**How to achieve Polymorphism in Java ?**

In java programming the Polymorphism principal is implemented with method overriding concept of java.

**Polymorphism principal is divided into two sub principal they are:**

Static or Compile time polymorphism

Dynamic or Runtime polymorphism

Static polymorphism in Java is achieved by method overloading and

Dynamic polymorphism in Java is achieved by method overriding

**Note: Java programming does not support static polymorphism because of its limitations and java always supports dynamic polymorphism.**

**Example of Runtime Polymorphism in Java**

In below example we create two class Person an Employee, Employee class extends Person class feature and override walk() method. We are calling the walk() method by the reference variable of Parent class. Since it refers to the subclass object and subclass method overrides the Parent class method, subclass method is invoked at runtime. Here method invocation is determined by the JVM not compiler, So it is known as runtime polymorphism.

Example of Polymorphism in Java

class Person

{

void walk()

{

System.out.println("Can Run....");

}

}

class Employee extends Person

{

void walk()

{

System.out.println("Running Fast...");

}

public static void main(String arg[])

{

Person p=new Employee(); //upcasting

p.walk();

}

}

**Dynamic Binding**

Dynamic binding always says create an object of base class but do not create the object of derived classes. Dynamic binding principal is always used for executing polymorphic applications.

The process of binding appropriate versions (overridden method) of derived classes which are inherited from base class with base class object is known as dynamic binding.

**Advantages of dynamic binding along with polymorphism with method overriding are.**

Less memory space

Less execution time

More performance

Advantages of Dynamic Polymorphism

Dynamic Polymorphism allows Java to support overriding of methods which is central for run-time polymorphism.

It allows a class to specify methods that will be common to all of its derivatives while allowing subclasses to define the specific implementation of some or all of those methods.

It also allows subclasses to add its specific methods subclasses to define the specific implementation of same.

Static polymorphism

The process of binding the overloaded method within object at compile time is known as Static polymorphism due to static polymorphism utilization of resources (main memory space) is poor because for each and every overloaded method a memory space is created at compile time when it binds with an object. In C++ environment the above problem can be solve by using dynamic polymorphism by implementing with virtual and pure virtual function so most of the C++ developer in real worlds follows only dynamic polymorphism.

**Dynamic polymorphism**

In dynamic polymorphism method of the program binds with an object at runtime the advantage of dynamic polymorphism is allocating the memory space for the method (either for overloaded method or for override method) at run time.

**Conclusion**

The advantage of dynamic polymorphism is effective utilization of the resources, So Java always use dynamic polymorphism. Java does not support static polymorphism because of its limitation

**Compile time vs Runtime**

Compile-time and Runtime are the two programming terms used in the software development. Compile-time is the time at which the source code is converted into an executable code while the run time is the time at which the executable code is started running. Both the compile-time and runtime refer to different types of error.

Compile-time errors

Compile-time errors are the errors that occurred when we write the wrong syntax. If we write the wrong syntax or semantics of any programming language, then the compile-time errors will be thrown by the compiler. The compiler will not allow to run the program until all the errors are removed from the program. When all the errors are removed from the program, then the compiler will generate the executable file.

Why we use git