Lazy or late initialization of properties

Lazy property

Lazy property

```
val lazyValue: String by lazy {
    println("computed!")
    "Hello"
}


fun main(args: Array<String>) {
    println(lazyValue)
    println(lazyValue)
}
```

computed!
Hello
Hello

How many times “computed!” will be printed?

```
val lazyValue: String by lazy {
    println("computed!")
    "Hello"
}

fun main(args: Array<String>) {
    // no lazyValue usage
}
```

1. 0
2. 1

How many times “computed!” will be printed?

```
val lazyValue: String by lazy {
    println("computed!")
    "Hello"
}

fun main(args: Array<String>) {
    // no lazyValue usage
}
```

1. 0
2. 1

lateinit

Late initialization

```
class KotlinActivity: Activity() {
    var myData: MyData? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        myData = intent.getParcelableExtra("MY_DATA")
    }

    ... myData?.foo ...
}
```

Late initialization

```
class KotlinActivity: Activity() {
    lateinit var myData: MyData

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        myData = intent.getParcelableExtra("MY_DATA")
    }

    ... myData.foo ...
}
```

If the property wasn't initialized…

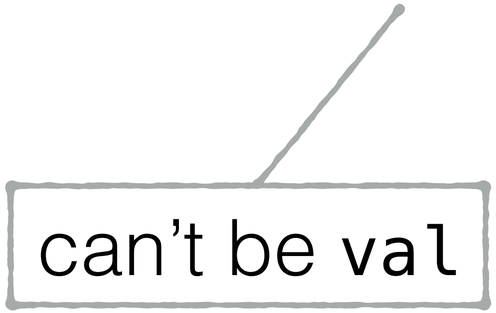
```
... myData.foo ...
```

…still a runtime exception is thrown,
but with the detailed message:

`kotlin.UninitializedPropertyAccessException:`
`lateinit` property `myData` has not been initialized

`lateinit` constraints

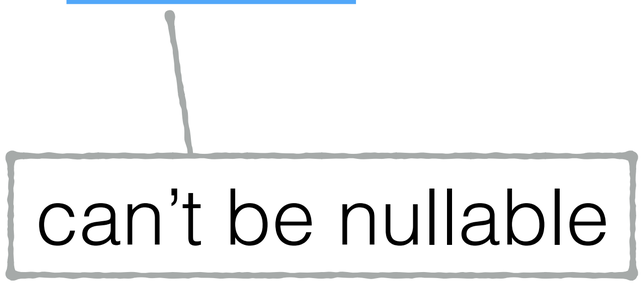
```
lateinit var myData: MyData
```



`lateinit` variable can’t be `final` under the hood (on JVM level), therefore it can be modified from the Java code

`lateinit` constraints

```
lateinit var myData: MyData
```



`lateinit` constraints

```
lateinit var myData: MyData
```

can’t be primitive type

only reference types might be initialized with `null` under the hood

Checking whether `lateinit var` was initialized

```
class MyClass {
    lateinit var lateinitVar: String

    fun initializationLogic() {
        println(this::lateinitVar.isInitialized)    // false
        lateinitVar = "value"
        println(this::lateinitVar.isInitialized)    // true
    }
}
```

Checking whether `lateinit var` was initialized

```
class MyClass {
    lateinit var lateinitVar: String

    fun initializationLogic() {
        println(this::lateinitVar.isInitialized)    // false
        lateinitVar = "value"
        println(this::lateinitVar.isInitialized)    // true
    }
}
```

property reference