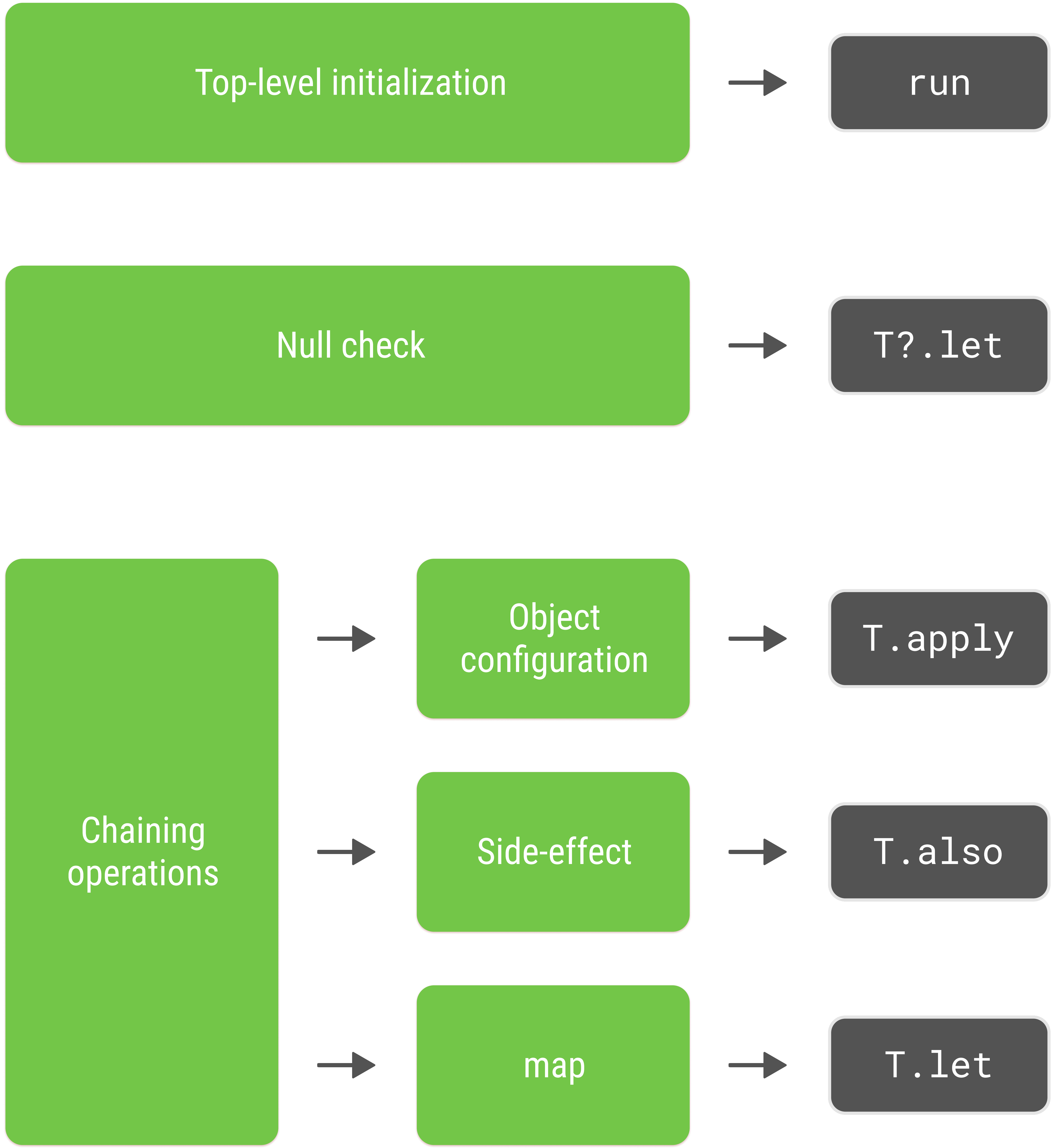
	receiver (this)	argument (it)	returns
`T.also { }`	unchanged	T	T
`T.let { }`	unchanged	T	{ body result }
`T.apply { }`	T	unchanged	T
`T.run { }, run { }`	T	unchanged	{ body result }
`with(T) { }`	T	unchanged	{ body result }



Top-level initialization → run

Null check → T?.let

Chaining operations → Object configuration → T.apply

Chaining operations → Side-effect → T.also

Chaining operations → map → T.let

```
val myObj = run {
    val generator = DateStringGenerator(2008, 9, 23)
    generator.locale = "US"
    generator.localizedDate
}
```

```
class MyClass {

    fun printExceptionMessage(exception: Exception?) {
        exception?.let {
            println(exception.message)
        }
    }
```

```
    val lastWeeksDate: String = Calendar.getInstance().apply {
        add(Calendar.DAY_OF_YEAR, -7)
    }
```

```
        .also {
            println("Created calendar of type: ${it.calendarType}")
        }
```

```
        .let { calendar ->
            val format = SimpleDateFormat("dd/M/yyyy hh:mm:ss", Locale.US)
            format.format(calendar.time)
        }
}
```