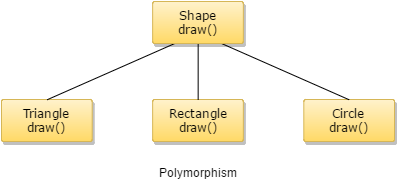
**POLYMORPHISM**

If one task is performed by different ways, it is known as polymorphism

The word polymorphism means having multiple forms. The term Polymorphism gets derived from the Greek word where poly + morphos where poly means many and morphos means forms.

* Static Polymorphism(Compile time)
* Dynamic Polymorphism.(Run time)



Polymorphism is another special feature of [object-oriented programming (OOPs)](http://www.w3schools.in/java-tutorial/object-oriented-programming-oops/). The approach which lies beneath this concept is "single interface with multiple implementations." It offers a single interface for controlling access to a general class of actions.

**Polymorphism can be achieved in two of the following ways:**

* Method Overloading(Compile time Polymorphism)
* Method Overriding(Run time Polymorphism)

**METHOD OVERLOADING:**

Two or more methods with same name under the same class but different parameters.

**Different ways to overload the method**

There are two ways to overload the method in java

1. By changing number of arguments
2. By changing the data type

**By changing number of arguments**

**public** **class** OverloadingEx {

**void** add(**int** a, **int** b) {

System.***out***.println(a+b);

}

**void** add(**int** a) {

System.***out***.println(a);

}

**public** **static** **void** main(String[] args) {

OverloadingEx e=**new** OverloadingEx();

e.add(10,15);

e.add(20);

}

}

**By changing the data type**

**public** **class** OverloadingEx {

**void** add(**int** a, **int** b) {

System.***out***.println(a+b);

}

**void** add(**int** a, **char** b) {

System.***out***.println(a+b);

}

**public** **static** **void** main(String[] args) {

OverloadingEx e=**new** OverloadingEx();

e.add(10,15);

e.add(20, 'h');

}

}

**METHOD OVERRIDING**

If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in Java**.

Two or more methods with same name, same parameter but different classes.

Example

**class** sample{

**void** add() {

System.***out***.println("hi");

}

}

**public** **class** OverridingEx **extends** sample {

**void add() {**

**System.*out*.println("hello");**

**}**

**public** **static** **void** main(String[] args) {

OverridingEx o=**new** OverridingEx();

o.add();

sample s=**new** sample();

s.add();

}

}