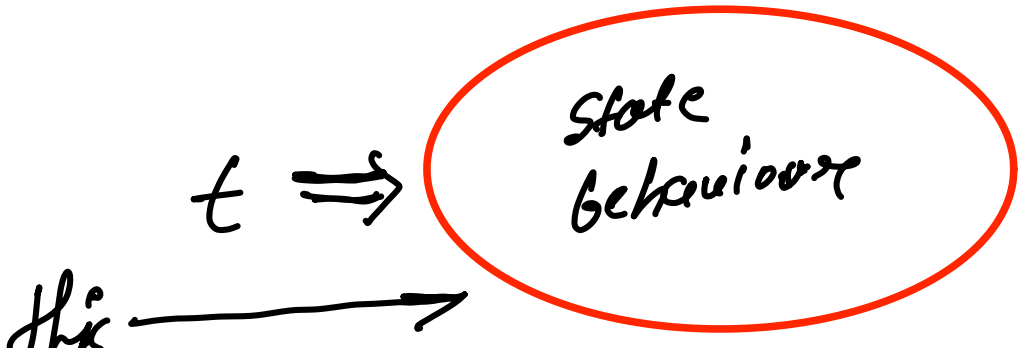
This Keyword

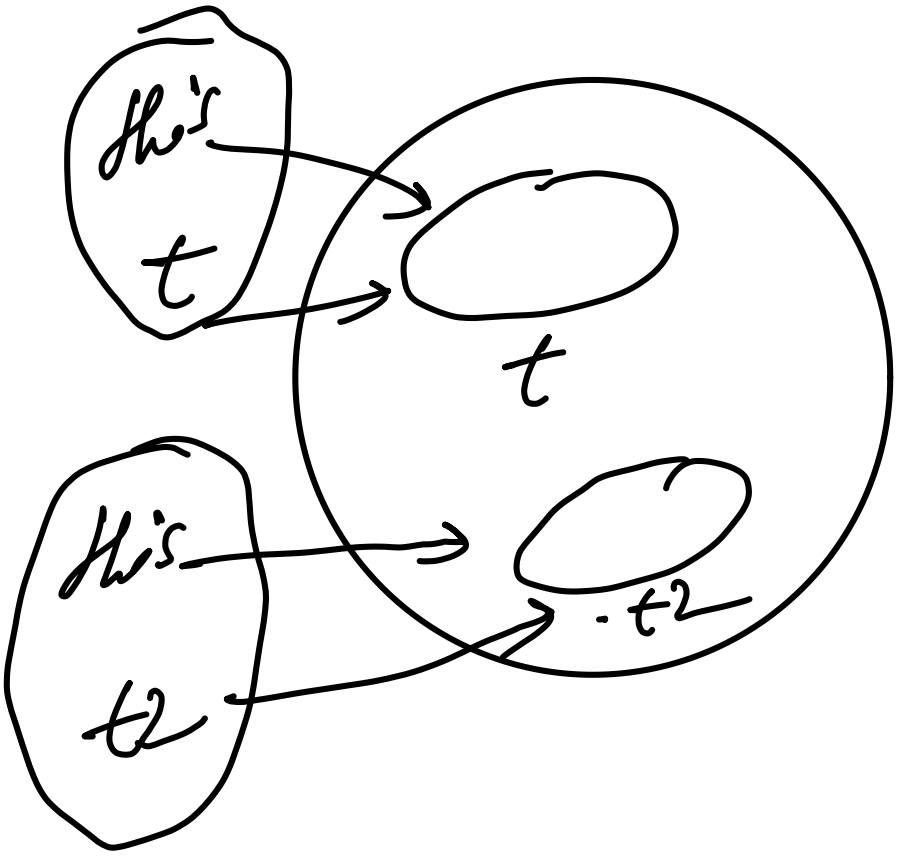
↳ reference Variable that refers to current class object.

```
Test {
    void m() {
        Test t = new Test();
    }
}
```

t ⟹ reference Variable

this ⟹ current class object





① this refers to current class instance variable

```
class Test {
    int no = 10;
    void m1(int no) {
        s.o.p/n(no)      // 20
        s.o.p/n(this.no) // 10
    }
}
```

obj.m(10);

② used to invoke current class method

this.methodname();

```
Test {
    m1() {
        s.o.p("m1");
        this.m2(); //m2()
    }

    m2() {
        s.o.p("m2");
    }

    psvm() {
        Test t = new Test();
        t.m1();
    }
}
```

internally

③ used to invoke current class constructor.

```
class Test {

    Test() {
        S.O.println("1");
        this(10);
    }

    Test(int a) {
        S.O.println("2");
    }

}
```

```
main {
    fsum() {
    Test t2 = new Test( );

    }
}
```

Constructor chaining

```
Test {

    Test() {
        System.out.println("1");
    }

    Test(int a) {
        System.out.println(a);
    }

    Test(String str) {
        System.out.println(str);
    }
}
```

```
fsum() {

Test t1 = new Test();

Test t2 = new Test(10);

Test t3 = new Test("A");
```

④ used to pass as an argument in a method class

```
Test {

    m1() {
        s.o.p("1");
        m2(this);
    }

    m2(Test t) {
        s.o.p(t);
    }

}

psvm() {
    Test t = new Test();
    t.m1();
}
```

⑤ used to pass as an argument in Constructor.

```
Test {
    void m1() {
        Test2 ob = new Test2(this);
    }
}

Test2 {
    Test2(Test t) {
        System.out.println("i");
    }
}
```

6) used to return the current class object

```
Test {
    Test m1() {
        return this;
    }
}

psvm() {
    Test t = new Test();
    Test obj = t.m1();
}
```

Super Keyword

↳ reference Variable which refers to parent class object

① refer to immediate parent class instance variable

```
A {
  int no = 10;
}
B extends A {
  int no = 20;
  void m1(no) {
    s.o.p(no) // 30
    s.o.p(this.no) // 20
    s.o.p(super.no); //10
  }
```

psvm() {

B obj = new B();

obj.m(30);

② used to invoke parent class method.

③ used to call the parent class constructor

↳ super();

& this, or this()

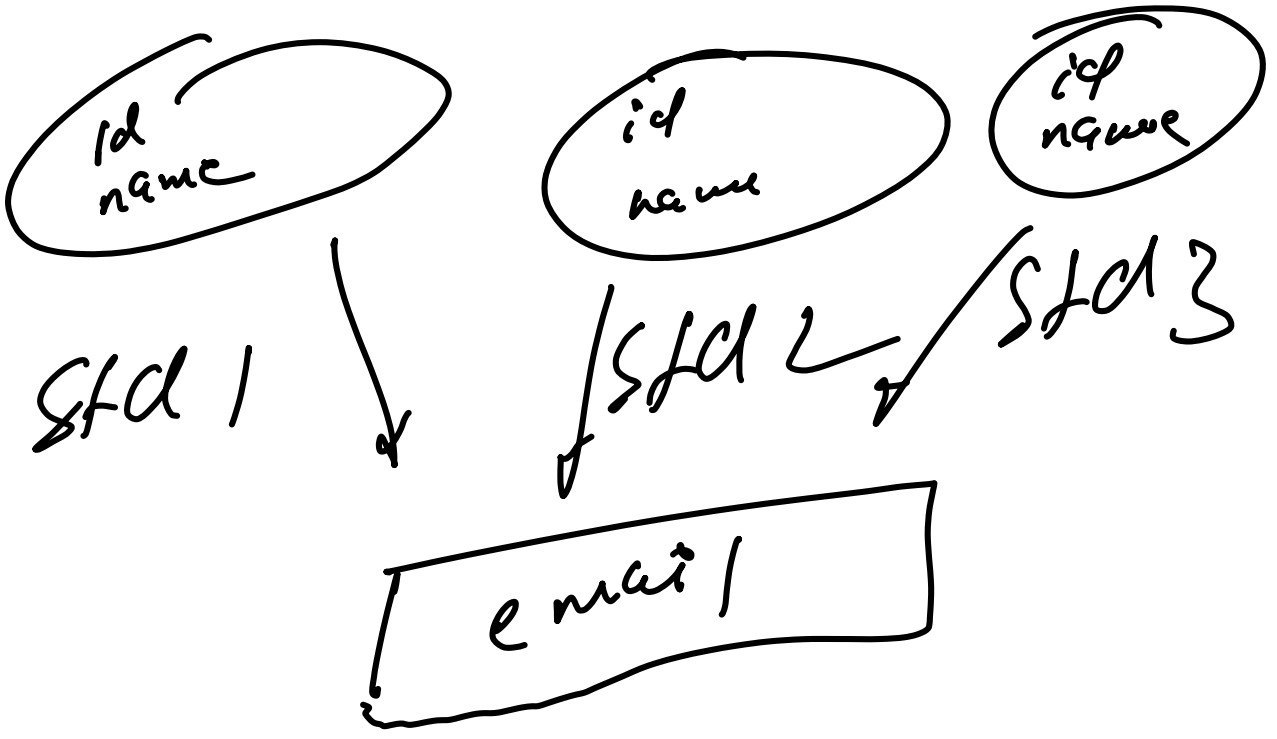
super or super()

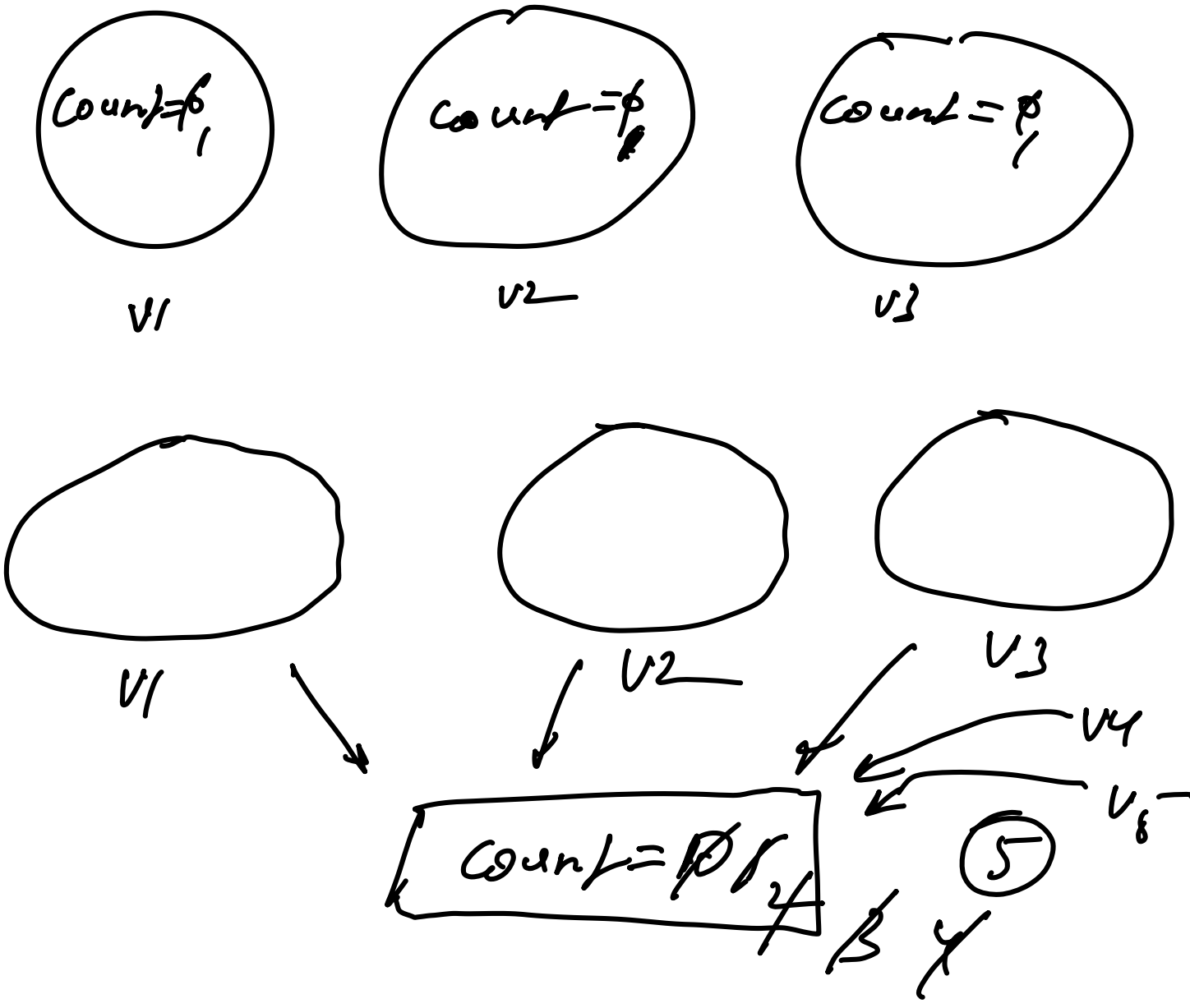
Static Keyword

↳ belongs to class Not object

```
Student {
    int id;
    String name;
    static email = "Arun@gmail.com";
    //Constructor
    //displayMethod
}
```

```
Stu1 = new Stu(1, "A");
Stu2 = new Stu(2, "B");
Stu3 = new Stu(3, "C");
```




count=6
count=6
count=6
V1
V2
V3
V1
V2
V3
V4
V8
count=6
5
B
4

* instance → obj
* static → class Level

int no;

static int a;

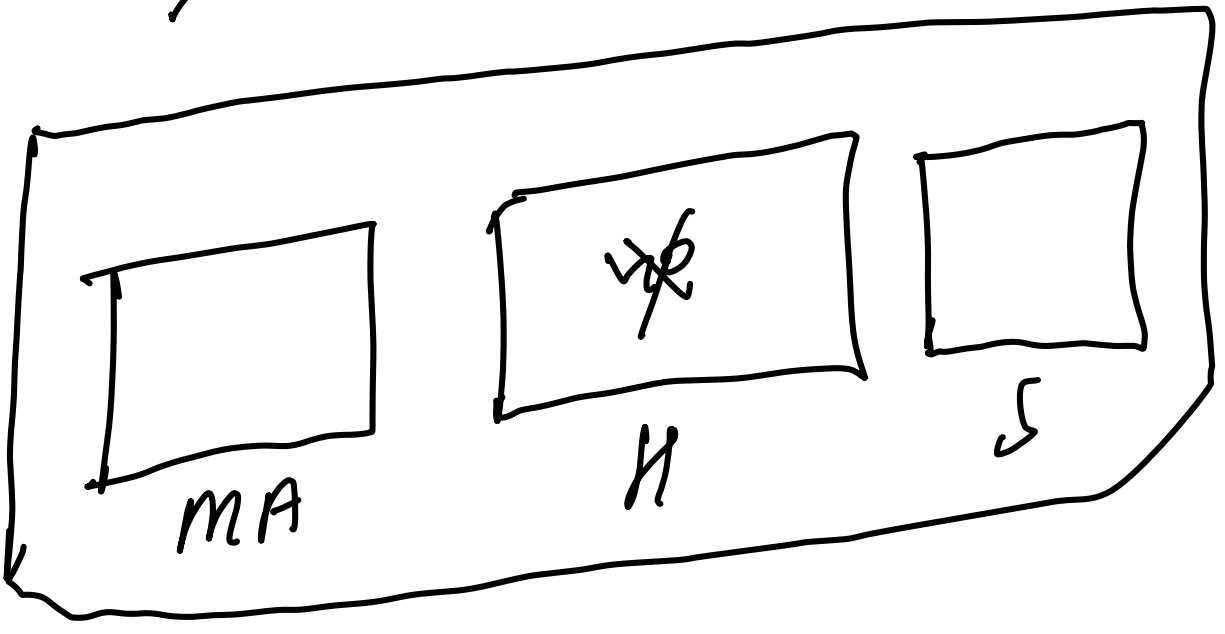
void display() {

S.O.println(a);

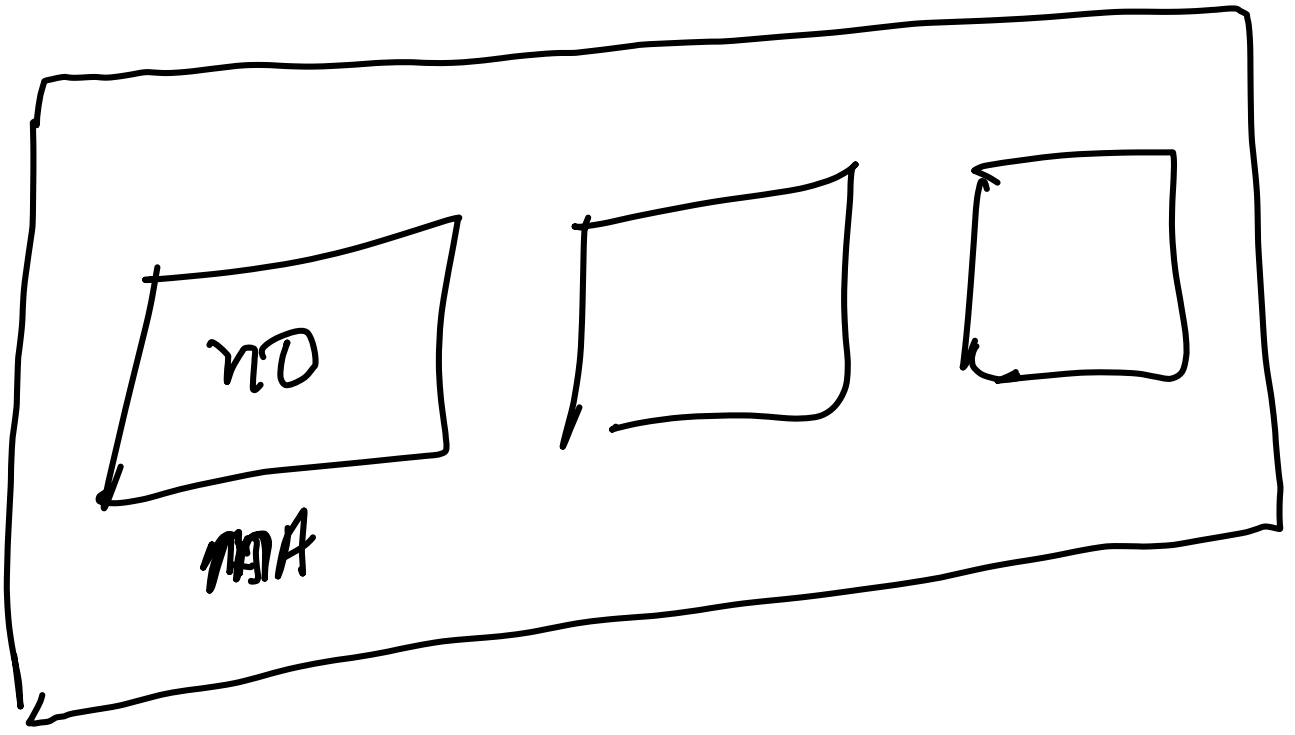
}

① static → Class loading (method Area)
② Instance → object creation (Heap)
③ local → method call (stack)

```
class Demo {
    int no;
    static void m1() {
        S.o.pln(no);
    }

    psvm() {
        Demo.m1();
    }
}
```



```
class Demo {
    static int no;
    void m1() {
        s.o.pln(no);
    }
    psvm() {
        Demo obj = new Demo();
        obj.m1();
    }
}
```



Static Block

static {

}

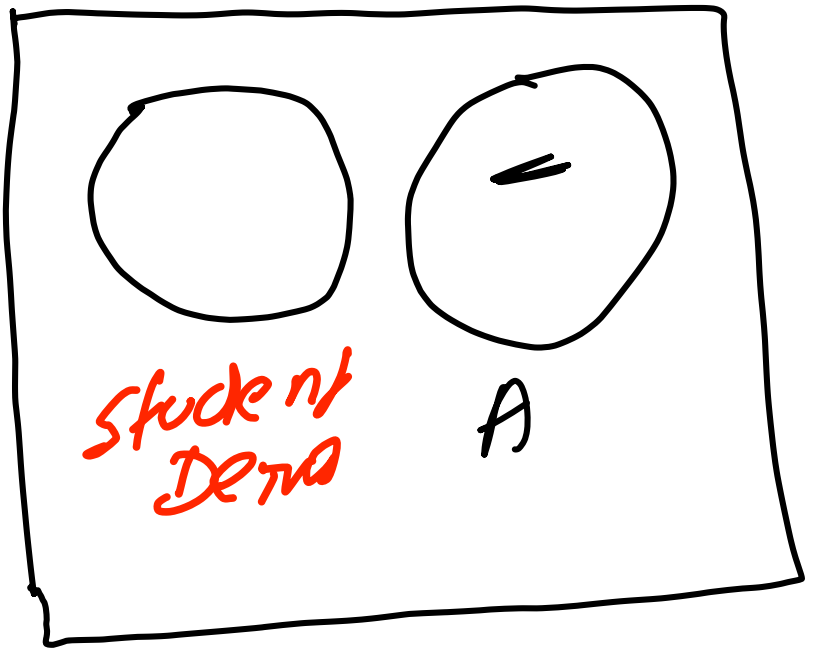
use
- Static Variable initialization
- Native libraries

Loading @ time class loading

```
Class StaticDemo {

    static {
        System.out.println("1");
    }

    psvm() {
        System.out.println("2");
    }

}
```

```
class A {

    static {
        System.out.println("1");
    }

class StaticDemo {

    static {
        System.out.println("2");
    }

    psvm() {
        A a = new A();
        System.out.println("3");
    }

}
```

O/P = 1 2 3
      2 1 3



Q: Why main method is static?

↳ bcoz JVM Call the main method without creating the object.

⇓

memory Management

Static Nested class

↳ Static class inside another class

```
class outer {

    class Inner {
        void sum() {
        }
    }

}

psvm() {
    Outer ob1 = new Outer();
    Outer.Inner ob2 = ob1.new Inner();
    ob2.show();
```

```
public class Outer {
    class Inner{
        void show(){
            System.out.println("inside Inner Class");
        }
    }
}

public class NestedMain {
    public static void main(String[] args) {
        Outer ob1 = new Outer() ;
        Outer.Inner ob2 = ob1.new Inner() ;
        ob2.show();
    }
}
```

==========

```
public class Outer {
    static class Inner{
        void show(){
            System.out.println("inside Inner Class");
        }
    }
}


public class NestedMain {
    public static void main(String[] args) {
        Outer.Inner ob = new Outer.Inner() ;
        ob.show();
    }
}
```

==========

```
public class Outer {
    static class Inner{
        static void show(){
            System.out.println("inside Inner Class");
        }
    }
}

public class NestedMain {
    public static void main(String[] args) {
        Outer.Inner.show();
    }
}
```

==========

Final Keyword

- Variable ✓
- method ✓
- class ✓
- object ✓

→ Final Keyword provide restriction to the user

① Final Variable

↳ Cannot reassign

```
final int a = 10;
s.o.pln(a+10);
```

```
public class FinalDemo1 {
    public static void main(String[] args) {
        final int a = 10 ;
        a = a+10 ;
        System.out.println(a+10);
        System.out.println(a);
    }
}
```

② final Method

↳ Cannot Override

③ final class

↳ final class cannot be inherited but it can inherit other classes.

(9) Object

↳ Cannot refer to any other object