS) or contract between service provider (or) 100% pure abstract class.

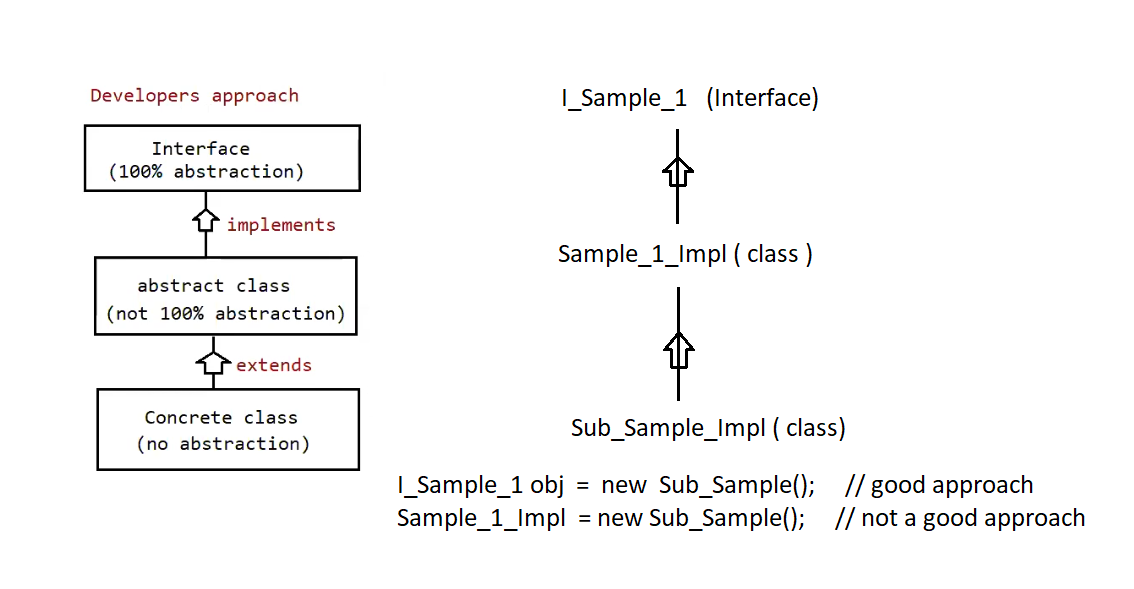
If any one of this requirement you are getting in your java code then use Interface.

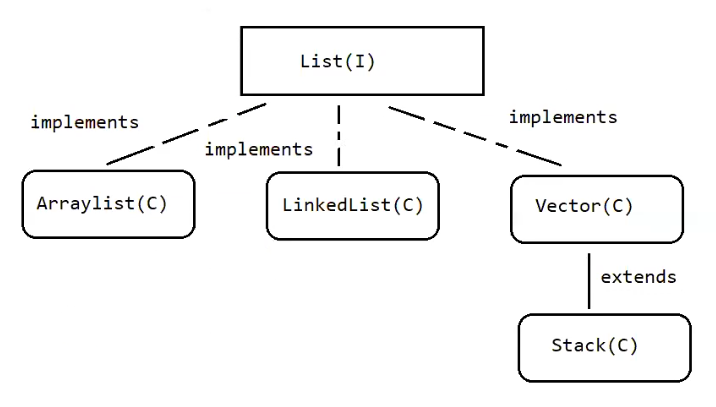
* Interface name should be preceded with "I" , to differentiate b/w class name and interface name
* since interface methods needs to be overridden , the class name that overrides should be same name as Interface , but prefix “I” should be removed and , “Impl” should be added at postfix
* “implements” keyword is used after the class name for the implementation for the interface
* If methods of interface is not overridden in the class , then by default the method that is not overridden will be in the class and access modifier “public” and keyword “abstract” will be with the method in class.
* If a class contain atleast one abstract class , then the class should also be abstract.
* If all the methods in interface are overridden manually in the class then overridden rules must be followed . and “public” access modifier should be preceded with overridden methods , because if we don’t write access modifier in the class it is marked with “default” modifier but the in interface “public” modifier is used by default . so visibility decreases which is not the rule of overriding , to avoid this overridden methods should also be public.
* Whenever we are implementing an interface compulsorily for every method of that interface, we should provide implementation otherwise we have to declare class as abstract class in that case child class is responsible to provide implementation for remaining methods.
* Whenever we are implementing interface method it should be marked as public otherwise it would result in compile time error.

Eg: Interface\_Eg1

Eg: Interface\_Eg2

// go through the code

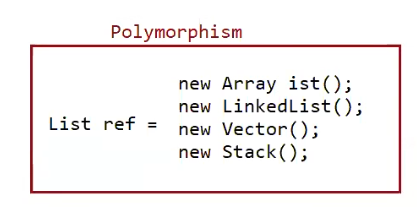




At runtime I don’t the object what is coming I only know

So reference should be of type list interface = object is of list implementation class

Assume at the left side reference of type Arraylist and object at right is of LinkedList (or) Stack then that would result in the error.



To avoid errors we need reference of parent type .