OBJECT ORIENTED PROGRAMMING

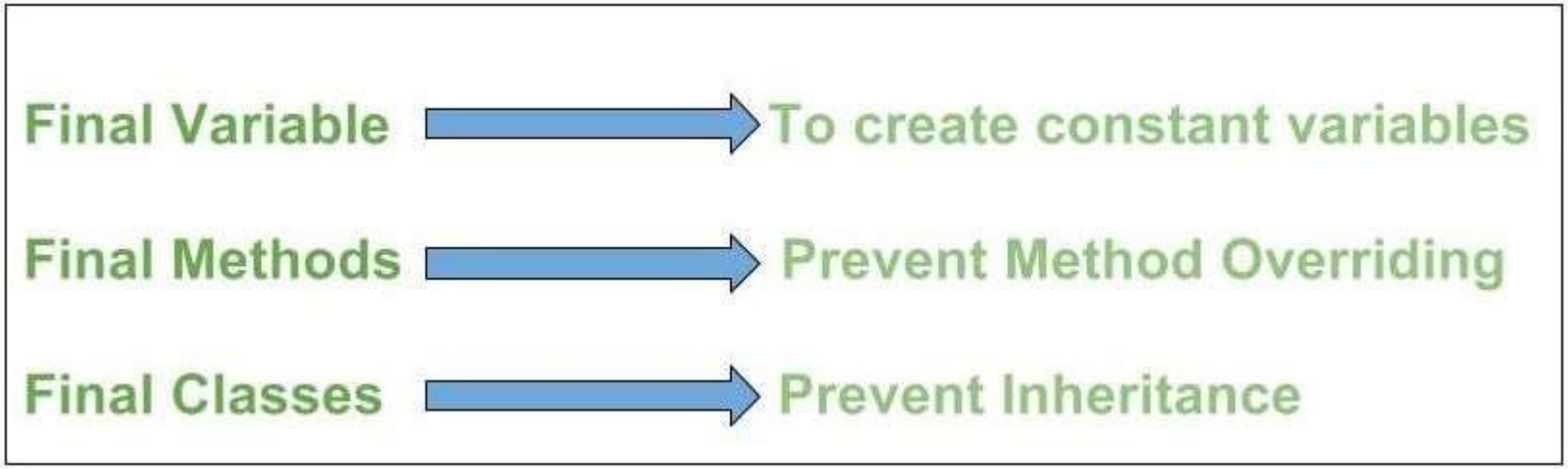
Affefah Qureshi

Department of Computer Science

Iqra University, Islamabad Campus.

FINAL

- The **final keyword** in java is used to restrict the user.
- The java final keyword can be used in many context.
  1. variable
  2. method
  3. class



FINAL VARIABLE

- A variable can be declared as final.
- Doing so prevents its contents from being modified.
- This means that you must initialize a final variable when it is declared.
- It is a common coding convention to choose all uppercase identifiers for **final** variables.
- Thus, a final variable is essentially a constant.
- A field that is both static and final has only one piece of storage that cannot be changed.
- The keyword final can also be applied to methods, and classes.
- A final variable that is not initialized at the time of declaration is known as blank final variable. We can initialize using constructors only.
- A static final variable that is not initialized at the time of declaration is known as static blank final variable. It can be initialized only in static block.

EXAMPLE

- final int FILE_NEW = 1;
- final int FILE_OPEN = 2;
- final int FILE_SAVE = 3;
- final int FILE_SAVEAS = 4;
- final int FILE_QUIT = 5;

```
class A{
  static final int data;//static blank final variable
  static{ data=50;}
  public static void main(String args[]){
    System.out.println(A.data);
 }}
```

Output:

50

FINAL VARIABLE

- If you declare any parameter as final, you cannot change the value of it.
- Can we declare a constructor final? No
- No, because constructor is never inherited.

```
class Bike11{
  int cube(final int n){
   n=n+2;  //can't be changed as n is final
   return n*n*n;
  }
  public static void main(String args[]){
    Bike11 b=new Bike11();
    System.out.println(b.cube(5));
  }
}
```

Output:

Compile Time Error

FINAL METHOD

Output:

Compile Time Error

- If you make any method as final, you cannot override it.
- Is final method inherited? Yes, final method is inherited but you cannot override it.

```
class Bike{
    final void run(){
            System.out.println("running"); } }

class Honda extends Bike{
    void run(){
            System.out.println("running safely with 100kmph"); }
    public static void main(String args[]){
            Honda honda= new Honda(); honda.run(); } }
```

FINAL CLASS

Output:

Compile Time Error

If you make any class as final, you cannot extend it.

```
final class Bike{
        void run(){
                System.out.println("running");}}
class Honda extends Bike{
        void run(){
                System.out.println("running safely with 100kmph");
    }

        public static void main(String args[]){
                Honda honda= new Honda();
                honda.run();}}
```