# Funções de escopo

Qual e quando utilizar?

# let - `inline fun <T, R> T.let(block: (T) -> R): R`

**Argumento**

Opera com o resultado de uma ou mais funções e com valores nulos

Cria variável com um escopo mais limitado com alguma modificação

# with - `inline fun <T, R> with(receiver: T, block: T.() -> R): R`

**Receptor**

Não é uma função de extensão

Chama membros do objeto de contexto diretamente

Também é usado para computar algo com o receptor e retornar o valor

Em código lemos “com esse objeto, faça”

# run - `inline fun <T, R> T.run(block: T.() -> R): R`

**Receptor**

Similar a proposta do **let**, mas o uso é parecido com o **with**

Utiliza na inicialização de um objeto

Retorna a computação do objeto

# run sem extensão - `inline fun <R> run(block: () -> R): R`

**Sem objeto de contexto**

Similar a execução do **with,** mas não recebe nenhum objeto

Executa um bloco de código e retorna a sua computação.

# apply - `inline fun <T> T.apply(block: T.() -> Unit): T`

**Receptor**

Recebe e retorna o objeto de contexto

Configura o objeto de contexto sem fazer computações

Em código lemos “Aplique as seguintes atribuições ao objeto”

# also - `inline fun <T> T.also(block: (T) -> Unit): T`

**Argumento**

Igual ao **apply** em retorno

Realiza ações a mais que não modificam o objeto

**A remoção do also não deve afetar o programa.**

Em código lemos “também faça isso”

# Diferenças das funções em tabela

| FUNÇÕES | CONTEXTO | RETORNO | EXTENSÃO |
|---|---|---|---|
| **let** | it | lambda | sim |
| **run** | this | lambda | sim |
| **run** | - | lambda | não: sem objeto |
| **with** | this | lambda | não: objeto via arg |
| **apply** | this | objeto de contexto | sim |
| **also** | this | objeto de contexto | sim |

# Resumo

Executar um lambda em objetos não nulos: **let()**

Introduzindo uma expressão como uma variável no escopo local: **let()**

Configuração de objeto: **apply()**

Configuração do objeto e computação do resultado: **run()**

Execução de instrução quando a expressão é necessária: **run() - sem extensão**

Execuções adicionais sem afetar o programa: **also()**

Agrupamento de chamadas de função em um objeto: **with()**

# takeIf e takeUnless

Utilizados para verificar o estado do objeto.

Retorna o objeto ou null dependendo do predicado:

- **Verdadeiro: takeIf()**
- **Falso: takeUnless()**