FAST

National University of Computer and Emerging Sciences Peshawar

Lecture # 11

Software Construction and Development (Java Programming)

Instructor: Muhammad Abdullah Orakzai

DEPARTMENT OF COMPUTER SCIENCE

الذی علم بالقلم۔ علم الانسان ما لم یعلم۔

Object Oriented Programming (this and static keyword)

Contents

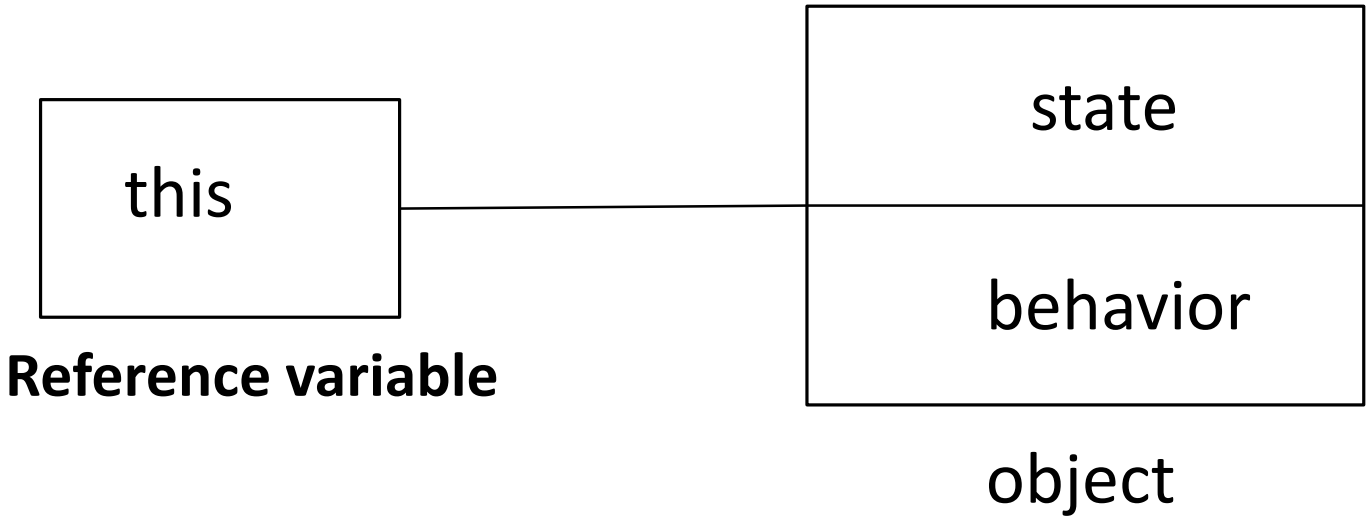
this keyword

- There can be a lot of usage of **this** keyword.
- In java **this** keyword is reference variable that refers to the current class object.

Usage of this keyword

1. **this** keyword can be used to refer current class instance variable.
2. **this()** can be used to invoke current class constructor.
3. **this** keyword can be used to invoke current class method (implicitly).
4. **this** keyword can be passed as an argument in the method call.
5. **this** keyword can be passed as an argument in the constructor call.
6. **this** keyword can also be used to return the current class instance.

Usage of this keyword



1) this keyword to refer current class instance variable

- If there is an ambiguity between the instance variable and the parameter, this keyword resolve the problem of ambiguity.
- this keyword is used to distinguish between local variables and instance variables.

Program

```
class Student
{
int id;
String name;
```

1) this keyword to refer current class instance variable…

```
Student(int id,   String name)
{
this.name   = name;
this.id   =   id;
}
public void display()
{
System.out.println(id + “ ” + name);
}
```

1) this keyword to refer current class instance variable…

```
public static void main(String args[])
{
Student s1   = new  Student(11,  “Akram”  );
Student s2   = new  Student(12,  “Asim”  );
 s1.display();
s2.display();
}
}   // class body end
```

1) this keyword to refer current class instance variable…

Note

1) If local variable (formal arguments) and instance variables are different , then there is no need to use this keyword.

2) this keyword is used to distinguish between local variables and instance variables.

2) this keyword to invoke current class constructor

- this() keyword is used to invoke current class constructor (constructor chaining).
- This is approach is better if you have many constructors in the class and we want to reuse that constructors.

Constructor Chinning : Calling a constructor from another constructor of the same class.

2) this keyword to invoke current class constructor…

Program

```
class Student
{
int id;
String name;
 Student()
        {
                System.out.println(“Default constructor is called”);
        }
```

2) this keyword to invoke current class constructor…

```
Student(int id,   String name)
{
this()      // used to call current class constructor
this.name   = name;
this.id   =   id;
}
public void display()
{
System.out.println(id + “ ” + name);
}
```

2) this keyword to invoke current class constructor…

```
public static void main(String args[])
{
Student s1   = new  Student(13,  “Akram”  );
Student s2   = new  Student(14,  “Asim”  );
 s1.display();
s2.display();
}
}   // class body end
```

2) this keyword to invoke current class constructor…

Where to use this() constructor call

The this() constructor call should be used to reuse the constructor in the constructor. It maintains chain between the constructor i.e. it is used for constructor chaining.

Program

```
class Student

{

int id;

String name;

String city;
```

2) this keyword to invoke current class constructor…

```
Student(int id,   String name)
{
this.name   = name;
this.id   =   id;
}
Student(int id,  String name, String city)
{
this(id ,  name);                    // no need to initialize id and name
this.city   = city;
}
```

2) this keyword to invoke current class constructor…

```
public void display()
{
System.out.println(id + “ ” + name+ “ ” + city);
}
public static void main(String args[])
{
Student s1   = new  Student(13,  “Akram”  );
Student s2   = new  Student(14,  “Asim”, “Peshawar”  );
```

2) this keyword to invoke current class constructor…

```
s1.display();

s2.display();

}

}   // class body end
```

Note

Call to this() must be the 1st statement in constructor otherwise you will get compile time error.

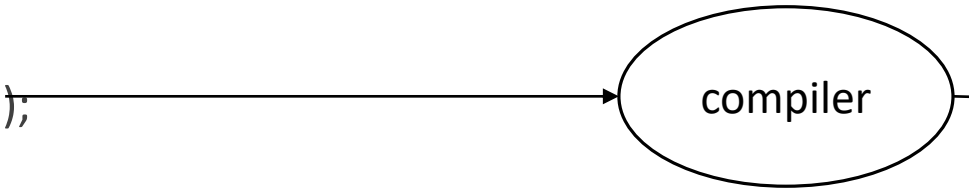
3) this can be used to invoke current class method (implicitly)

- You may invoke the method of the current class by using this keyword.

- If you do not use this keyword compiler automatically adds this keyword while invoking the methods.

3) this can be used to invoke current class method (implicitly)…

```
class A{
void  m()  { }
void  n()
{
m();
}
public static void main(String args[])
{
new A().n();
}
}  // class A end
```




```
class A{

void  m()  { }
void  n()
{
this.m();
}
public static void main(String args[])
{

new A().n();
}
}  // class A end
```

3) this can be used to invoke current class method (implicitly)…

Program

```
class A{
void  m()
{
System.out.println(“method m is invoked”)
}
void  n()
{
this.m();    // no need because compiler does it for you
}
```

3) this can be used to invoke current class method (implicitly)

```
void  p()
{
n();  // compiler will add this to invoke n() method as  “this.n()”
}
public static void main(String args[])
{
A a   = new A();
a.p();
a.n();
}
}   // class A end
```

Output

method m is called
method m is called

Static keyword

- Java static keyword is used for memory management mainly.
- We can apply java static keyword with variables , methods, blocks and nested class.
- This static keyword belongs to the class than instance of the class.

Static keyword…

This can be:

1. Variable (also known as class variable)
2. Method (also known as class method)
3. Block
4. Nested class

Normally to access member of class

Instance variable

obj.variableName = value;

Eg: obj.rollNo = 1525;

Instance method

obj.methodName(parameter list); // parameter list is optional

Eg: obj.getData();

Note: Without object of class we cannot access instance methods and instance variable.

static variable

- If you declare any variable as static , it is called static variable.
- The static variable can be used to refer the common property of all objects (that is not unique for each object)

Example: 1) Company name of employees.

2) College name of students.

- The static variable gets memory only once in a class area at the time of class loading.
- Static variable is shared by all the objects.

static variable…

- Static variable is not related with individual objects but related to whole class.
- No need to create object for accessing static variable. It can be accessed directly or by a class name.

Advantages:

It makes your memory efficient (it saves memory).

static variable…

Declaration of static variable

static datatype variableName;

Eg: static int rollNo; // class variable

Accessing static variable

ClassName.variableName = value;

Eg: Student.rollNo =109 ;

Java static method

- If you apply static keyword with any method , it is known as static method.
- A static method belong to the class rather than object of a class.
- A static method can be invoked without the need of creating an instance (object) of a class i.e. can be invoked directly or with class name.
- Static method can access static data member and can change the value of it.

Java static method…

Declaration of static method

static methodName(parameter list); // parameter list is optional

Eg: static void getData(); // class variable

Accessing static method

ClassName.methodName();

Eg: Student.getData();

static variable & method vs instance variable & method

- copy of instance variable is created in all objects.
- Copy of static variable is not created but and it is shared by all the objects.
- When we create variables and methods in a class then it is called instance variable and instance methods, because for each and every object separate copy is created.
- Static variable is called class variable because it is related to the whole class.

static variable & method vs instance variable & method

- For static method and static variable only one memory is allocated and it is not allocated for every object.
- Both static method and static variable are accessed directly or with class name.

Note: Java static property is shared by all the objects

Static variable Example

```
class Student{
int rollNo;
String name;
Static String college = “ICP”;
Student(int r, String n)
{
rollNo=r;
name= n;
}
```

Static variable Example…

```
void display()
{
System.out.println(rollNo+ “ ” + name+ “ ”+ college);
}
public static void main(String args[])
{
Student s1 = new Student(11, “Ali”);
Student s2 = new Student(12, “Asad”);
s1.display();
s2.display();
}
}
```

Output

11 Ali ICP

12 Asad ICP

Program counter without static variable

```
class  Counter{
int count = 0; // will get memory when object or instance is created
Counter()
{
count++;
System.out.println(count);
}
}
```

Program counter without static variable…

```
public static void main(String args[])
{
Counter c1 = new Counter();
Counter c2 = new Counter();
Counter c3 = new Counter();
}
}
```

Output

1
1
1

Program counter by static variable

As we have mentioned static variable will get memory only once, if any object changes the value of static it will retain its value.

Program counter by static variable

```
class  Counter{
static int count = 0;       // will get memory at once and retain its value.
Counter()
{
count++;
System.out.println(count);
}
}
```

Program counter by static variable…

```
public static void main(String args[])
{
Counter c1 = new Counter();
Counter c2 = new Counter();
Counter c3 = new Counter();
}
}
```

Output

1
2
3

Example of static method

Program of changing the common property of all objects (static field).

```
class Student{
int rollNo;
String name;
Static String college = “Islamia College Peshawar”;
static void change()      //static method
{
college  = “Edward College Peshawar”;
}
```

Example of static method…

```
Student(int r, String n)          // constructor
{
rollNo=r;
name= n;
}
void display()
{
System.out.println(rollNo+ “ ” + name+ “ ”+ college);
}
}
```

Example of static method…

```
public static void main(String args[])
{
Student s1 = new Student(11, “Ali”);
Student s2 = new Student(12, “Asad”);
Student s3 = new Student(13, “Alim”);
s1.display();
s2.display();
s3.display();
}
}
```

Output

11 Ali Edward College Peshawar
12 Asad Edward College Peshawar
13 Alim Edward College Peshawar

Example of static method that perform normal calculation

```
class  Calculate{
static int cube(int x)
{
return x*x*x;
}
  public static void main(String args[])
  {
  int result  = Calculate.cube(5);   // method call with class name
  System.out.println(result);
  }
  }
```

Restriction of static method

- The static method cannot use non static data member or call non-static method directly.
- this and super cannot be used in static context.

Example:

```
class A{
int  a =4 ;  // non static
 public static void main(String args[])
 {
 System.out.println(a);      // compile time error
 }
```

Output

Compile time error

Java static block

- It is used to initialize the static data member.
- It is executed before main method at the time of class loading.

Java static block…

Example:

```
class A{
Static
{
System.out.println(“Static Block is invoked”);
}
 public static void main(String args[])
 {
 System.out.println(“Hello main”);}
```

Output

Static Block is invoked
Hello main

Why java main method is static

Because object is not required to call static method if it was not static method, JVM create object first then call main() method that will lead the problem of extra memory allocation

THANK YOU