IIB , SIB, COnstructors and Main method examples

note:

Always SIB will run first then main method and if object is created then IIB and finally constructor

would run

Example 1:

package sibandiib;

public class A {

static

{

System.out.println("From SIB");

}

{

System.out.println("From IIB");

}

public static void main(String[] args) {

System.out.println("From main");

}

}

Output:

From SIB

From main

Example 2:

public class A {

static

{

System.out.println("From SIB");

}

{

System.out.println("From IIB");

}

public static void main(String[] args) {

System.out.println("From main");

A a = new A();

}

}

Output:

From SIB

From main

From IIB

Example 3:

public class A {

static

{

System.out.println("From SIB");

}

{

System.out.println("From IIB");

}

A(){

System.out.println("From Constructor");

}

public static void main(String[] args) {

System.out.println("From main");

A a = new A();

}

}

Output:

From SIB

From main

From IIB

From Constructor

Example 4:

public class A {

static

{

System.out.println("From SIB");

}

{

System.out.println("From IIB");

}

A(){

System.out.println("From Constructor");

}

{

System.out.println(100);

}

public static void main(String[] args) {

System.out.println("From main");

A a = new A();

}

}

Output:

From SIB

From main

From IIB

100

From Constructor

Example 5:

public class A {

static

{

System.out.println("From SIB");

A a1 = new A();

}

{

System.out.println("From IIB");

}

A(){

System.out.println("From Constructor");

}

public static void main(String[] args) {

System.out.println("From main");

}

}

Output:

From SIB

From IIB

From Constructor

From main

this keyword in java ?

1. this keyword is a special reference variable that stores objects address and it is automatically

created by java compiler

Example 1:

public class A {

public static void main(String[] args) {

A a1 = new A();

System.out.println("Address of a: "+a1);

a1.test();

}

public void test(){

System.out.println("Address of this: "+this);

}

}

Ouput:

Address of a: sibandiib.A@2a139a55

Address of this: sibandiib.A@2a139a55

Example 2:

public class A {

int i =10;

public static void main(String[] args) {

A a1 = new A();

System.out.println(a1.i);

a1.test();

}

public void test(){

System.out.println(this.i);

}

}

Output:

10

10

Example 3:

public class A {

public static void main(String[] args) {

A a1 = new A();

a1.test();

}

public void test(){

this.x();

}

public void x(){

System.out.println("From x method()");

}

}

Output:

From x method()

2. this keyword points to current object running

Example 1:

public class A {

public static void main(String[] args) {

A a1 = new A();

System.out.println(a1);

a1.test();

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

A a2 = new A();

System.out.println(a2);

a2.test();

}

public void test(){

System.out.println(this);

}

}

Output:

sibandiib.A@2a139a55

sibandiib.A@2a139a55

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

sibandiib.A@15db9742

sibandiib.A@15db9742

Limitations of this keyword ?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. we cannot use this keyword inside static blocks / methods/ context

Example 1:

public class A {

public static void main(String[] args) {

A a1 = new A();

a1.test();

}

public static void test(){

System.out.println(this);//Error

}

}

Output:

Error

Example 2:

public class A {

public static void main(String[] args) {

A a1 = new A();

}

static{

System.out.println(this);//Error

}

}

Output:

Error

Example 3:

public class A {

public static void main(String[] args) {

A a1 = new A();

A a2 = new A();

}

{

System.out.println(this);

}

}

Output:

sibandiib.A@2a139a55

sibandiib.A@15db9742

Calling a constructor using this keyword:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. calling a constructor using this keyword can be done only when we use this keyword in another

constructor

2. this keyword should be always first statement inside a constructor while calling another

constructor

Example 1:

public class A {

A(int i){//Step 2

this();//Step 3//use parenthesis with this to constructor

}

A(){//Step 4

System.out.println(500);

}

public static void main(String[] args) {

A a1 = new A(100); //Step 1

}

}

Output:

500

Example 2:

public class A {

A(int i){

System.out.println(i);

}

A(){

this(100);

}

public static void main(String[] args) {

A a1 = new A();

}

}

Output:

100

Example 3:

public class A {

A(int i){

System.out.println(i);

}

A(){

System.out.println("From Cons A");

this(100);//Error

}

public static void main(String[] args) {

A a1 = new A();

}

}

Output:

Error because this() keyword is second statement

Example 4:

public class A {

A(int i){

System.out.println(i);

}

A(){

this(100);

System.out.println("From Cons A");

}

public static void main(String[] args) {

A a1 = new A();

}

}

Output:

100

From Cons A

Example 5:

public class A {

int i = 10;

A(){

System.out.println("From COnstructor");

System.out.println(this.i);

}

public static void main(String[] args) {

A a1 = new A();

}

}

Output:

From COnstructor

10