SYNTAX TECHNOLOGIES

JAVA

Class 21

Agenda

Method Overloading

Method Overloading

Whenever same method name is existing multiple times in the same class with different number of parameter or different order of parameters or different types of parameters is known as method overloading

Java does **not allow overloading by changing the return type only**, though overloaded methods can have different return types.

Method Overloading

Syntax:

```
class  class_Name {

    Returntype  method(datatype1 variable1) {
    ………}

    Returntype  method (datatype1 variable1) {
    ………}

}
```

In java, method overloading is not possible by just changing the return type of the method

Method Overloading

There are three ways to overload the method in java:

1. By changing number of parameters
2. By changing the data type
3. By changing the sequence/order of parameters

Different ways to overload the method

By changing number of arguments

In this example, we have created two overloaded methods, first sum method performs addition of two numbers and second sum method performs addition of three numbers.

```
class Addition
{
  void sum(int a, int b) {
   SOP(a+b);
  }
  void sum(int a, int b, int c) {
  SOP(a+b+c);
 }
public static void main(String args[]) {
Addition obj=new Addition();
obj.sum(10, 20);
obj.sum(10, 20, 30);
}
}
```

By changing the data type

In this example, we have created two overloaded methods that differs in data type. The first sum method receives two integer arguments and second sum method receives two float arguments.

```
class Addition
{
 void sum(int a, int b) {
 SOP(a+b);
}
 void sum(float a, float b) {
 SOP(a+b);
}
public static void main(String args[]) {
Addition obj=new Addition();
obj.sum(10, 20);
obj.sum(10.05, 15.20);
}
}
```

Advantage of Method Overloading

- The main advantage of this is cleanliness of code.
- Method overloading increases the readability of the program.
- Flexibility
- Overloaded methods give programmers the flexibility to call a similar method for different types of data. If you are working on a mathematics program, for example, you could use overloading to create several multiply classes, each of which multiplies a different number of type of argument: the simplest multiply(int a, int b) multiplies two integers; the more complicated method multiply(double a, int b, int c) multiplies one double by two integers -- you could then call "multiply" on any combination of variables that you created an overloaded method for and receive the proper result.