



Kotlin

Rafael Vieira Coelho
rafaelvc2@gmail.com

<https://kotlinlang.org>

If como Expressão

- Aqui, o valor max recebe a. Caso contrário, recebe b.

```
val max = if (a > b) a else b
```

- Não existe if ternário em Kotlin.

```
(a > b) ? a : b
```

If como Expressão

- No exemplo abaixo, é mostrado na tela o valor 99.

```
fun max(a: Int, b: Int) = if (a > b) a else b  
println(max(99, -42))
```

Verificação de Igualdade

1. Retorna true pois invoca authors.equals(writers)
2. Retorna false pois authors e writers tem referências distintas.

```
val authors = setOf("Shakespeare", "Hemingway", "Twain")
val writers = setOf("Twain", "Shakespeare", "Hemingway")

println(authors == writers)    // 1
println(authors === writers)  // 2
```

When como Switch

- Neste caso, testamos qual o conteúdo do parâmetro color (tipo enum).

```
enum class Color {  
    BLUE, ORANGE, RED  
}  
  
fun getDescription(color: Color): String =  
    when (color) {  
        BLUE -> "cold"  
        ORANGE -> "mild"  
        RED -> "hot"  
    }
```

When como Switch

- Não precisamos de break!

```
switch (color) {  
    case BLUE:  
        System.out.println("cold");  
        break;  
  
    case ORANGE:  
        System.out.println("mild");  
        break;  
  
    default:  
        System.out.println("hot");  
}
```



```
when (color) {  
    BLUE -> println("cold")  
    ORANGE -> println("mild")  
    else -> println("hot")  
}
```



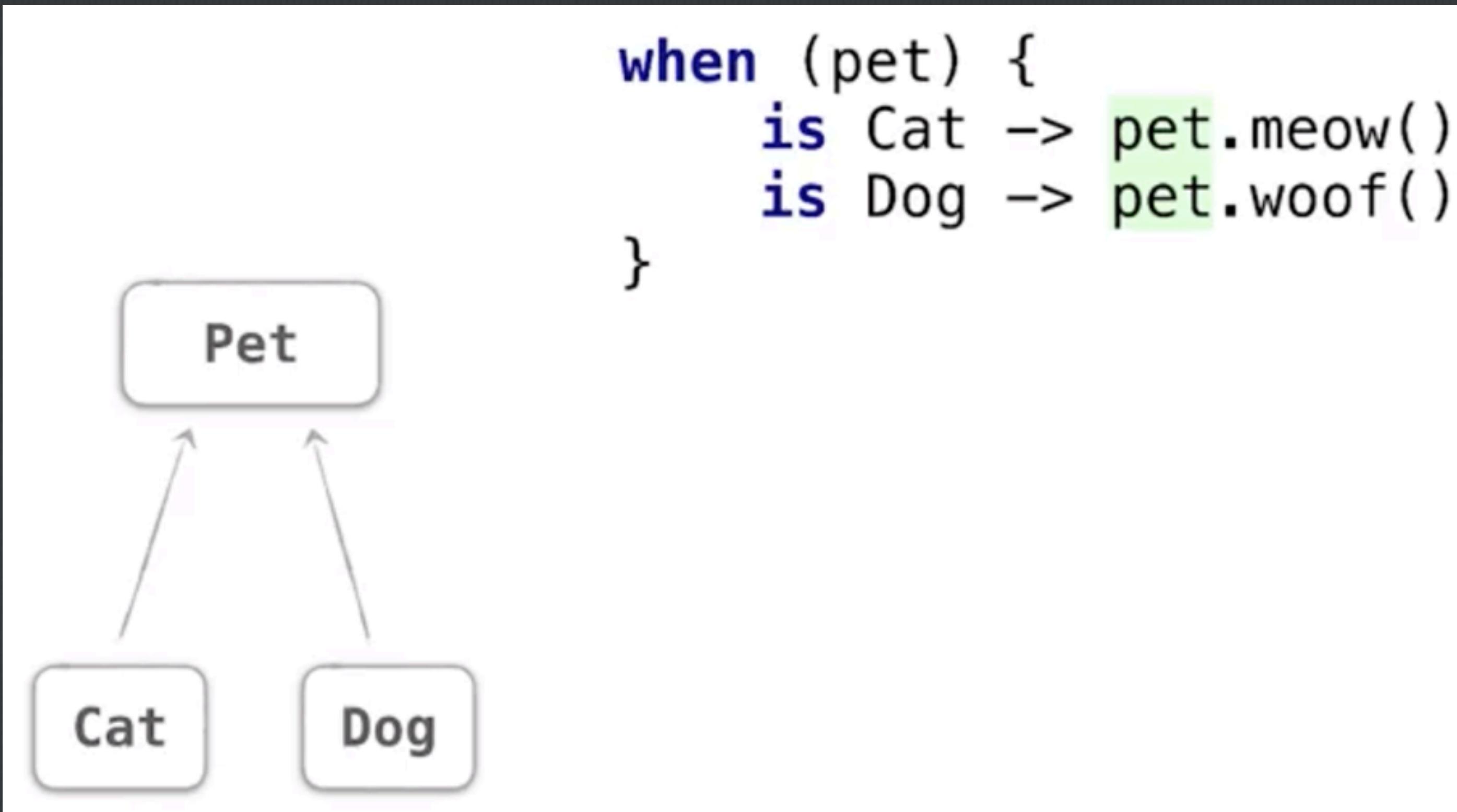
When como Switch

- Podemos testar múltiplos valores com o When.

```
fun respondToInput(input: String) = when (input) {  
    "y", "yes" -> "I'm glad you agree"  
    "n", "no" -> "Sorry to hear that"  
    else -> "I don't understand you"  
}
```

Testando Tipos

- Estamos testando se o objeto pet é da classe Cat ou Dog



Testando Tipos

- Em Java, usariamos o `instanceOf`

```
if (pet instanceof Cat) {  
    ((Cat) pet).meow();  
}  
else if (pet instanceof Dog) {  
    Dog dog = (Dog) pet;  
    dog.woof();  
}
```

The diagram illustrates the equivalence between Java's `instanceOf` operator and Kotlin's `when` expression. On the left, a Java code snippet uses `instanceOf` to check if a `pet` is a `Cat`, calling `meow()` on it. It then checks if `pet` is a `Dog`, casting it to `Dog` and calling `woof()`. On the right, a Kotlin code snippet uses a `when` expression to achieve the same result. The `when` expression has `pet` as its subject. It branches into two cases: one for `Cat` leading to `pet.meow()`, and another for `Dog` leading to `pet.woof()`. A curved arrow points from the `instanceOf` check in the Java code to the `when` expression in the Kotlin code, indicating their functional equivalence.

```
when (pet) {  
    is Cat -> pet.meow()  
    is Dog -> pet.woof()  
}
```

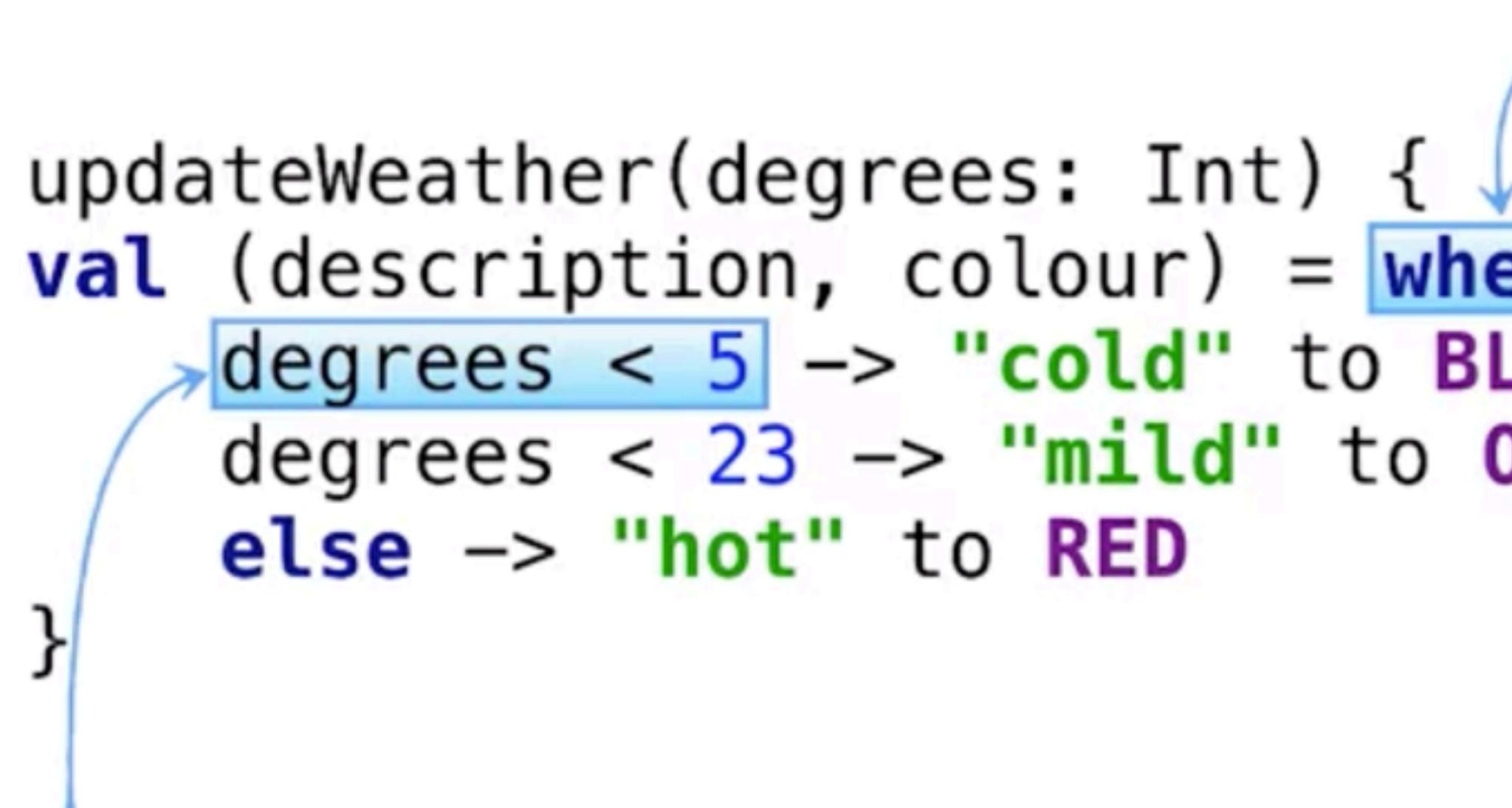
Podemos Declarar uma Variável no When

```
when (val pet = getMyFavouritePet()) {  
    is Cat -> pet.meow()  
    is Dog -> pet.woof()  
}
```

Omitindo Argumentos no When

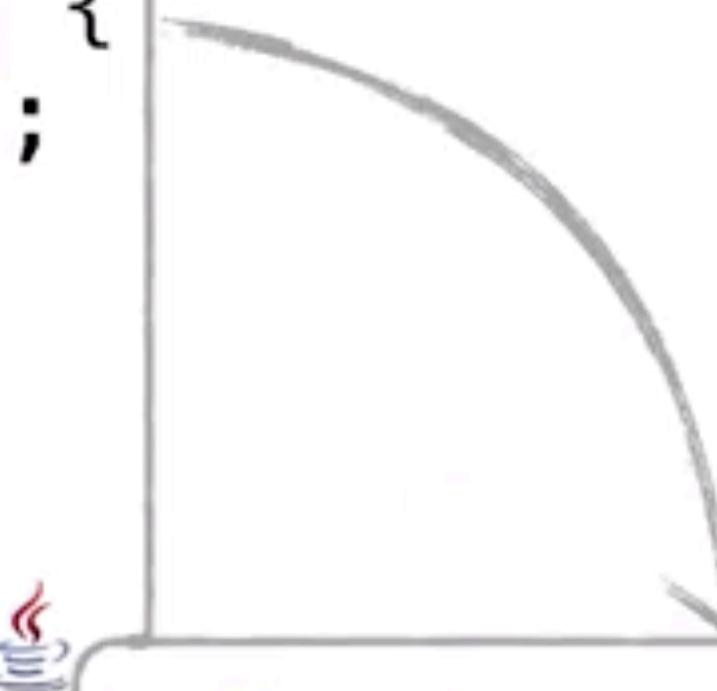
```
fun updateWeather(degrees: Int) {  
    val (description, colour) = when {  
        degrees < 5 -> "cold" to BLUE  
        degrees < 23 -> "mild" to ORANGE  
        else -> "hot" to RED  
    }  
}  
  
any Boolean  
expression
```

no argument



Java x Kotlin

```
String description;  
Colour colour;  
if (degrees < 5) {  
    description = "cold";  
    colour = BLUE;  
} else if (degrees < 23) {  
    description = "mild";  
    colour = ORANGE;  
} else {  
    description = "hot";  
    colour = RED;  
}
```



```
val (description, colour) = when {  
    degrees < 5 -> "cold" to BLUE  
    degrees < 23 -> "mild" to ORANGE  
    else -> "hot" to RED  
}
```



Exemplo de When

- Note que cada teste é realizado na ordem apresentada, até uma delas ser verificada (1 a 6)

```
fun main() {  
    cases("Hello")  
    cases(1)  
    cases(0L)  
    cases(MyClass())  
    cases("hello")  
}  
  
fun cases(obj: Any) {  
    when (obj) {  
        1 -> println("One")  
        "Hello" -> println("Greeting")  
        is Long -> println("Long")  
        !is String -> println("Not a string")  
        else -> println("Unknown")  
    }  
}
```

Exemplo de When

- Aqui usamos when como uma expressão.

```
fun main() {  
    println(whenAssign("Hello"))  
    println(whenAssign(3.4))  
    println(whenAssign(1))  
    println(whenAssign(MyClass()))  
}  
  
fun whenAssign(obj: Any): Any {  
    val result = when (obj) {  
        1 -> "one" // 1  
        "Hello" -> 1 // 2  
        is Long -> false // 3  
        else -> 42 // 4  
    }  
    return result // 5  
}
```