

Object Meets Function

Everything is Object

Steven Lolong

`steven.lolong(at)uni-tuebingen.de`

**Programming Language Research Group
Tuebingen University**

Dec. 18, 2023



Everything is Object

- 
- 1 Introduction
 - 2 Class
 - 3 Object
 - 4 Trait

- 5 Abstract Class
- 6 Case Class
- 7 Enumeration
- 8 Case Study



Object

What is it?

Object in Computer Science can be a variable, a data structure, a function, or a method.

Object in OOP can be a combination of variables, functions, and data structure.

Object in Scala is a group of variable(s) and / or function(s).



Class

Definition

```

1 class ClassName[typParam](param) extends ext :
2   constructorPlace
3   scope var / val varName: vType
4   scope def methodName(methodParam): methodRetType =
5 end ClassName

```

ClassName the name of class.

typParam type parameter(s), separated by comma.

param parameter(s) of constructor, separated by comma.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Class

Property

- Can be instantiated. Since Scala 3, the **new** keyword is optional (just for compatibility).
- Can have parameter for constructor.
- One class can **inherit** only one parent class.



Class

Demo

- Constructor.
- Scope in constructor.
- Inheritance.



Object

Definition

```

1 object ObjectName extends ext :
2   constructorPlace
3   scope var / val varName: vType
4   scope def methodName(methodParam): methodRetType =
5 end ObjectName

```

ObjectName the name of object and its intance.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Object

Property

- Instantiate automatically when it was abstracted
- A singleton class (instance name is same with class name)
- Can't inheritance by other object/ class.
- Can't use in mixin.
- Doesn't support parameter for constructor.



Object

Demo

DEMO



Trait

Definition

Ideally, trait is pure class. Pure class is class abstraction without detail (concrete) definition.

```

1 trait TraitName [typParam] extends ext :
2     constructorPlace
3     scope var / val varName: vType
4     scope def methodName(methodParam): methodRetType =
5 end TraitName
    
```

TraitName the name of trait.

typParam type parameter(s), separated by comma.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Trait

Property

- Cannot be instantiated.
- Can use in mixin for multiple inheritance.
- Parameter for constructor (only on Scala 3).
- Supports type parameter.



Trait

Demo

DEMO



Abstract Class

Definition

```

1 abstract class ClassName [typParam] (param) extends ext :
2     constructorPlace
3     scope var / val varName: vType
4     scope def methodName(methodParam): methodRetType =
5 end ClassName
    
```

ClassName the name of class.

typParam type parameter, separated by comma.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Abstract Class

Property

- Cannot be instantiated.
- Can have parameter for constructor.
- One class can inherit only one parent class.
- Can't use in mixin.



Abstract Class

Demo

DEMO



Case Class

Definition

```
1  scope var / val varName: vType  
2  scope def methodName(methodParam): methodRetType =  
3  end ClassName
```

ClassName the name of class.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Case Class

Property

- the same with standard class.
- can be use for comparison (by structure and not by reference).



Enumeration

Definition

An enumeration is used to define a type consisting of a set of named values (Scala3 reference). Let's think it is the same with **sealed class**.

```

1 enum EnumName [typParam](prm1, ... , prmN) extends ext :
2   case caseName (param) extend ext
3   scope var / val varName: vType
4   scope def methodName(methodParam): methodRetType =
5 end EnumName

```

EnumName the name of trait.

typParam type parameter(s), separated by comma.

constructorPlace list of expression(s) for constructor.

scope private / protected / public. The **public** is default scope in Scala.

varName the name of variable / value.

methodName the name of method in class.

methodRetType the return's type of method.



Enum

Property

- Automatically instantiate.
- Support variable parameter for constructor.
- Supports type parameter.



Case Study

a Simple arithmetic PL

DEMO

