83-2、kotlin可变长参数的注意点

（1）定义

- 正常来说，主构造函数或普通函数只能一个一个的定义参数，kotlin中可以**通过vararg定义可变长的参数**，可以同时传多个参数，等同于java的...

（2）与java比较

- java中可变长参数只能声明在方法形参的末尾，需要先定义方法前面的形参，剩余的形参都是可变长参数。
- kotlin中由于可以使用具名参数，所以可变长参数可以声明在方法形参的任意位置。

复制代码

```
fun main(args:Array<String>){
    printArray(3,"H","ello","world",arg2 =
"OK")
}

fun printArray(arg1:Int,vararg
array:String,arg2:String){
    println("arg1=$arg1")
    array.forEach(::print)
    println("\narg2=$arg2")
 }

//打印结果
arg1=3
Helloworld
```

```
arg2=OK

```

java

（3）给方法的可变长参数传递的是一个数组时

- 需要使用spread操作符*来标识数组，会将数组自动平铺展开到可变长参数中。
- 或者使用具名方式替换掉操作符*也可以达到数组平铺到可变长参数中的效果。
- 唯一不足的是截止目前操作符，只支持将数组展开，不支持将集合展开。

复制代码

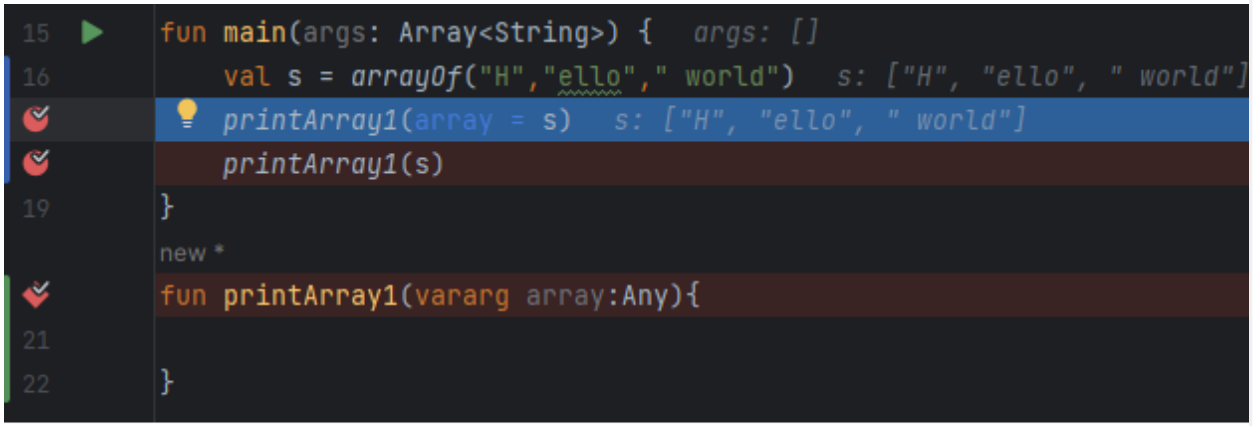
```
val s = arrayOf("H","ello"," world")
  printArray(3,s,arg2 = "OK") ×
  printArray(3,*s,arg2 = "OK") √
  printArray(3,array = s,arg2 = "OK") √

fun printArray(arg1:Int,vararg
array:String,arg2:String){
     println("arg1=$arg1")
   array.forEach(::print)
   println("\narg2=$arg2")
 }
```

java

（4）直观演示

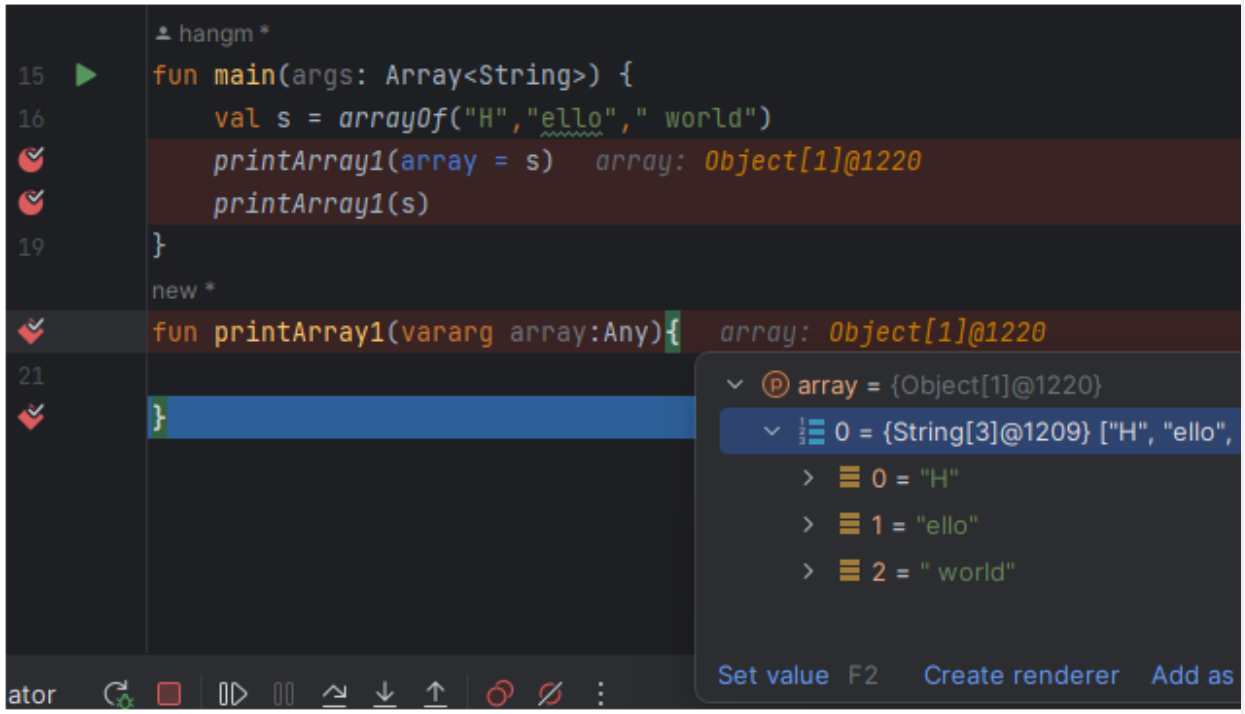
笔记目录

- 使用具名方式传入数组实参，使用可变长形参接收，此时数组元素已经平铺到可变长参数中



- 不使用具名方式，直接传入数组实参，使用可变长形参接收，此时可变长参数就是数组本身


hangm *
15
fun main(args: Array<String>) { args: []
16
val s = arrayOf("H","ello"," world") s: ["H", "ello", " world"]
printArray1(array = s)
printArray1(s) s: ["H", "ello", " world"]
19
}
new *
fun printArray1(vararg array:Any){
21
22
}


hangm *
15
fun main(args: Array<String>) {
16
val s = arrayOf("H","ello"," world")
printArray1(array = s) array: Object[1]@1220
printArray1(s)
19
}
new *
fun printArray1(vararg array:Any){ array: Object[1]@1220
21
}
array = {Object[1]@1220}
0 = {String[3]@1209} ["H", "ello",
0 = "H"
1 = "ello"
2 = " world"
Set value F2
Create renderer
Add as
ator