**Nome:** Danieli Maria Lefchak

**RESUMO**

**Android *View binding vs findViewById***

O *view binding* faz com que a quantidade de código que será escrito para inicializar todas as *views* seja menor do que quando feito com *findviewbyid,* o *findViewById* não traz tanta segurança quando se trata dos tipos das *views* e de tipos nulos, já o *view binding* vem com a promessa de tratar isso de uma forma melhor. Na maioria dos casos o *view binding* substitui o *findviewbyid*.

Para que possa ser realizada a instancia da classe *binding* é necessário fazer os passos em seguida no método *onCreate*: chamar o método *inflate*() na classe de ligação gerada, após isso chamar o *getRoot*() para obter uma referência raiz e por fim para exibir o raiz chame o método *setContextView*().

**Kotlin**

***Lateinit***

Faz com que seja possível inicializar a variável depois de sua declaração, por exemplo, já foi codificado uma boa parte do projeto e agora você irá precisar dessa variável que ainda não foi inicializada, você pode chama-la e inicializa-la quando ela for necessária.

O *lateinit* somente pode ser utilizado em variáveis com referência mutável, ou seja, as variáveis de tipo var, e o tipo dessas variáveis devem ser diferentes de nulo.

Um exemplo de implementação do *lateinit* seria atribuir objetos *View* na função *onViewCreated*:

class LoginFragment : Fragment() {  
  
    private lateinit var usernameEditText: EditText  
    private lateinit var passwordEditText: EditText  
    private lateinit var loginButton: Button  
    private lateinit var statusTextView: TextView  
  
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
        super.onViewCreated(view, savedInstanceState)  
  
        usernameEditText = view.findViewById(R.id.username\_edit\_text)  
        passwordEditText = view.findViewById(R.id.password\_edit\_text)  
        loginButton = view.findViewById(R.id.login\_button)  
        statusTextView = view.findViewById(R.id.status\_text\_view)  
    }  
    // rest of the code  
}

***Lazy***

É uma função lambda que faz com que a inicialização da propriedade seja feita na primeira vez que em que ela será utilizada, e nas próximas vezes de uso, o valor atribuído será devolvido imediatamente. Lazy só pode ser utilizado em variáveis imutáveis, ou seja, variáveis de tipo val.

Um exemplo de implementação do *lazy* é na inicialização de um *RecyclerView*:

class NoteListActivity : AppCompatActivity() {  
  
    private val adapter: NoteListAdapter by *lazy* {  
            NoteListAdapter(notes(), this)  
    }

    private val recyclerView: RecyclerView by *lazy* {  
            note\_list\_recyclerview  
    }  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
            super.onCreate(savedInstanceState)  
            setContentView(R.layout.*activity\_note\_list*)  
            configureRecyclerView()  
    }  
  
    private fun configureRecyclerView() {  
            // initialize the recyclerView and adapter here  
            recyclerView.*adapter* = adapter  
            val layoutManager = StaggeredGridLayoutManager(  
            2, StaggeredGridLayoutManager.*VERTICAL*)  
            recyclerView.*layoutManager* = layoutManager  
    }  
    // rest of the code  
}

**REFERENCIAS**

* <https://medium.com/android-dev-br/quando-utilizar-inicializa%C3%A7%C3%A3o-lazy-do-kotlin-45377c8b68ad>
* <https://medium.com/@joelamalio/kotlin-lateinit-598ffdfb8f5a>
* <https://bignerdranch.com/blog/kotlin-when-to-use-lazy-or-lateinit/>
* <https://blog.logrocket.com/initializing-lazy-lateinit-variables-kotlin/>
* <https://amitshekhar.me/blog/lateinit-vs-lazy-in-kotlin>
* <https://agrawalsuneet.github.io/blogs/lateinit-vs-lazy-property-in-kotlin/>
* <https://www.educative.io/answers/when-should-you-use-lateinit-over-lazy-initialization-in-kotlin>
* <https://developer.android.com/topic/libraries/view-binding?hl=pt-br>
* <https://proandroiddev.com/new-in-android-viewbindings-the-difference-from-databinding-library-bef5945baf5e>
* <https://www.techyourchance.com/viewbinding-vs-findviewbyid-in-android/>
* <https://developer.android.com/kotlin/common-patterns?hl=pt-br>