Trabalhando com Fragmentos em Kotlin Parte 1 (Herança)

As seções a seguir usamos exemplos de “Fragment” para destacar alguns dos mais incríveis recursos do Kotlin.

Herança

Você pode declarar uma classe no Kotlin com a palavra-chave “Class”. No exemplo abaixo,'**`LoginFragment`**' é uma subclasse de '**`Fragment`**'. Você pode indicar a herança usando o operador ':'entre a subclasse e o pai:

```
class LoginFragment : Fragment()
```

Nessa declaração de classe, '**`LoginFragment`**' é o responsável por chamar o construtor da superclasse, '**`Fragment`**'.

Dentro de '**`LoginFragment`**', você pode modificar vários callbacks de ciclo de vida para responder a mudanças de estado no '**`Fragment`**'. Para substituir uma função, use a palavra-chave '**`override`**', conforme mostrado no exemplo a seguir:

```
override fun onCreateView(

     inflater: LayoutInflater,
     container: ViewGroup?,
     savedInstanceState: Bundle?
): View? {
     return inflater.inflate(R.layout.login_fragment, container,
false)
}
```

Para fazer referência a uma função na classe pai, use a palavra-chave **`super`**, conforme mostrado no exemplo abaixo:

```
override fun onViewCreated(view: View, savedInstanceState:
Bundle?) {
     super.onViewCreated(view, savedInstanceState)
}
```