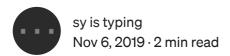
Kotlin Cheatsheet: Scope Functions (let, run, apply, also, with)



		returns	
		original object (Foo)	any object (eg, Bar)
original object (Foo) exposed as	this	<pre>apply (this> foo) val foo = Foo().apply { this.field1 = 1 }</pre>	<pre>run (this> bar) val bar = Foo().run { this.field1 = 1 this.toBar() }</pre>
	it	<pre>also (it> foo) val foo = Foo().also { doSomethingTo(it) }</pre>	<pre>let (it> bar) val bar = getFoo()?.let { it.toBar() }</pre>

Note: this can be omitted.

There are recommended use-cases for each scope function but at the end of the day, choose the function that returns you the correct output and fits your code convention.

<u>apply</u>

Useful for:

• configuring an object:

```
val foo = Foo().apply {
    this.field1 = 1
}
```

run

Useful for:

• configuring an object and computing the result:

```
val bar = Foo().run {
    this.field1 = 1
    this.toBar()
}
```

• grouping several statements into an expression: run has a non-extension form too:

```
val bar = run {
   val foo = Foo()
   foo.field1 = 1
   foo.toBar()
}
```

<u>also</u>

Useful for:

• doing additional effects involving the object:

```
val foo = Foo().also {
     doSomethingTo(it)
}
```

<u>let</u>

Useful for

• executing a lambda on non-null objects:

```
val bar = getFoo()?.let {
    it.toBar()
}
```

• introducing an expression as a variable in local scope:

```
(...complicated expression...).let {
    doSomethingWith(it)
}
```

with

Useful for:

• grouping function calls on an object. Similar to run but not an extension:

```
val bar = with(Foo()) {
    this.field1 = 1
    this.toBar()
}
```

Other stories you may like

- Write fluent code in Kotlin
- Software Estimates are a Two-Sided Relationship
- <u>Use Java Nullability Annotations to facilitate conversion to Kotlin</u>

Hi, if you enjoyed this post, I thought that you might also enjoy these <u>t-shirts with code-inspired designs</u>.

Java Kotlin Programming Android App Development Software Engineering

About Help Legal

Get the Medium app



