

zoom Workplace

Esta reunião está sendo gravada

You are viewing Steffeson Lira's screen REC View Options

## Agenda

7 Ciclo de vida da Activity e do fragment

- Ciclo de vida da Activity
- Registro de Log
- Ciclo de vida do Fragment
- Componentes que reconhecem o ciclo de vida
- Tarefas(Tasks) and pilha de volta (back stack)
- Resumo

venturus

Participants Chat Reactions Share Screen AI Companion More

Unmute Stop Video

Leave

Esta reunião está sendo gravada

You are viewing Steffeson Lira's screen REC View Options

## Por que isso é importante?

- Preserve os dados e o estado dos usuários se:
  - o O usuário sai temporariamente do app e depois retorna
  - o O usuário é interrompido (por exemplo, por uma chamada telefônica)
  - o O usuário gira o dispositivo
- Evite vazamentos de memória e falhas no aplicativo.

venturus

Steffeson Lira André Luiz Barbosa RH Zoom Daniela Pinheiro Anderson fernandes

You are viewing Steffeson Lira's screen REC View Options

## Ciclo de vida da activity de forma simplificada

```

graph TD
    A[Activity iniciada] --> B[onCreate()]
    B --> C[App em execução]
    C --> D[Activity finalizada]
  
```

No onCreate que começa a transformação da tela  
Parte completa do ciclo de vida

venturus

Programa de Capacitação | Trilha Android Nativo

https://app.zoom.us/wc/8954344...

zoom Workplace

You are viewing Steffeson Lira's screen REC View Options

## Ciclo de vida da Activity

```
graph TD; A[Activity iniciada] -- onCreate() --> B[onStart()]; B --> C[onResume()]; C --> D[Activity em execução]; D -- onPause() --> E[onStop()]; E -- onDestory() --> F[Activity finalizada]
```

Activity iniciada → onCreate() → onRestart() → onStart() → onResume() → Activity em execução → onPause() → onStop() → onDestory() → Activity finalizada

Nezi Pimentel To Meeting Group... ah que bom

venturus

Unmute Stop Video Participants Chat Reactions Share Screen AI Companion More Leave

18:41 02/10/2025

Status

zoom Workplace

You are viewing Steffeson Lira's screen REC View Options

## Status das Activities

```
graph TD; A[CREATED] --> B[STARTED]; B --> C[RESUMED]; C --> D[Activity is running]; D -- PAUSED --> E[STOPPED]; E -- DESTROYED --> F[DESTROYED]
```

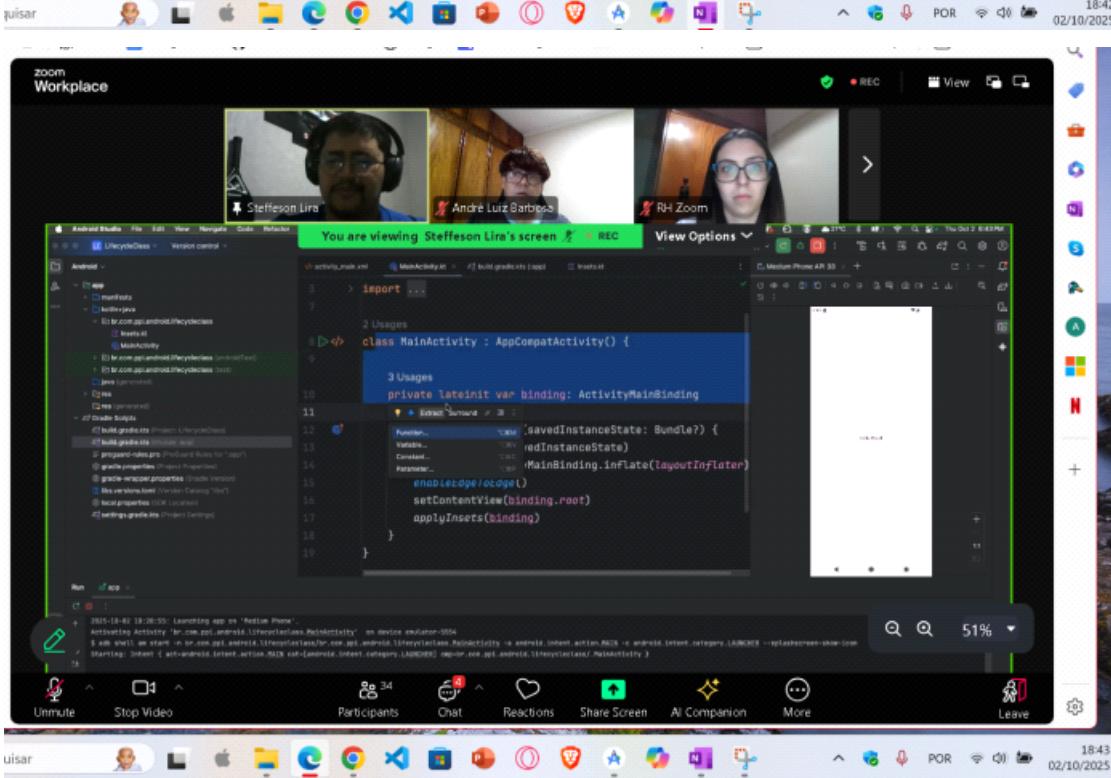
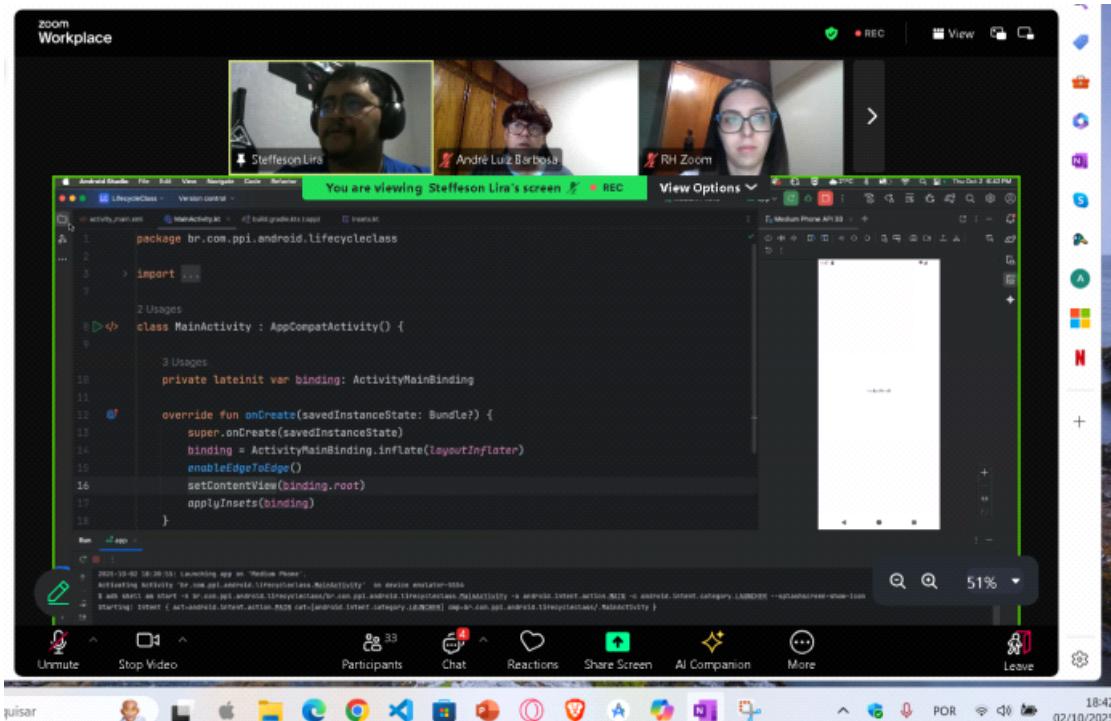
CREATED → STARTED → RESUMED → Activity is running → PAUSED → STOPPED → DESTROYED

venturus

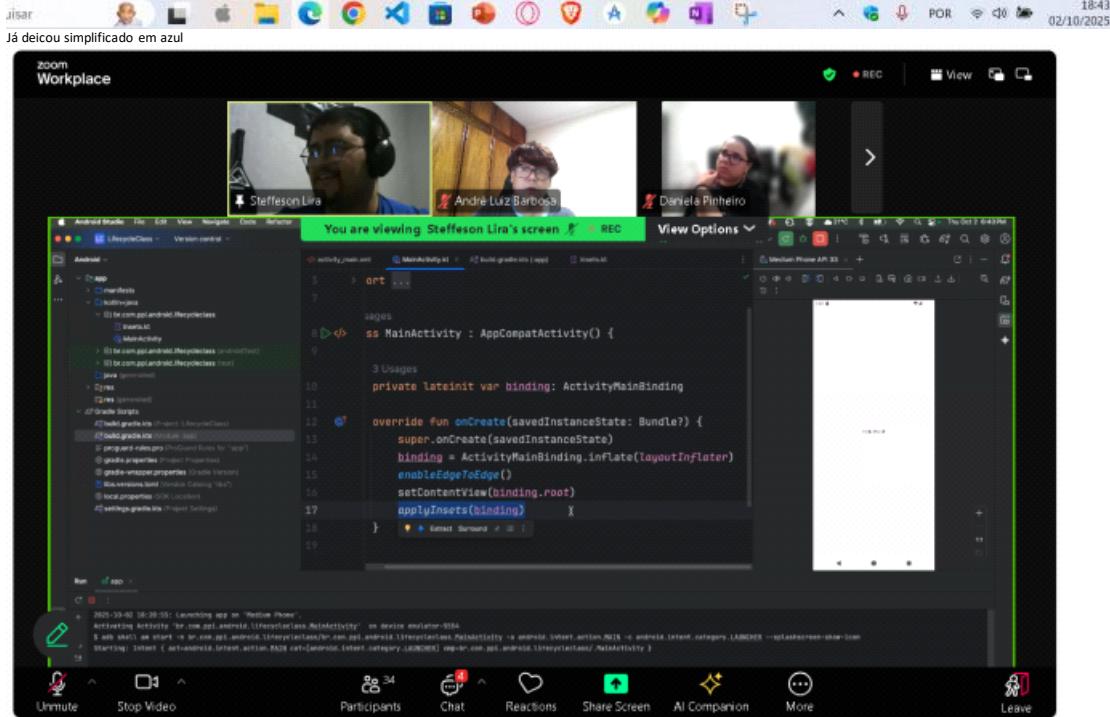
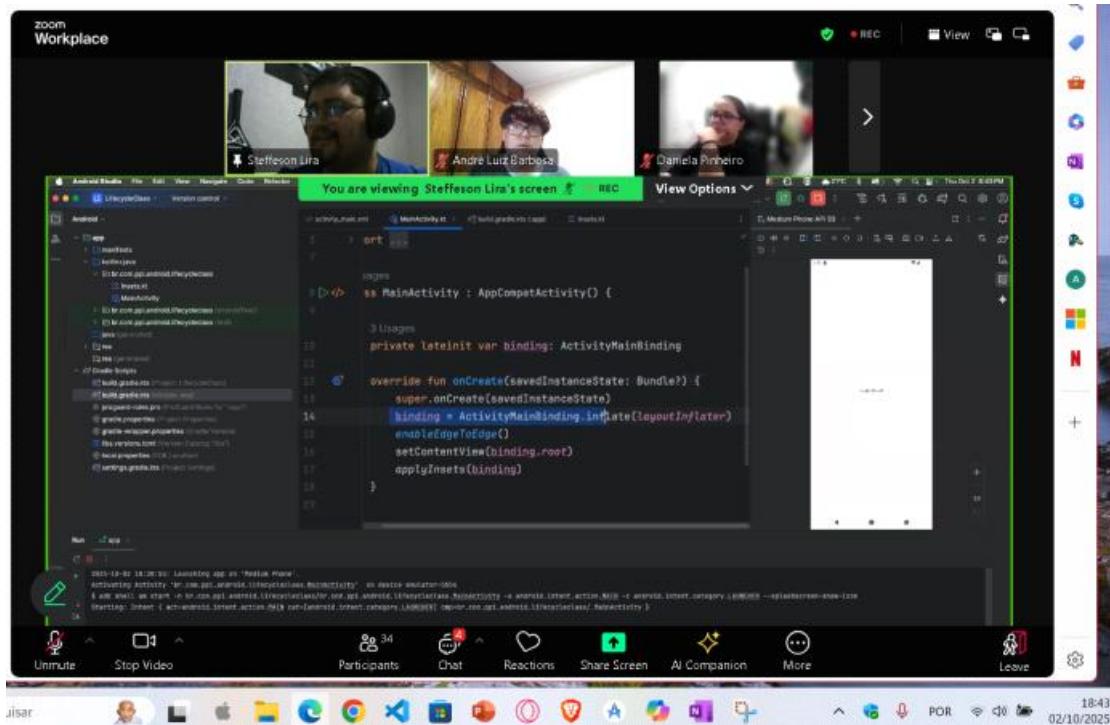
Unmute Stop Video Participants Chat Reactions Share Screen AI Companion More Leave

18:42 02/10/2025

Vendo no código

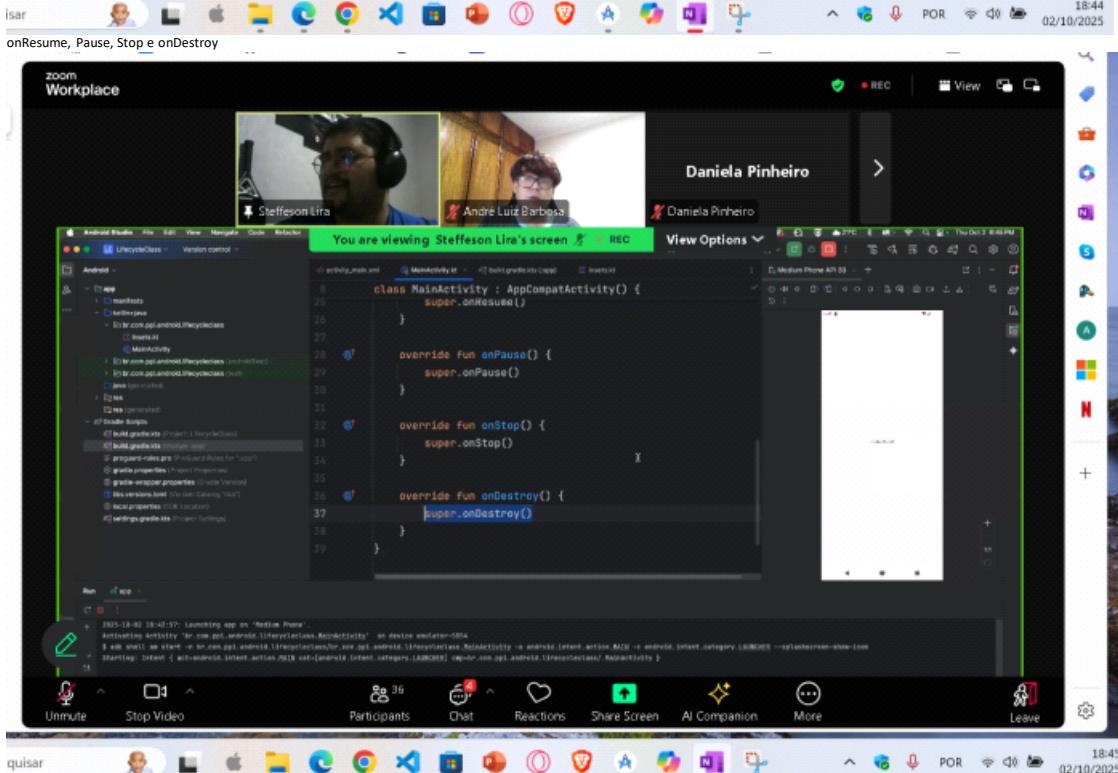
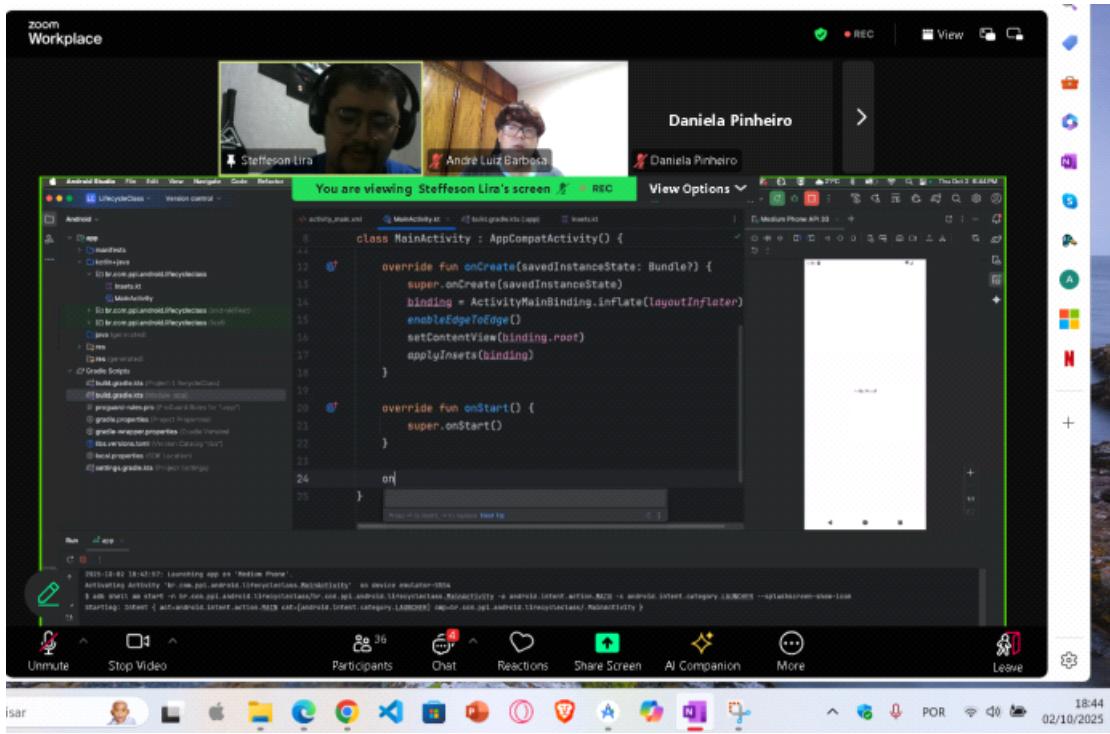


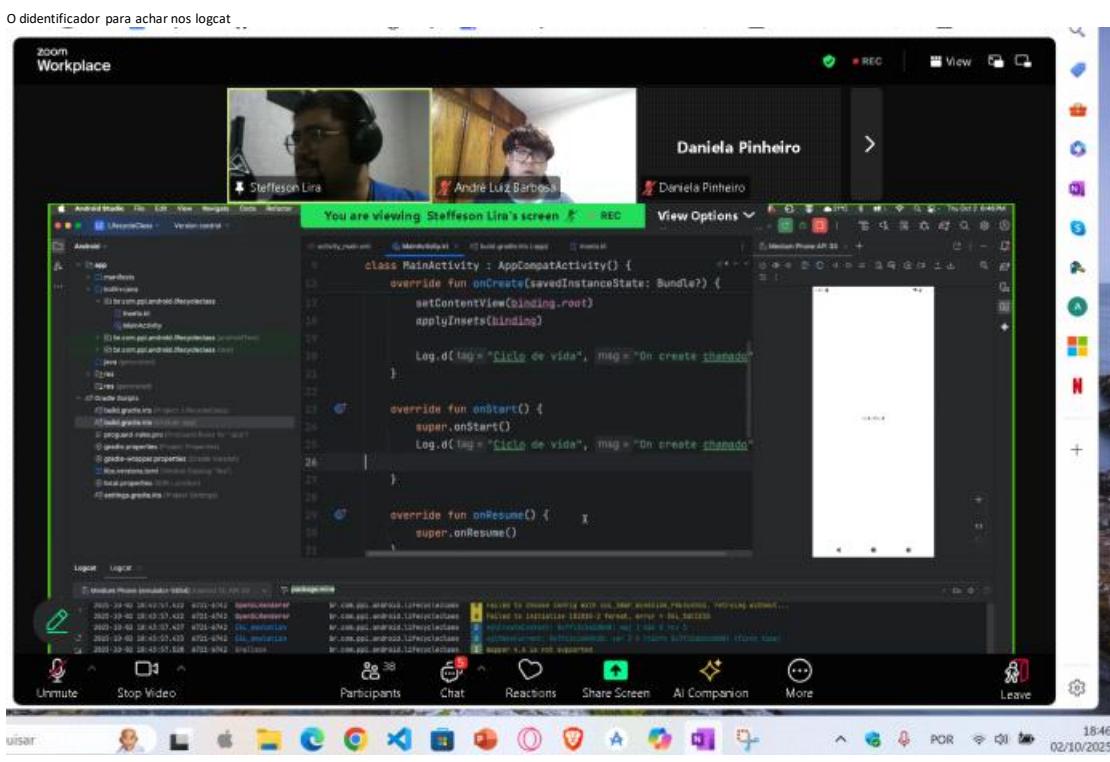
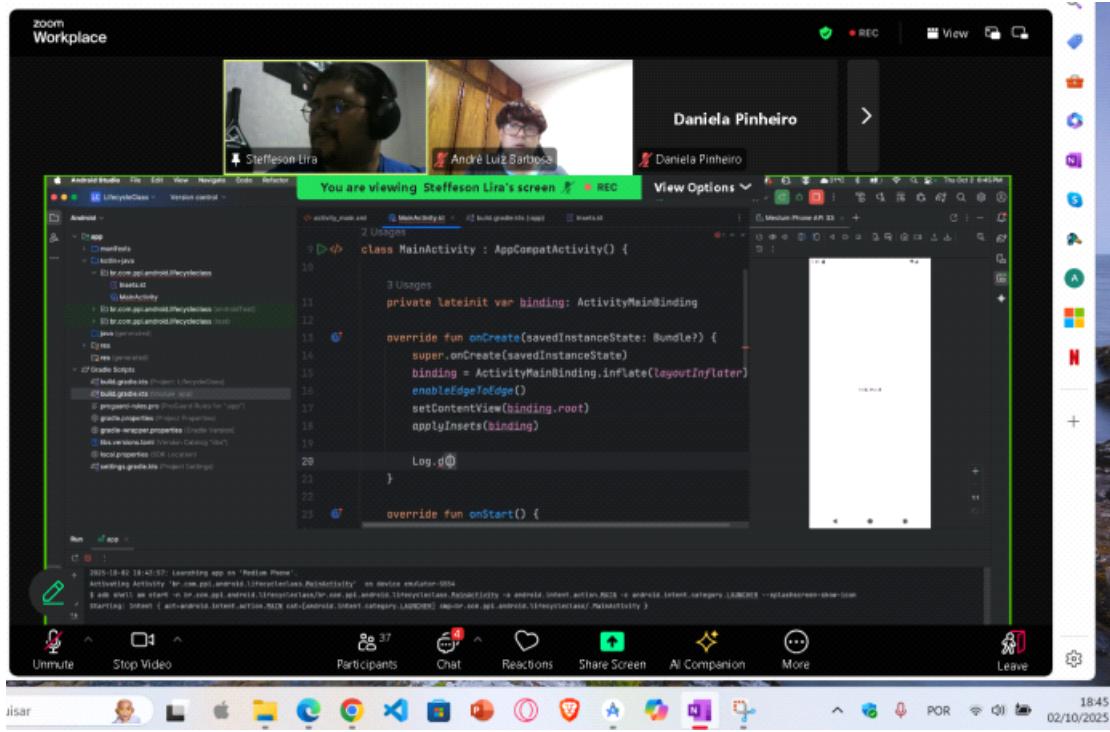
Criou o oncreate e infla os componentes



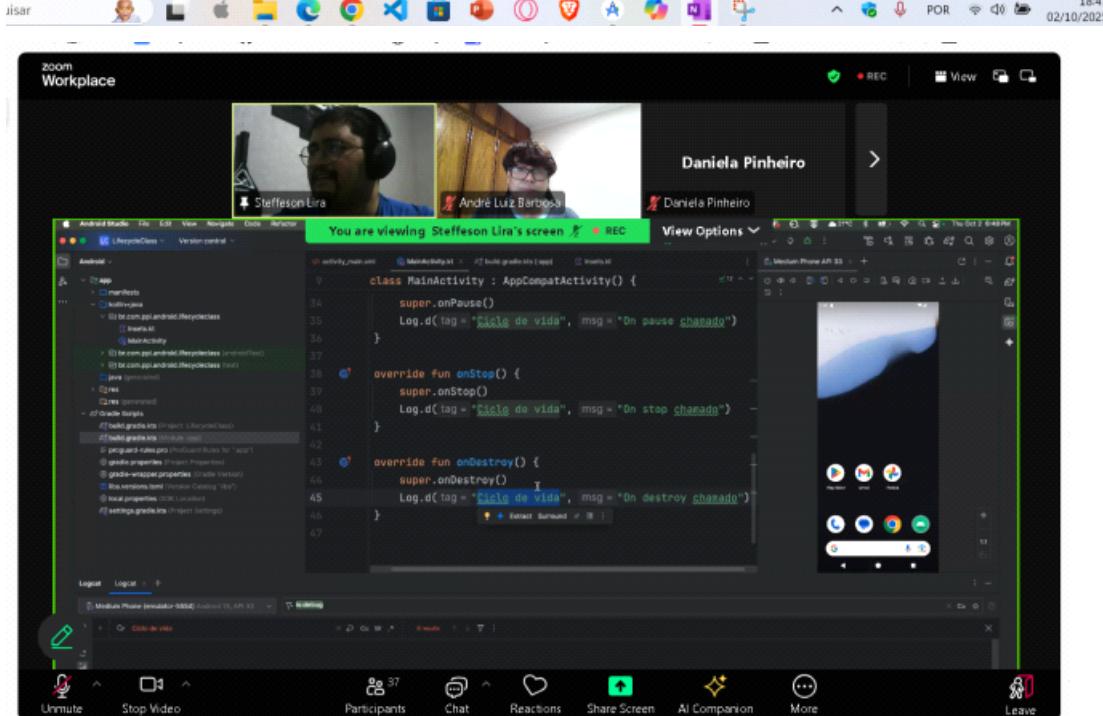
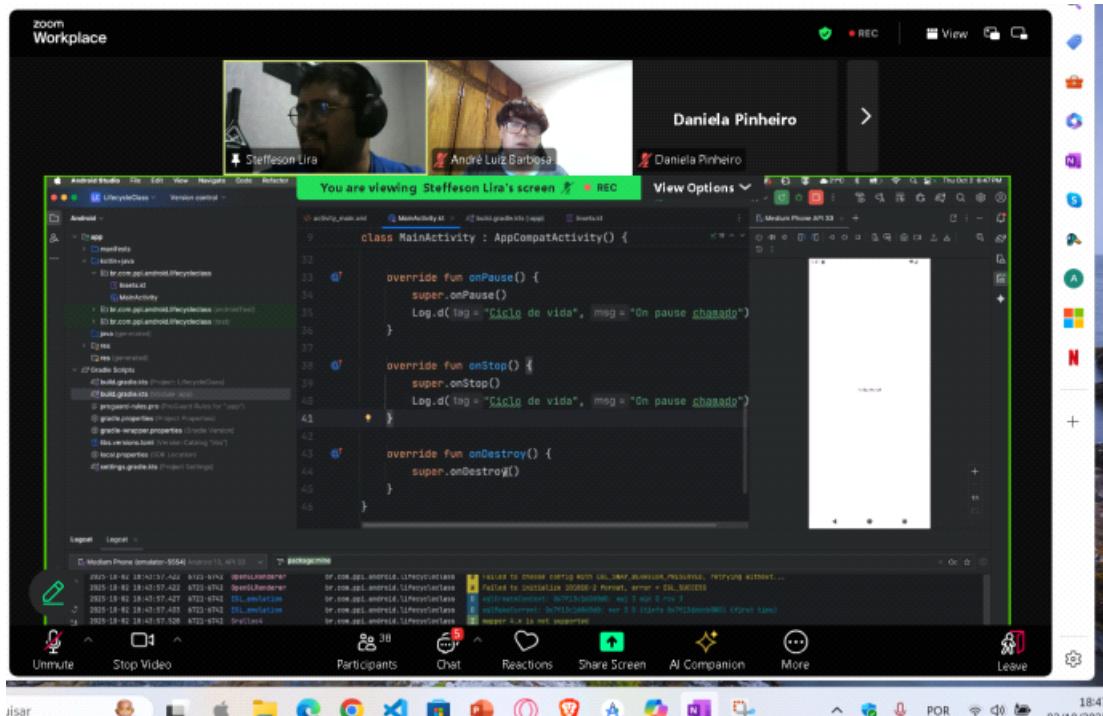
Clica e executa o app.  
Agora os outros métodos do ciclo de vida

A segunda opção era o onStart





Vai criar log para cada um dos onXXXX



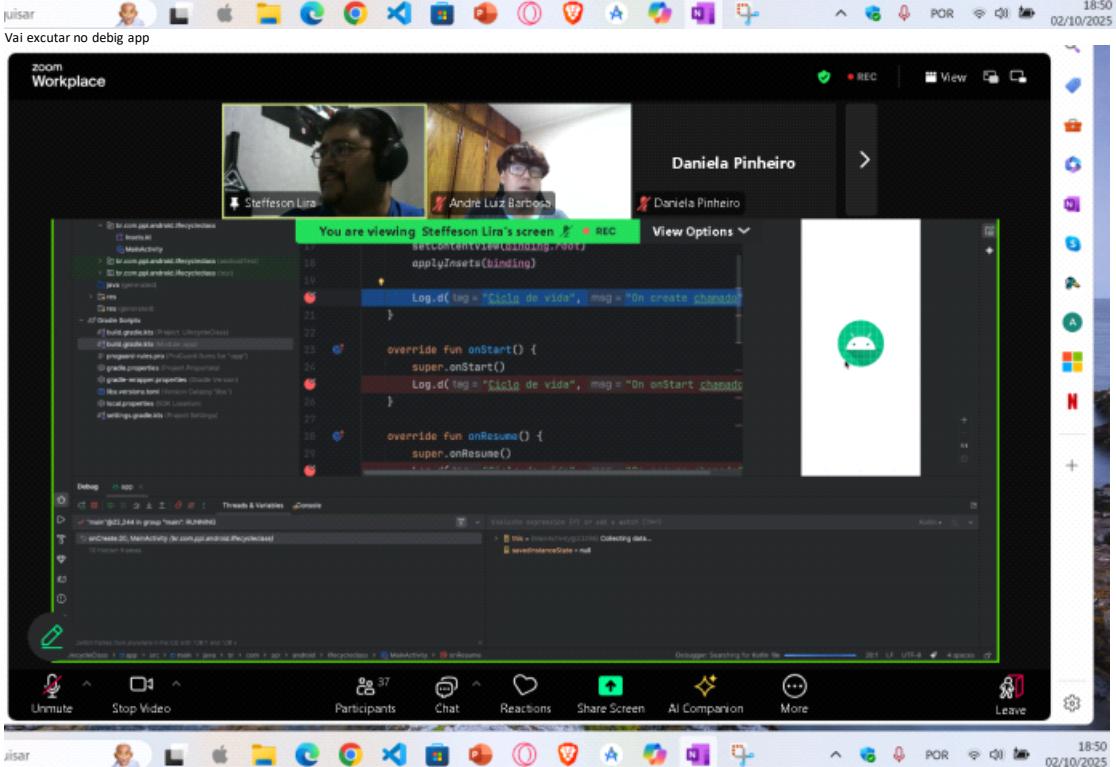
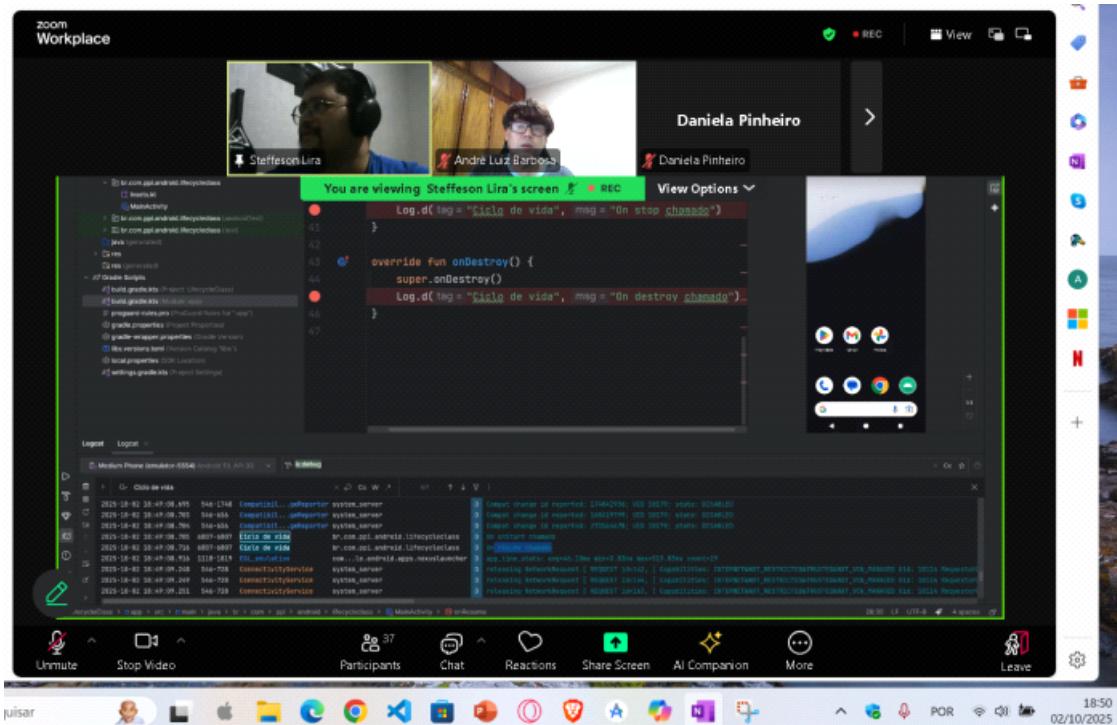
A screenshot of a Zoom video conference interface. At the top, there are four participant video feeds: Steffeson Lira, André Luiz Barbosa, Daniela Pinheiro, and RH Zoom. A green bar at the top center indicates "You are viewing Steffeson Lira's screen" and shows a "REC" button. Below the video feeds, the main content area displays a Java code editor with a file named "Ciclo de vida". The code contains two overridden methods: `onStop()` and `onDestroy()`, which log messages to the console. Below the code editor is a terminal window showing command-line logs. The terminal output includes several lines of text starting with "2023-10-02 20:48:33.921" and "2023-10-02 20:48:33.970". The logs mention "onStop()", "onDestroy()", and "onDestroy()". The terminal also shows other system logs related to network connections and system processes. The Zoom interface includes standard controls like Unmute, Stop Video, Participants, Chat, Reactions, Share Screen, AI Companion, More, and Leave.

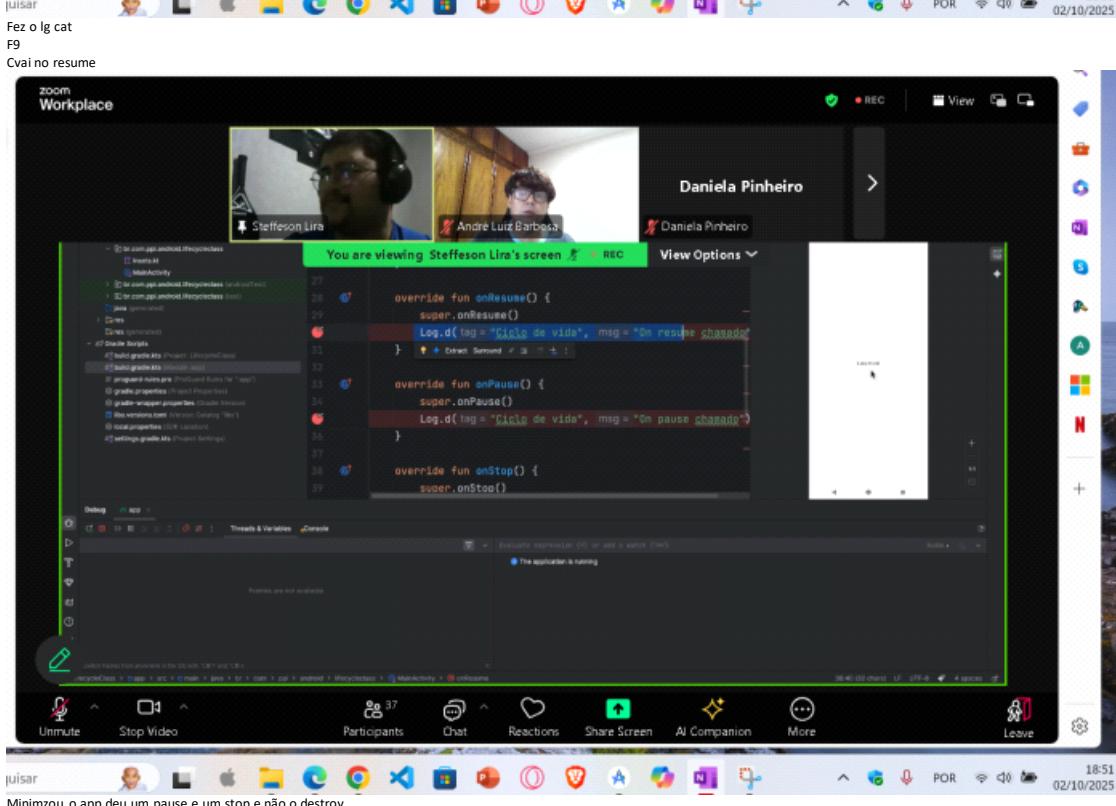
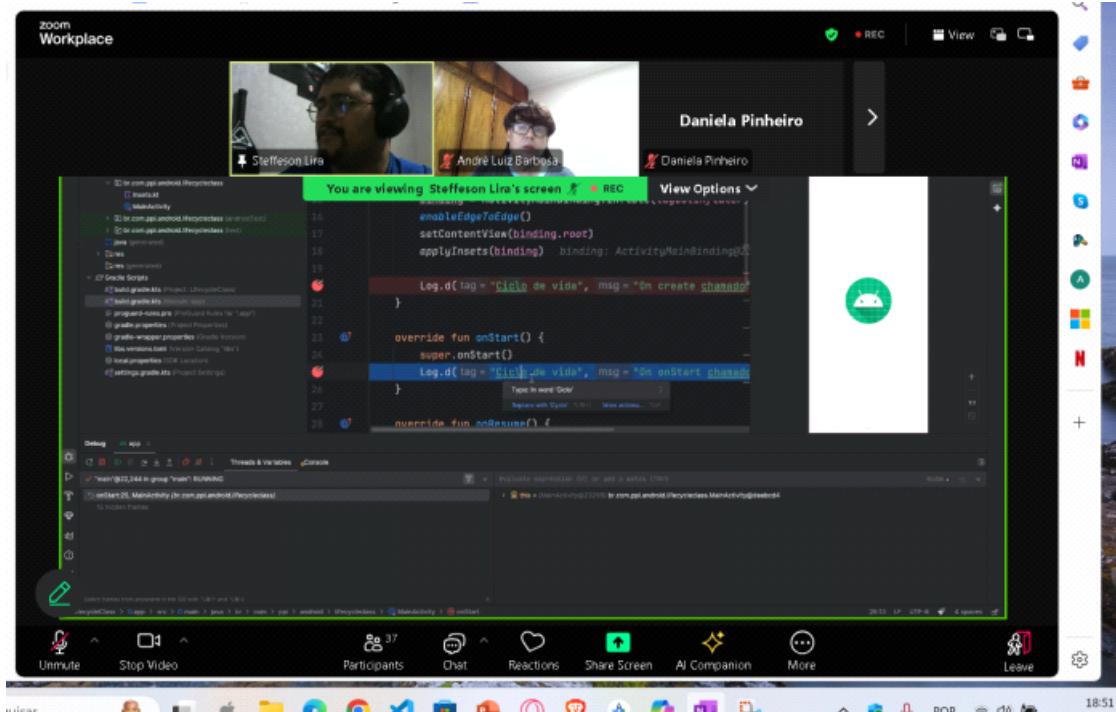
A second screenshot of a Zoom video conference, identical in layout to the first. It shows the same four participants: Steffeson Lira, André Luiz Barbosa, Daniela Pinheiro, and RH Zoom. The "You are viewing Steffeson Lira's screen" bar is present. The main content area shows the same Java code editor with the "Ciclo de vida" file and its `onStop()` and `onDestroy()` methods. The terminal window below shows the same command-line logs as the previous screenshot, including multiple entries for `onStop()` and `onDestroy()`. The Zoom interface includes Unmute, Stop Video, Participants, Chat, Reactions, Share Screen, AI Companion, More, and Leave buttons.

Vai ensinar a usar o debug

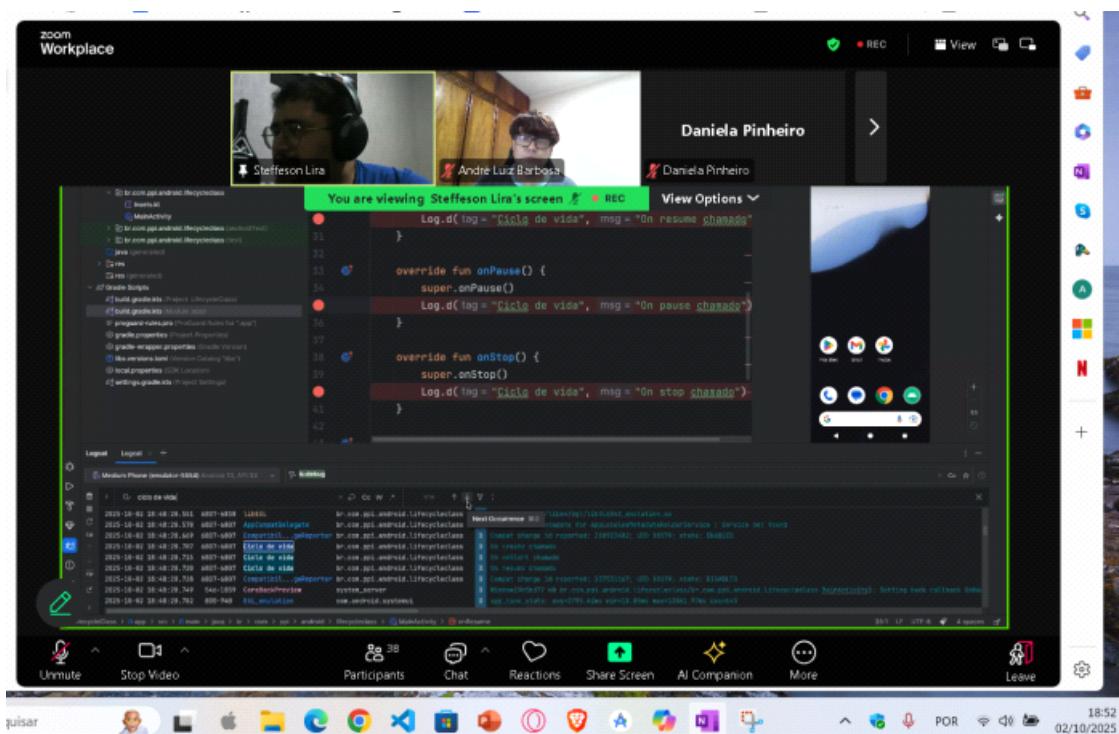
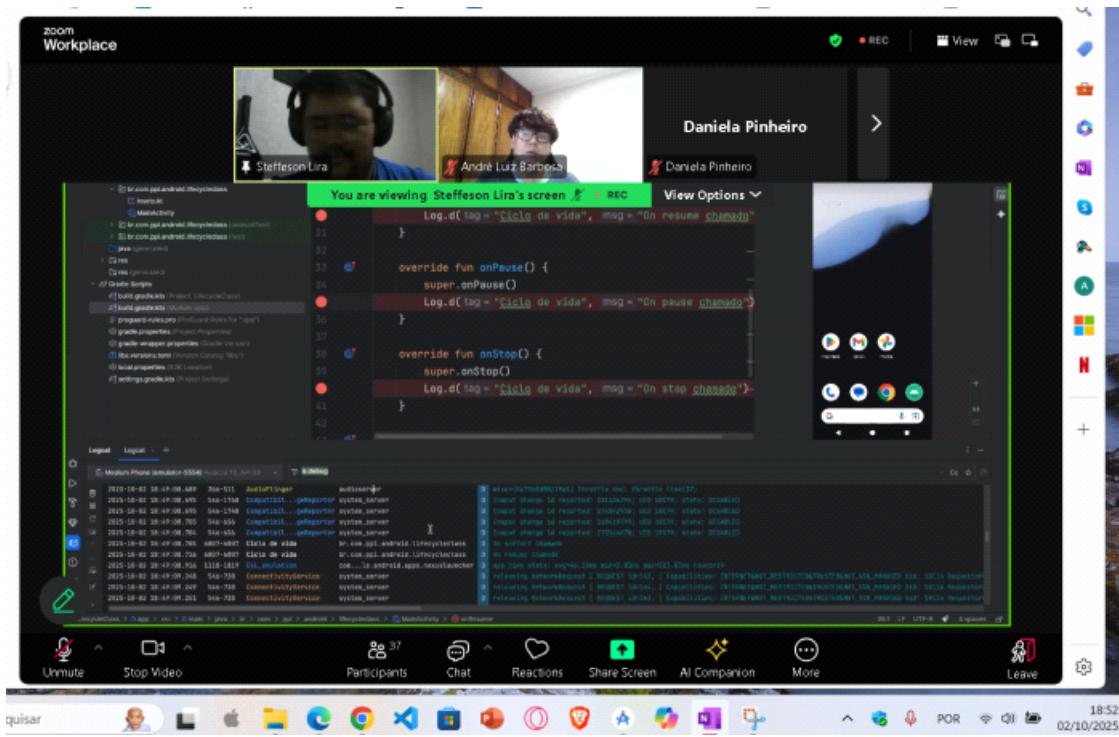
Para encontrar bugs, óbvio

Para fazer isto clicar no número onde quer que o ponto seja visto, o sistema vai parar até autorizar seguir





Mirimzou o app deu um pause e um stop e não o destroy  
Voltou no app veio direto no star, no resume e se emnccrar perde o debug



zoom  
Workplace

You are viewing Steffeson Lira's screen REC View Options

- A activity é criada e outras tarefas de inicialização são executadas
- Você deve implementar esse callback
- Inflar a interface da activity e executar outras lógicas de inicialização do app

venturus

Unmute Stop Video Participants Chat Reactions Share Screen AI Companion More Leave

18:53 02/10/2025

## onStart()

- A activity torna-se visível para o usuário
- Chamado após a activity:
  - onCreate()
  - ou
  - onRestart() se a activity estava previamente parada

venturus

## onResume()

- A activity ganha foco de entrada:
  - A activity recebe foco para entrada:
- A activity permanece no estado resumido até o sistema acionar a pausa da activity

venturus

You are viewing Steffeson Lira's screen REC View Options

## onPause()

- A activity perdeu o foco (não está em primeiro plano)□
- A activity ainda está visível, mas o usuário não está interagindo ativamente com ela
- Equivalente ao onResume()

venturus

You are viewing Steffeson Lira's screen REC View Options

## onStop()

- A activity não está mais visível para o usuário
- Liberar recursos que não são mais necessários
- Salvar qualquer estado persistente que o usuário esteja editando para que ele não perca seu trabalho

venturus

You are viewing Steffeson Lira's screen REC View Options

## onDestroy()

- A activity está prestes a ser destruída, o que pode ser causado por:
  - A activity foi finalizada ou descartada pelo usuário
  - Mudança de configuração
- Realize a limpeza final dos recursos.□
- Não confie neste método para salvar dados do usuário (faça isso antes)

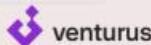
venturus

Ter cuidado com persistência de dados

You are viewing Steffeson Lira's screen REC View Options

## Resumo dos estados da activity

State	Callbacks	Description
Criada	onCreate()	A activity está sendo inicializada.
Iniciada	onStart()	A activity está visível para o usuário.
Ativa	onResume()	A activity tem o foco para entrada de dados.
Pausada	onPause()	A activity não possui foco para entrada de dados.
Parada	onStop()	A activity deixou de estar visível.
Finalizada	onDestroy()	A activity foi finalizada.





Voltando para a aplicação

Fcar claro o ciclo de vida, comparando com o comportamento na escola, o onCreate é quando chega na escola, onStart é quando chega na sala de aula, onXXX é quando senta na cadeira, on quando vai no banheiro, é quando o destroy é quando temrina

You are viewing Steffeson Lira's screen REC View Options

## Save state

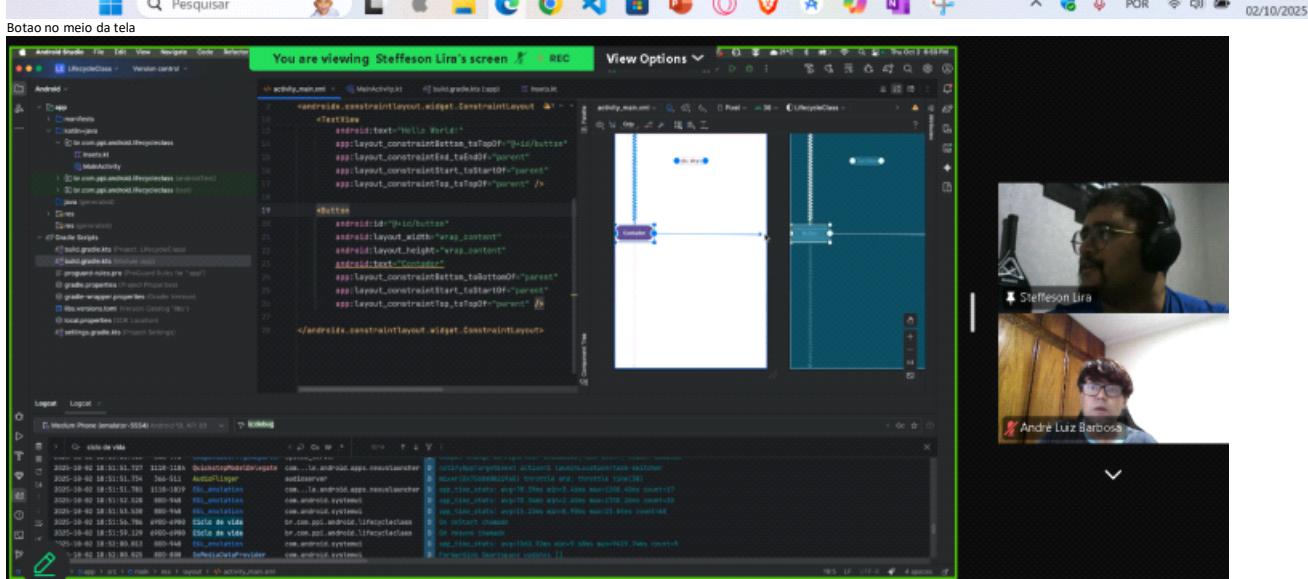
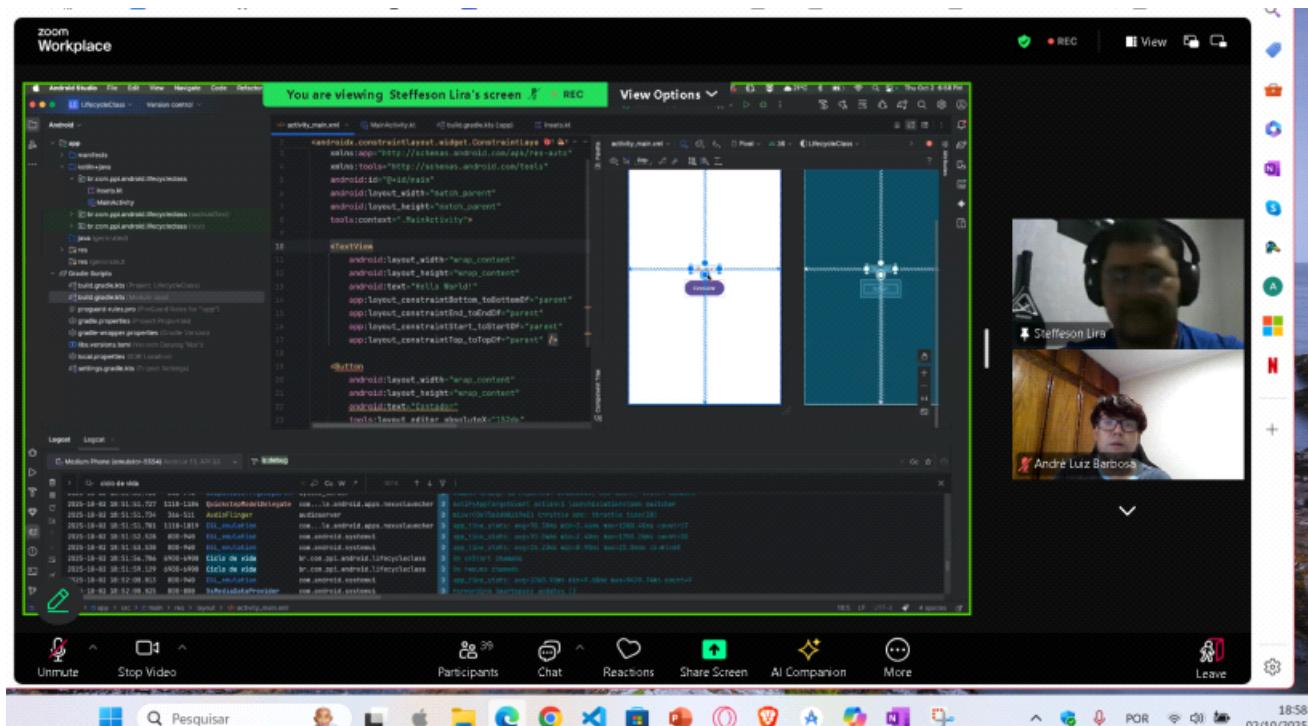
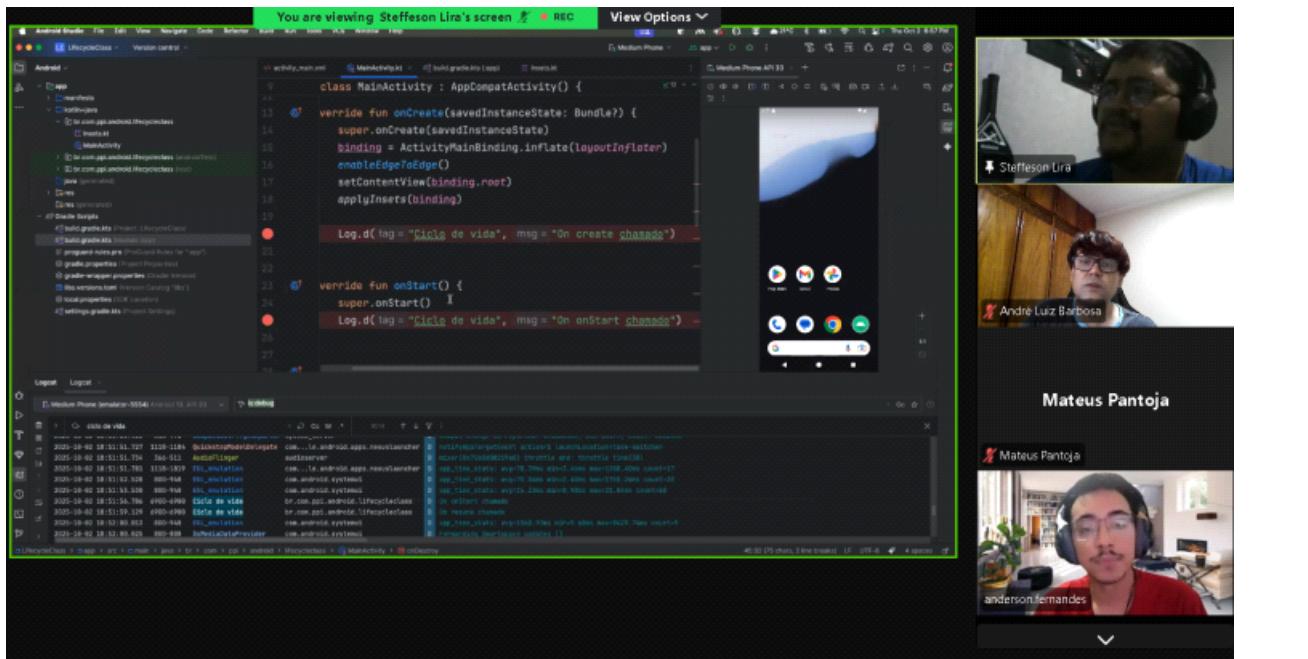
O usuário espera que a interface permaneça igual após mudança de configuração ou encerramento do app em segundo plano.

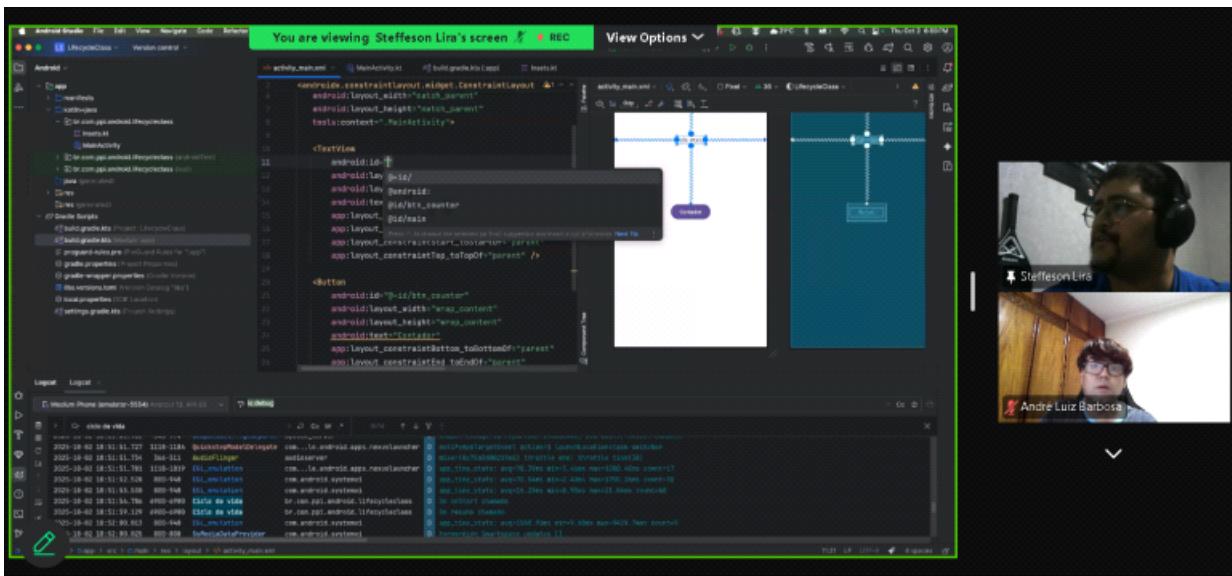
- A activity é destruída e reiniciada, ou o app é encerrado e a activity é iniciada.
- Armazene os dados do usuário necessários para reconstruir o app e as mudanças no ciclo de vida da activity:
  - Use Bundle fornecido pelo onSaveInstanceState().
  - onCreate() recebe o Bundle como argumento quando a activity é criada novamente.



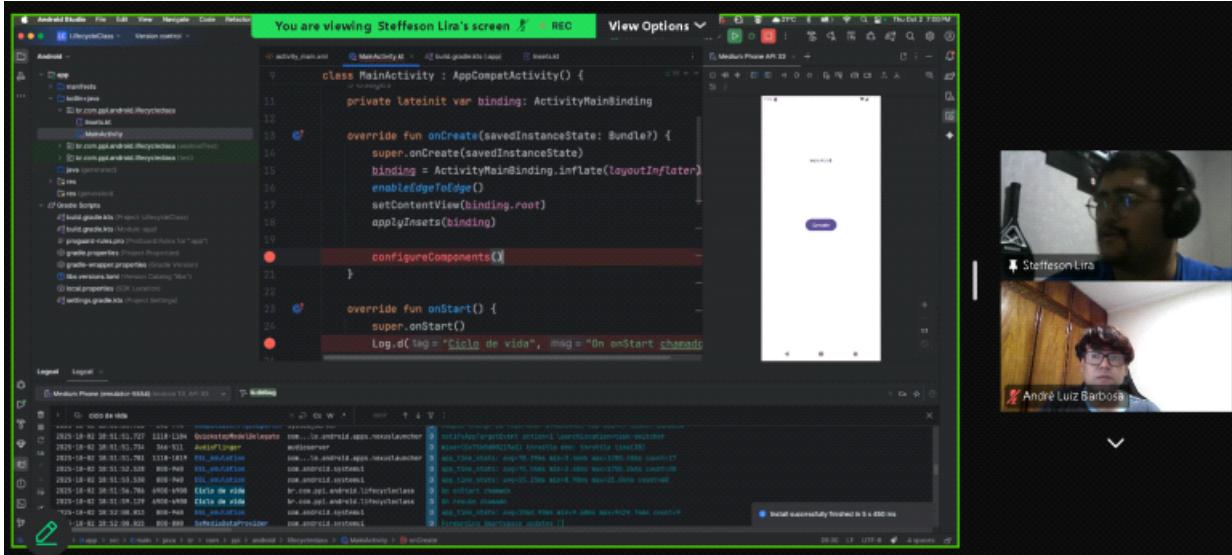


O que tem que ter é sempre o onCreate

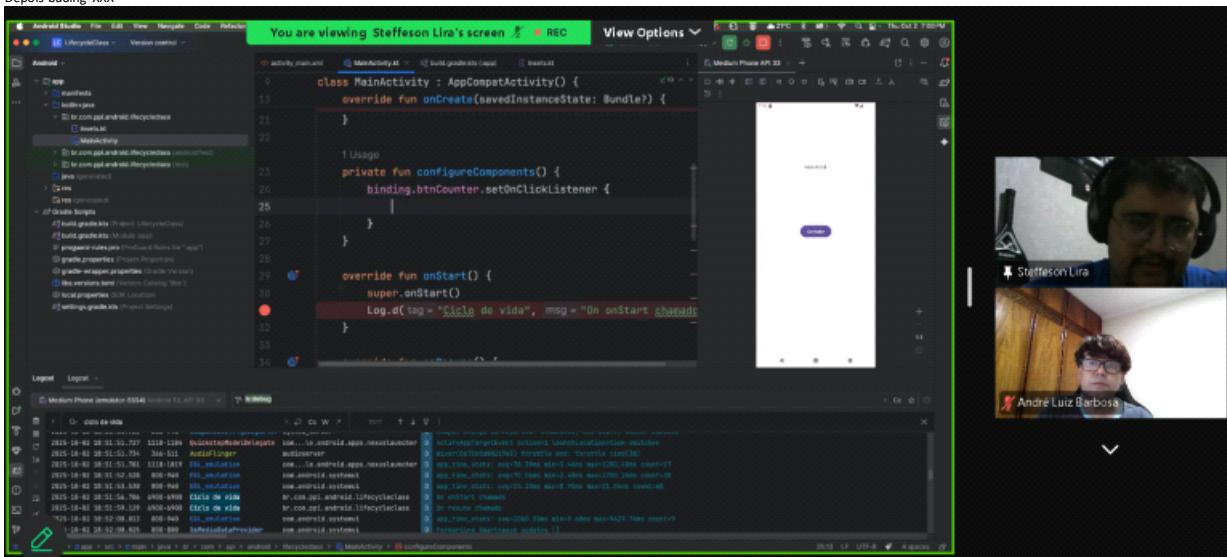




Fez configure components



Depois buding XXX



Para que? O contador?

The screenshot shows the Android Studio interface. On the left is the project structure for 'LifecycleClass'. The main area displays the code for `MainActivity.kt`. A video call window on the right shows André Luiz Barbosa wearing headphones. The bottom status bar indicates the device is connected to a 'Medium Phone'.

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        binding = ActivityMainBinding.inflate(layoutInflater)
        enableEdgeToEdge()
        setContentView(binding.root)
        applyInsets(binding)
    }

    private fun configureComponents() {
        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }
}
```

Professor disse que o contador é para transição para ver algio que para não perder

The screenshot shows the Android Studio interface. On the left is the project structure for 'LifecycleClass'. The main area displays the code for `MainActivity.kt`. A video call window on the right shows André Luiz Barbosa wearing headphones. The bottom status bar indicates the device is connected to a 'Medium Phone'.

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        binding = ActivityMainBinding.inflate(layoutInflater)
        enableEdgeToEdge()
        setContentView(binding.root)
        applyInsets(binding)
    }

    private fun configureComponents() {
        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }
}
```

Vai mostrar cm os savings e se os savings

The screenshot shows the Android Studio interface. On the left is the project structure for 'LifecycleClass'. The main area displays the code for `MainActivity.kt`. A video call window on the right shows André Luiz Barbosa wearing headphones. The bottom status bar indicates the device is connected to a 'Medium Phone'.

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        binding = ActivityMainBinding.inflate(layoutInflater)
        enableEdgeToEdge()
        setContentView(binding.root)
        applyInsets(binding)
    }

    private fun configureComponents() {
        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }
}

override fun onStart() {
    super.onStart()
    Log.d(tag = "Ciclo de vida", msg = "On start chamado")
}

override fun onResume() {
    super.onResume()
    Log.d(tag = "Ciclo de vida", msg = "On resume chamado")
}

override fun onPause() {
    super.onPause()
    Log.d(tag = "Ciclo de vida", msg = "On pause chamado")
}
```

You are viewing Steffeson Lira's screen REC View Options

```
class MainActivity : AppCompatActivity() {  
    private fun configureComponents() {  
        binding.btnCounter.setOnClickListener {  
            counter++  
            binding.txtExample.text = "Contador: $counter"  
        }  
    }  
  
    override fun onStart() {  
        super.onStart()  
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")  
    }  
  
    override fun onResume() {  
        super.onResume()  
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")  
    }  
  
    override fun onPause() {  
        super.onPause()  
        Log.d(tag = "Ciclo de vida", msg = "On pause chamado")  
    }  
}
```

Se deixa o celular perder

You are viewing Steffeson Lira's screen REC View Options

```
class MainActivity : AppCompatActivity() {  
    private fun configureComponents() {  
        binding.btnCounter.setOnClickListener {  
            counter++  
            binding.txtExample.text = "Contador: $counter"  
        }  
    }  
  
    override fun onStart() {  
        super.onStart()  
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")  
    }  
  
    override fun onResume() {  
        super.onResume()  
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")  
    }  
  
    override fun onPause() {  
        super.onPause()  
        Log.d(tag = "Ciclo de vida", msg = "On pause chamado")  
    }  
}
```

You are viewing Steffeson Lira's screen REC View Options

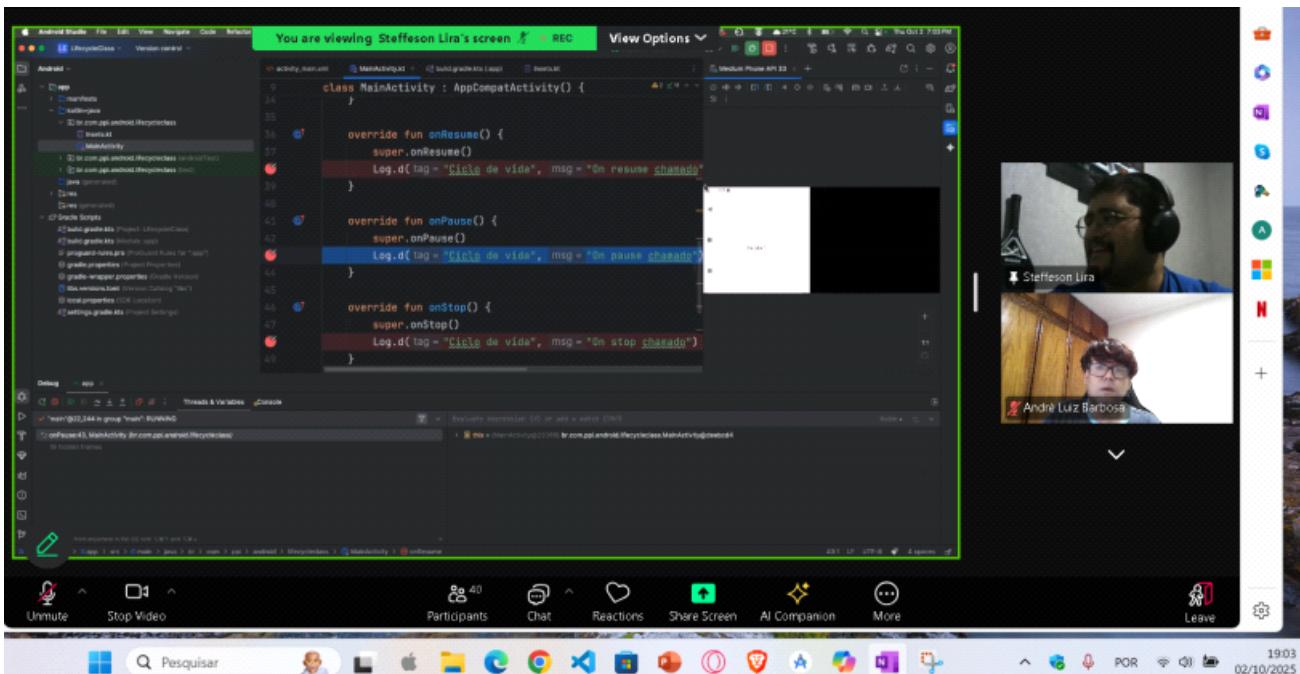
```
class MainActivity : AppCompatActivity() {  
    private fun configureComponents() {  
        binding.btnCounter.setOnClickListener {  
            counter++  
            binding.txtExample.text = "Contador: $counter"  
        }  
    }  
  
    override fun onStart() {  
        super.onStart()  
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")  
    }  
  
    override fun onResume() {  
        super.onResume()  
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")  
    }  
}
```

Debug app

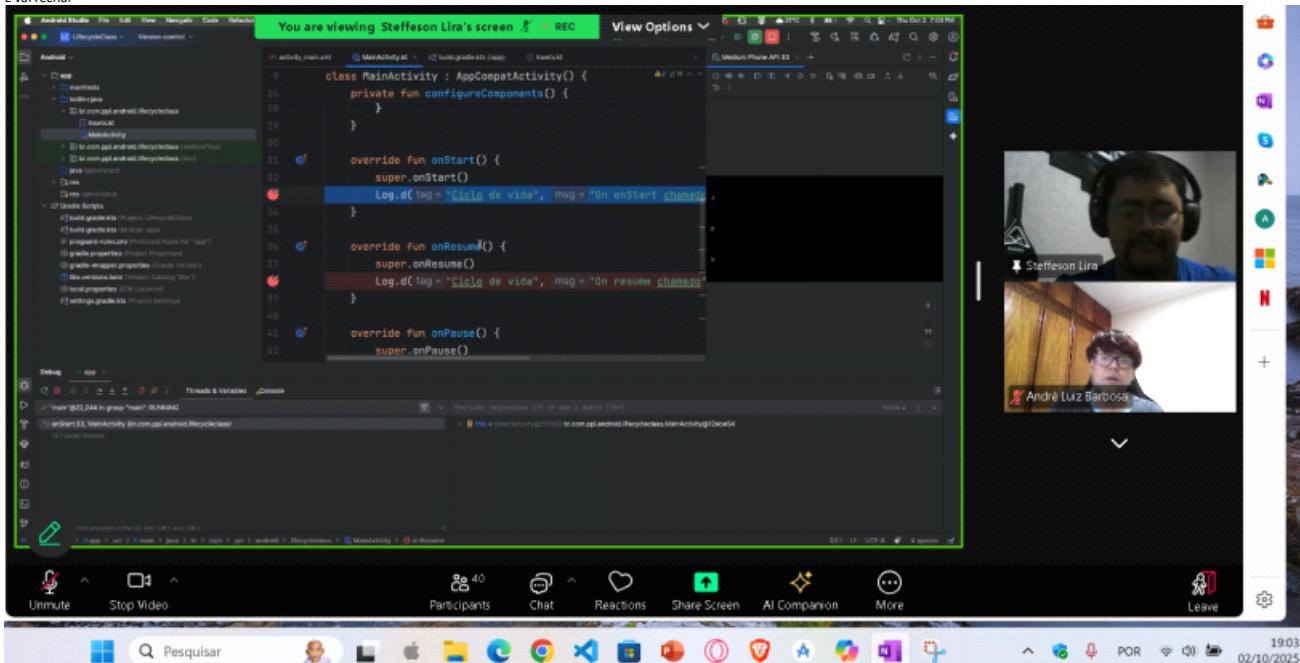
Stepping expression 65 or less a seconds (her)

configureComponents is not called, MainActivity.onResume() is called

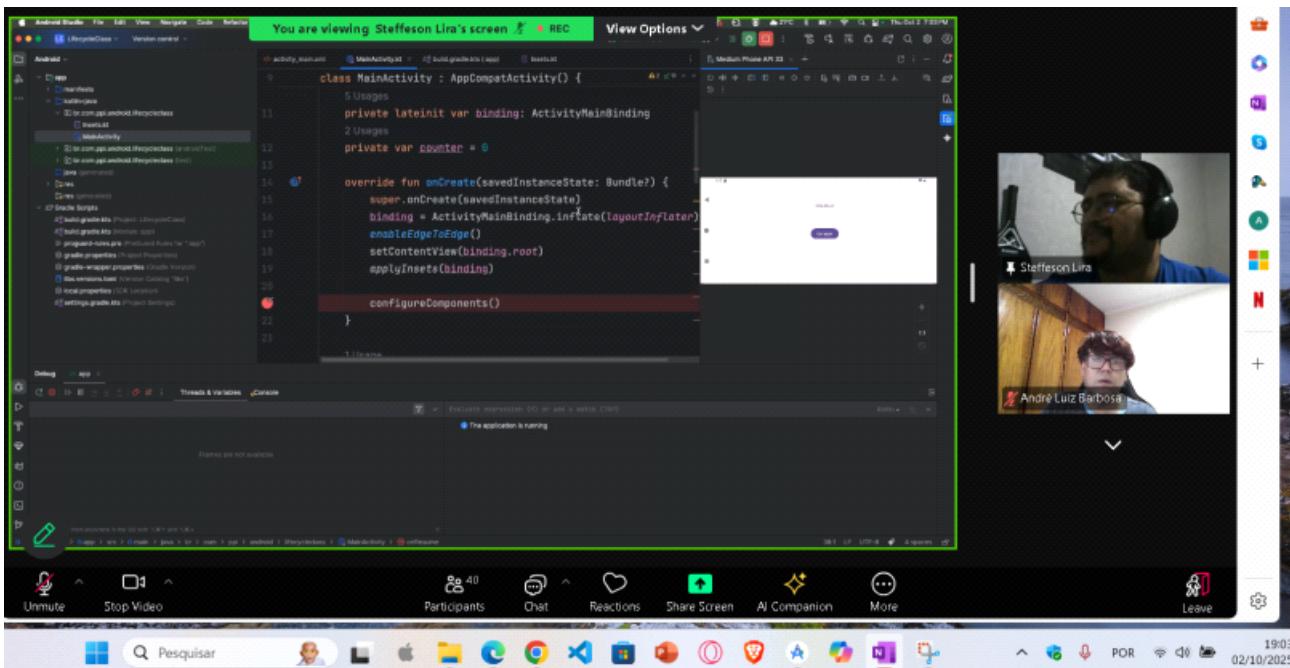
MainActivity.onResume() is called, collecting data...



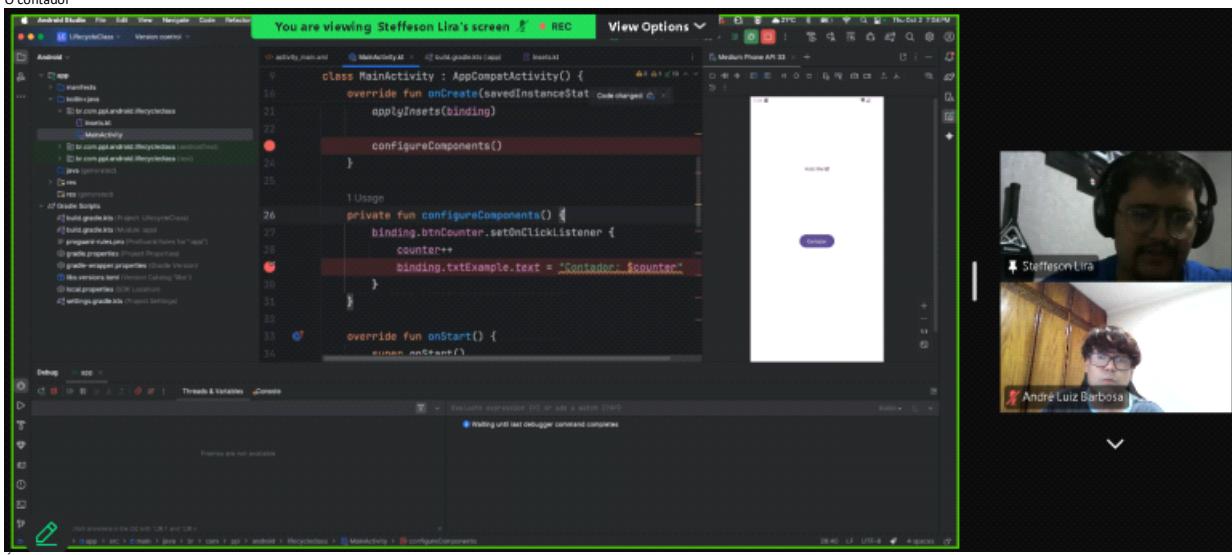
Rodou a tela ocorre um pause, e destroi  
E vai recarregar



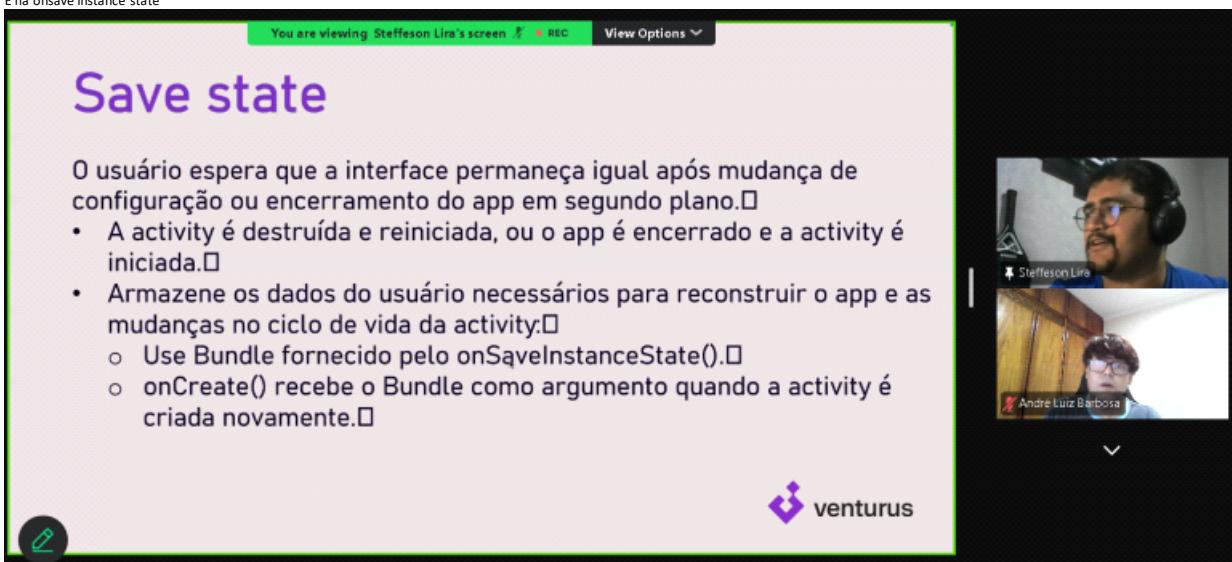
E vai ter que iniciar do zero



O contador



É na onSaveInstanceState()



The screenshot shows the Android Studio interface with the code editor open to `activity_main.kt`. The code defines a `MainActivity` that overrides `onStop()`, `onDestroy()`, and `onSaveInstanceState()` methods to log messages. The preview window shows a simple UI with a button labeled "Iniciar". On the right, there are two video feeds: one for Steffeson Lira and one for André Luiz Barbosa.

```
class MainActivity : AppCompatActivity() {  
    override fun onStop() {  
        super.onStop()  
        Log.d(tag = "Ciclo de vida", msg = "On stop chamado")  
    }  
  
    override fun onDestroy() {  
        super.onDestroy()  
        Log.d(tag = "Ciclo de vida", msg = "On destroy chamado")  
    }  
  
    override fun onSaveInstanceState(savedInstanceState: Bundle) {  
        super.onSaveInstanceState(savedInstanceState)  
        Log.d(tag = "Ciclo de vida", msg = "On save instance state chamado")  
    }  
}
```

The screenshot shows the Android Studio interface with the code editor open to `activity_main.kt`. The code defines a `MainActivity` that overrides `onDestroy()` and `onSaveInstanceState()` methods to log messages. The preview window shows a simple UI with a button labeled "Iniciar". On the right, there are two video feeds: one for Steffeson Lira and one for André Luiz Barbosa.

```
class MainActivity : AppCompatActivity() {  
    override fun onDestroy() {  
        super.onDestroy()  
        Log.d(tag = "Ciclo de vida", msg = "On destroy chamado")  
    }  
  
    override fun onSaveInstanceState(savedInstanceState: Bundle) {  
        super.onSaveInstanceState(savedInstanceState)  
        Log.d(tag = "Ciclo de vida", msg = "On save instance state chamado")  
    }  
}
```

Tem o do bundle e da persistencia  
So o bundle para este exemplo serve

The screenshot shows the Android Studio interface with the code editor open to `activity_main.kt`. The code defines a `MainActivity` that overrides `onDestroy()` and `onSaveInstanceState()` methods to log messages. The preview window shows a simple UI with a button labeled "Iniciar". On the right, there are two video feeds: one for Steffeson Lira and one for André Luiz Barbosa.

```
class MainActivity : AppCompatActivity() {  
    override fun onDestroy() {  
        super.onDestroy()  
        Log.d(tag = "Ciclo de vida", msg = "On destroy chamado")  
    }  
  
    override fun onSaveInstanceState(savedInstanceState: Bundle) {  
        super.onSaveInstanceState(savedInstanceState)  
        Log.d(tag = "Ciclo de vida", msg = "On save instance state chamado")  
    }  
}
```

Vai criar uma ??? Variavel para manipular ela quem?

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        Log.d(tag = "Ciclo de vida", msg = "On create change")  
    }  
  
    override fun onDestroy() {  
        super.onDestroy()  
        Log.d(tag = "Ciclo de vida", msg = "On destroy change")  
    }  
  
    override fun onSaveInstanceState(savedInstanceState: Bundle) {  
        super.onSaveInstanceState(savedInstanceState)  
        savedInstanceState.putInt(KEY_CONTADOR, counter)  
    }  
}
```

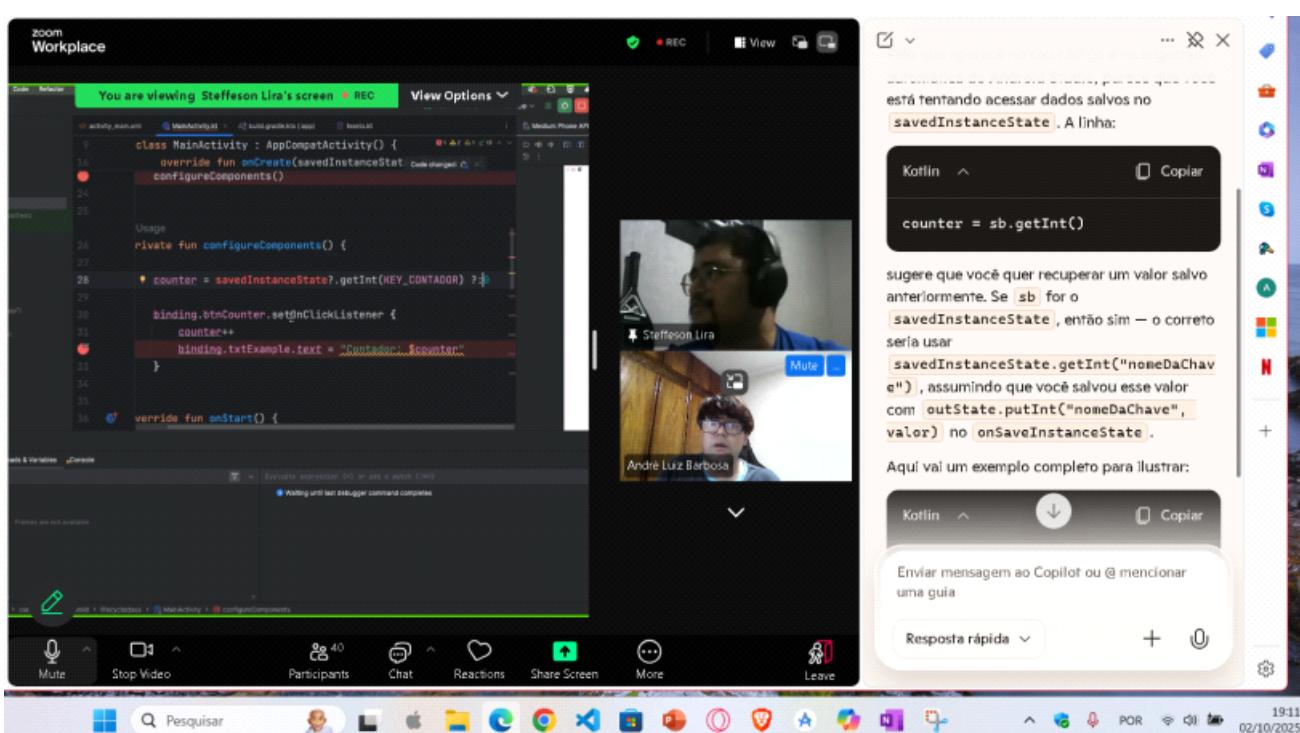
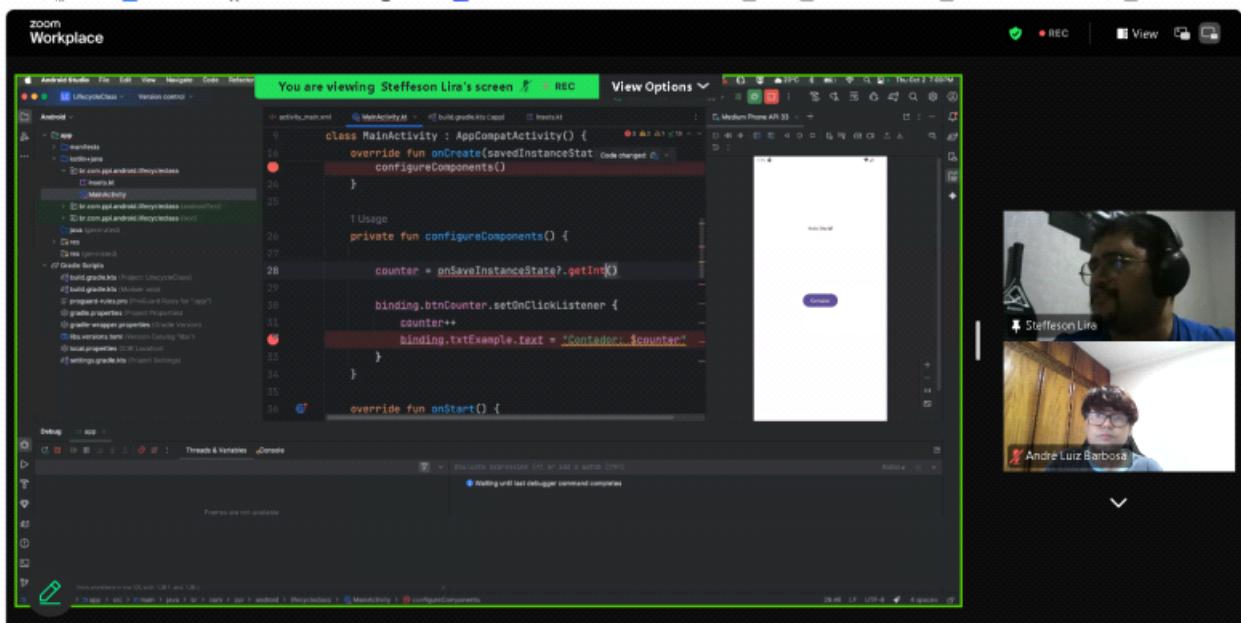
Querer que salve o contador e vai ter o identificador =key\_contador

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        Log.d(tag = "Ciclo de vida", msg = "On create change")  
    }  
  
    override fun onDestroy() {  
        super.onDestroy()  
        Log.d(tag = "Ciclo de vida", msg = "On destroy change")  
    }  
  
    override fun onSaveInstanceState(savedInstanceState: Bundle) {  
        super.onSaveInstanceState(savedInstanceState)  
        savedInstanceState.putInt(KEY_CONTADOR, counter)  
    }  
}
```

O out state é um bundle, dafuncionalidade para o parametro, tipo pegar info e poder manipular o negocio (qual???)

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        configureComponents()  
    }  
  
    private fun configureComponents() {  
        counter = 0  
        binding.btnCounter.setOnClickListener {  
            counter++  
            binding.txtExample.text = "Contador: $counter"  
        }  
    }  
}
```

Precisa pegar o que salvou e verifica se é nulo e ao invés de ir out vai fazer um get int



You are viewing Steffeson Lira's screen REC View Options

```

    class MainActivity : AppCompatActivity() {
        override fun onCreate(savedInstanceState: Bundle?) {
            binding = ActivityMainBinding.inflate(layoutInflater)
            enableEdgeToEdge()
            setContentView(binding.root)
            applyInsets(binding)

            configureComponents(savedInstanceState)
        }

        private fun configureComponents(savedInstanceState: Bundle?) {
            binding.btnCounter.setOnClickListener {
                counter++
                binding.txtExample.text = "Contador: $counter"
            }
        }

        override fun onStart() {
    
```

Mute Stop Video Participants Chat Reactions Share Screen More

Pesquisar 19:11 02/10/2025

Precisa de um this  
mas o recisa da activity precisa ser criado no onCreate  
Precisa do bundle  
Quando cria uma função não esquecer de criar a variável

You are viewing Steffeson Lira's screen REC View Options

```

    class MainActivity : AppCompatActivity() {
        override fun onCreate(savedInstanceState: Bundle?) {
            binding = ActivityMainBinding.inflate(layoutInflater)
            enableEdgeToEdge()
            setContentView(binding.root)
            applyInsets(binding)

            configureComponents(savedInstanceState)
        }

        private fun configureComponents(savedInstanceState: Bundle?) {
            binding.btnCounter.setOnClickListener {
                counter++
                binding.txtExample.text = "Contador: $counter"
            }
        }

        override fun onStart() {
    
```

Mute Stop Video Participants Chat Reactions Share Screen More

Sim, exatamente! A criação da Activity e a configuração inicial dos componentes devem acontecer dentro do método onCreate. Esse método é chamado quando a Activity está sendo criada pela primeira vez — é o ponto ideal para:

- Inflar o layout com setContentView.
- Inicializar o ViewBinding (como você fez com ActivityMainBinding.inflate).
- Restaurar estados salvos com savedInstanceState.
- Configurar listeners e lógica de interface com métodos como configureComponents().

No seu exemplo:

```
override fun onCreate(savedInstanceState: Bundle?) {
```

```
    super.onCreate(savedInstanceState)
```

```
    binding = ActivityMainBinding.inflate(layoutInflater)
```

```
    setContentView(binding.root)
```

```
    applyInsets(binding)
```

```
    counter = savedInstanceState.getInt(key = KEY_COUNTER) ?: 0
```

```
    configureComponents()
```

Está certinho! Tudo que precisa estar pronto para o usuário interagir com a tela deve ser configurado aqui. Se você deixar isso para outro momento do ciclo de vida (como onStart ou onResume), pode acabar com comportamentos inesperados ou até erros de interface.

Se quiser, posso te mostrar um fluxograma do ciclo de vida da Activity para visualizar melhor onde cada coisa entra.

De <<https://copilot.microsoft.com/chats/e19yspcn99fqSjScgZsZl>>

O professor está certo — quando você cria uma função que depende de um Bundle, como o savedInstanceState, é essencial declarar e usar essa variável corretamente. No contexto do Android, isso geralmente aparece no ciclo de vida da Activity, especialmente no onCreate.

Aqui vai uma explicação prática:

#### Exemplo correto com Bundle:

```
override fun onCreate(savedInstanceState: Bundle?) {
```

```
    super.onCreate(savedInstanceState)
```

```
    val counterFromBundle = savedInstanceState.getInt("counter") ?: 0
```

```
    counter = counterFromBundle
```

#### Erro comum:

Criar uma função que tenta acessar savedInstanceState sem ter a variável declarada ou passada como parâmetro:

```
private fun restoreCounter() {
```

```
    // Erro: savedInstanceState não existe aqui
```

```
    counter = savedInstanceState.getInt("counter") ?: 0
```

#### Correção:

Você precisa passar o Bundle como argumento:

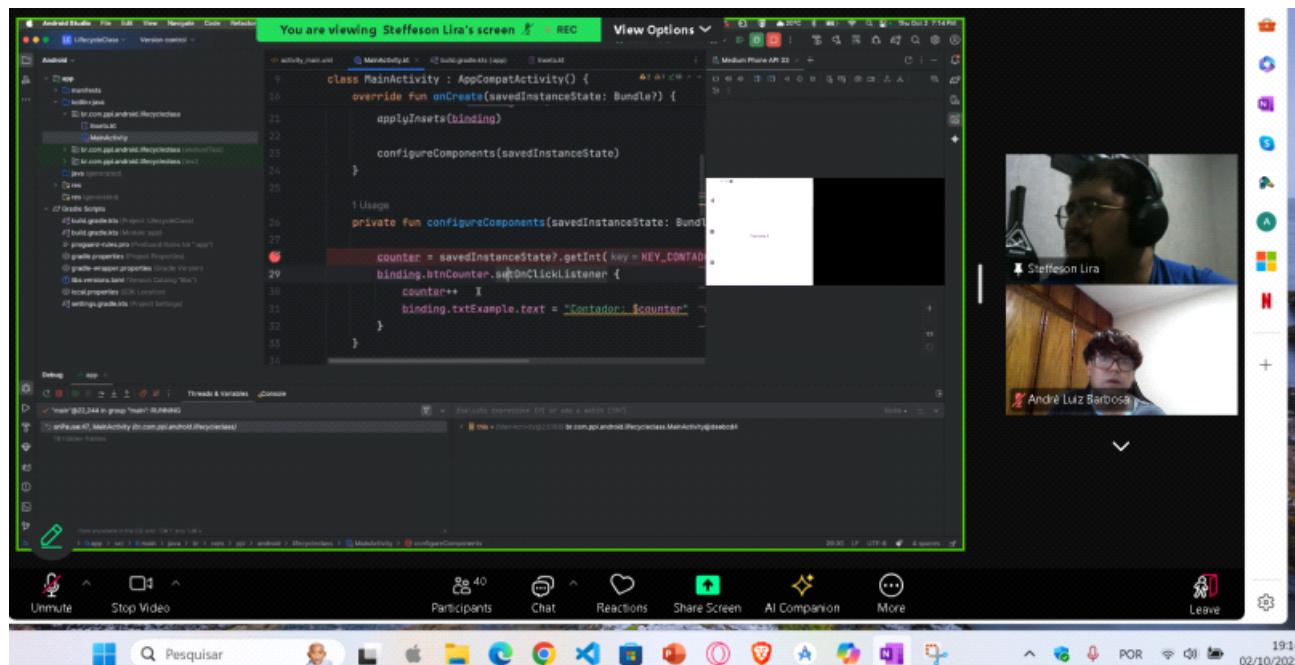
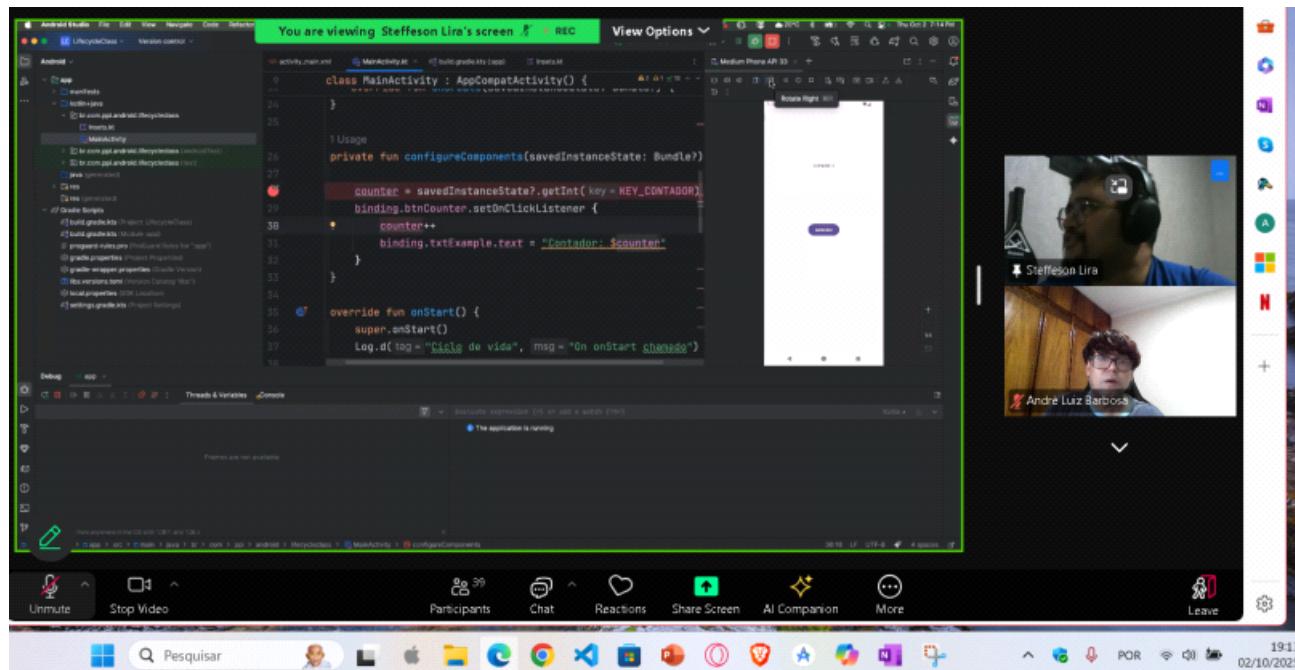
```
private fun restoreCounter(savedInstanceState: Bundle?) {
    counter = savedInstanceState?.getInt("counter") ?: 0
}
```

E chamar assim:

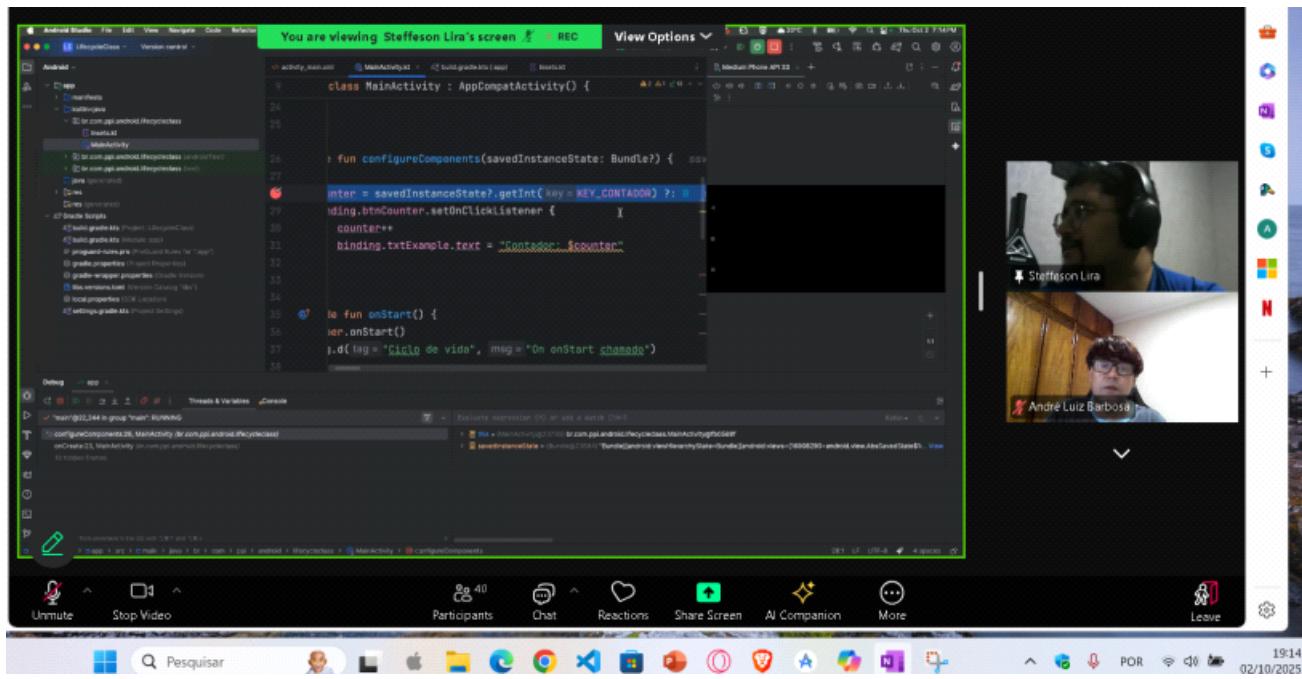
```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    restoreCounter(savedInstanceState)
}
```

Se quiser, posso te ajudar a refatorar seu código para seguir essa estrutura certinha. Só me mostrar como está sua função atual.

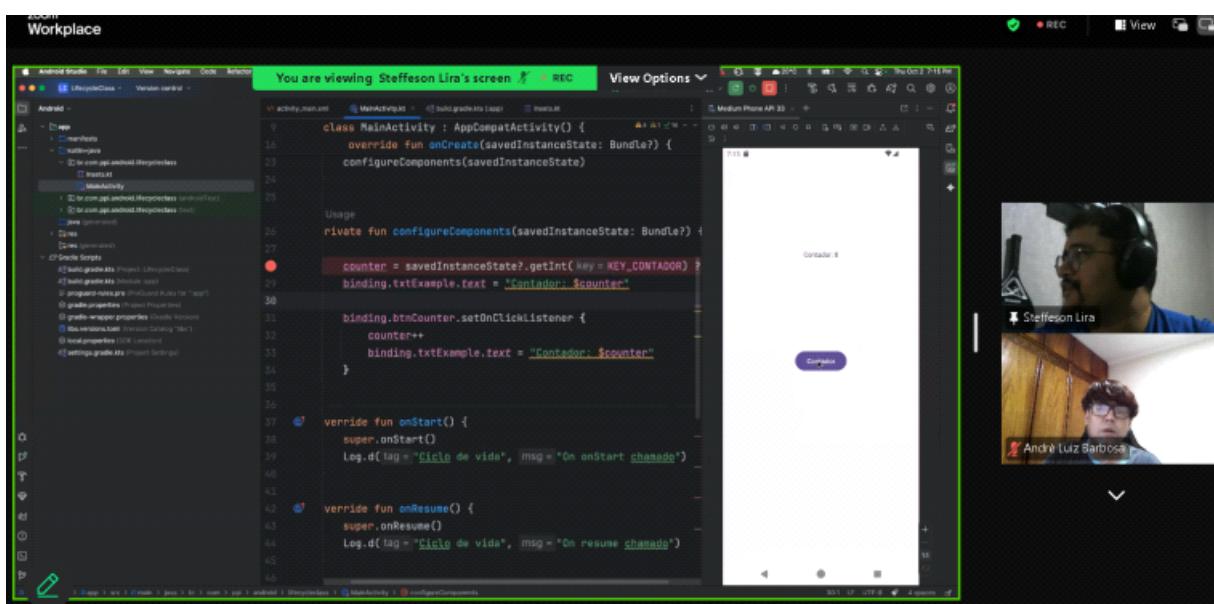
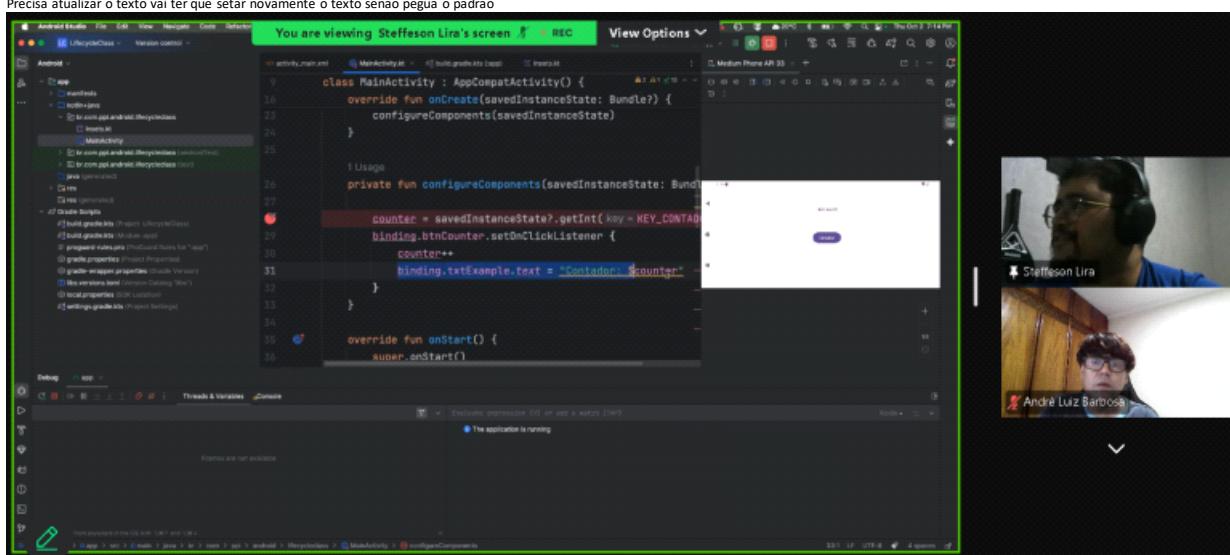
De <<https://copilot.microsoft.com/chats/e19yspcn99fq5j5cg2szl>>



Deu o destroy



Deu f8  
Tá com 3 (Não entendi)  
Precisa atualizar o texto vai ter que setar novamente o texto senão pegua o padrao



You are viewing Steffeson Lira's screen REC View Options

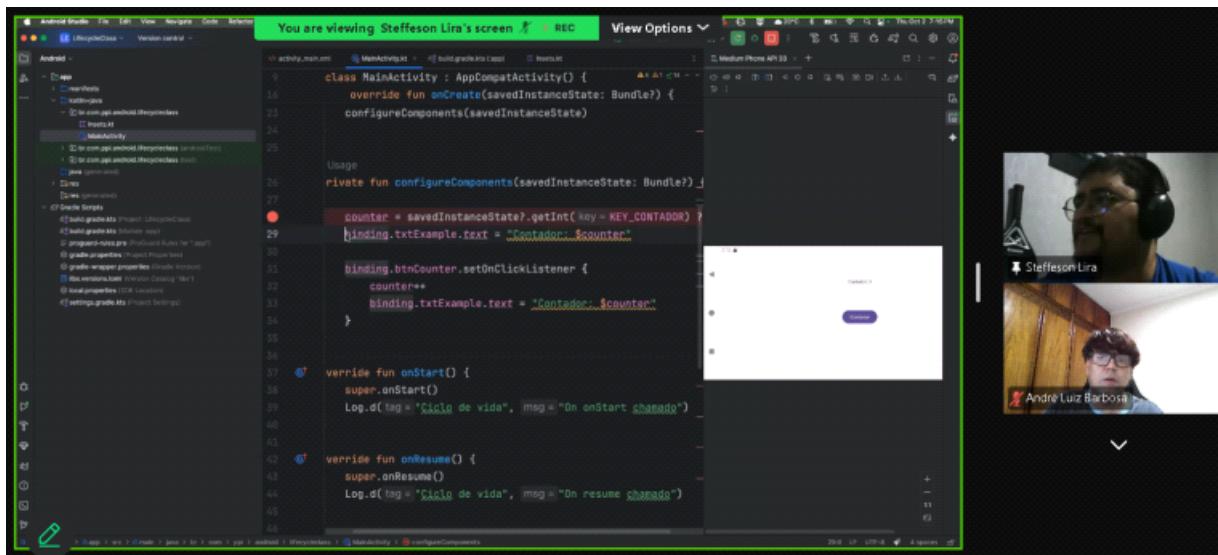
```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        configureComponents(savedInstanceState)
    }

    private fun configureComponents(savedInstanceState: Bundle?) {
        counter = savedInstanceState.getInt(key = KEY_CONTADOR)
        binding.txtExample.text = "Contador: $counter"

        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }

    override fun onStart() {
        super.onStart()
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")
    }

    override fun onResume() {
        super.onResume()
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")
    }
}
```



You are viewing Steffeson Lira's screen REC View Options

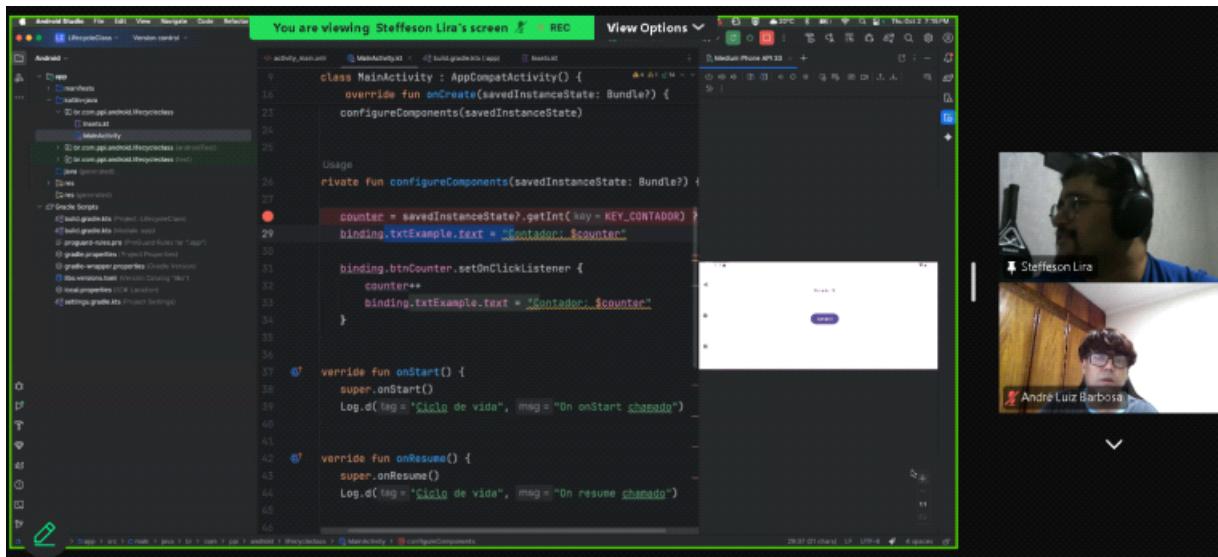
```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        configureComponents(savedInstanceState)
    }

    private fun configureComponents(savedInstanceState: Bundle?) {
        counter = savedInstanceState.getInt(key = KEY_CONTADOR)
        binding.txtExample.text = "Contador: $counter"

        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }

    override fun onStart() {
        super.onStart()
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")
    }

    override fun onResume() {
        super.onResume()
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")
    }
}
```



Girou e contador continuou

You are viewing Steffeson Lira's screen REC View Options

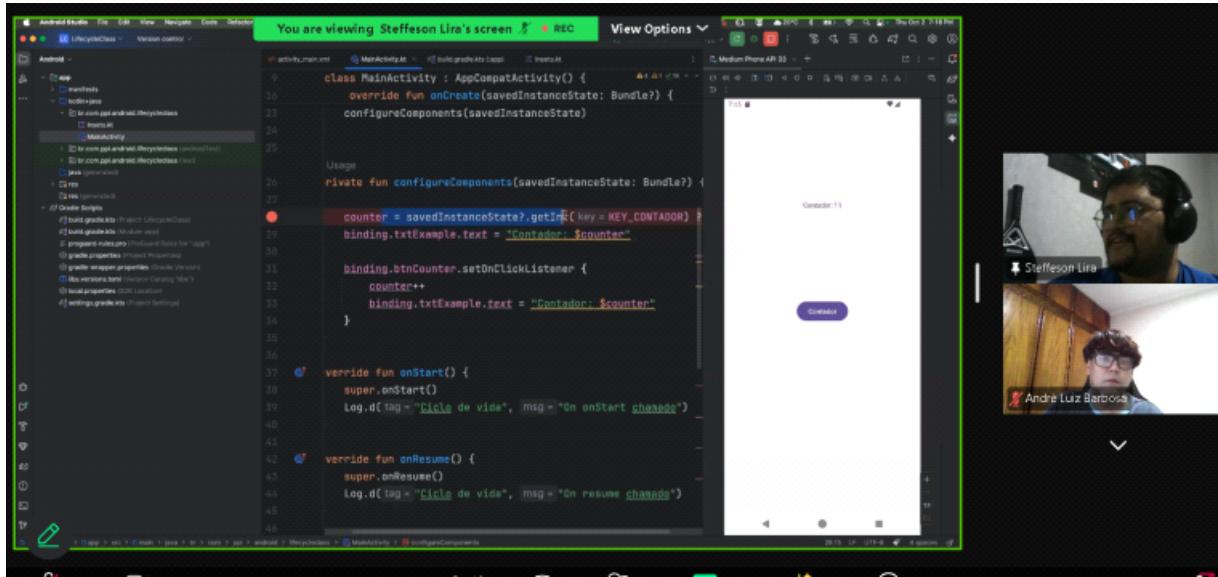
```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        configureComponents(savedInstanceState)
    }

    private fun configureComponents(savedInstanceState: Bundle?) {
        counter = savedInstanceState.getInt(key = KEY_CONTADOR)
        binding.txtExample.text = "Contador: $counter"

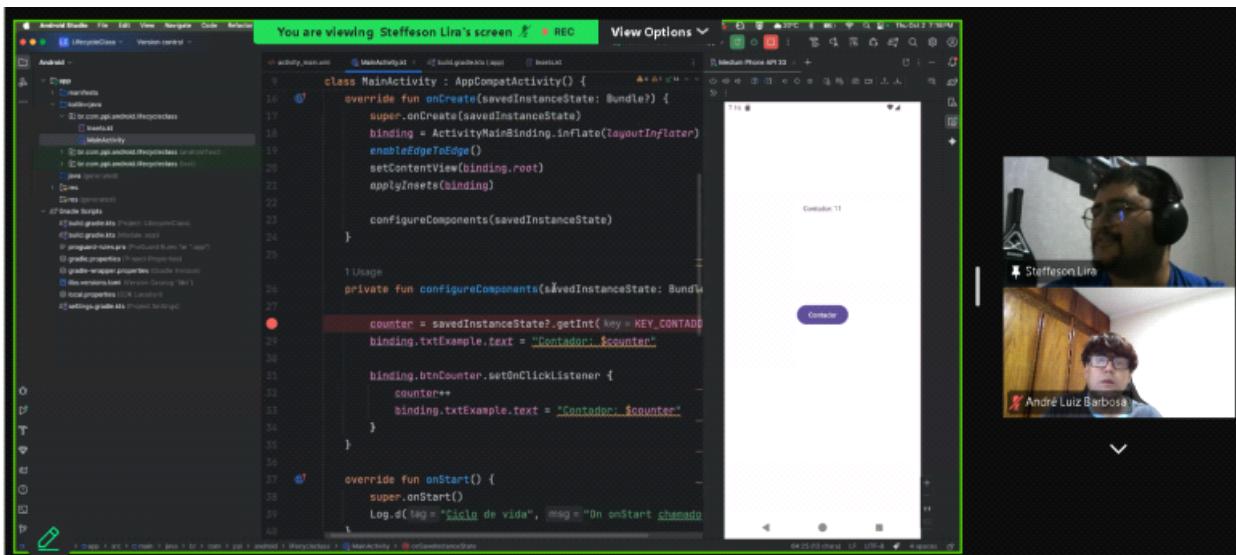
        binding.btnCounter.setOnClickListener {
            counter++
            binding.txtExample.text = "Contador: $counter"
        }
    }

    override fun onStart() {
        super.onStart()
        Log.d(tag = "Ciclo de vida", msg = "On onStart chamado")
    }

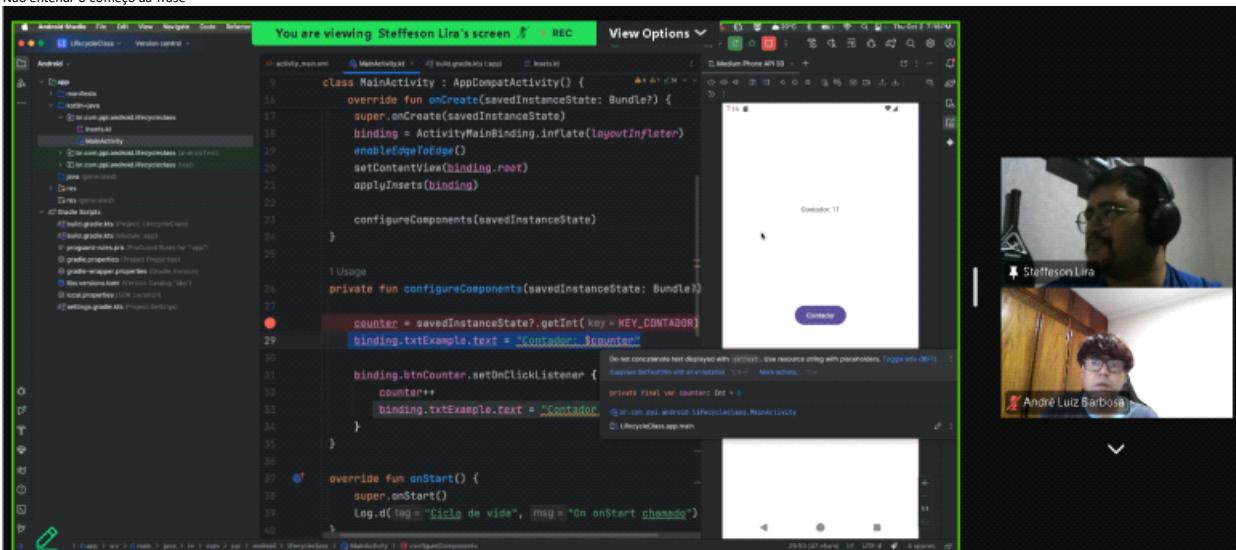
    override fun onResume() {
        super.onResume()
        Log.d(tag = "Ciclo de vida", msg = "On resume chamado")
    }
}
```



Bundlé que faz a função não sei qual put ????

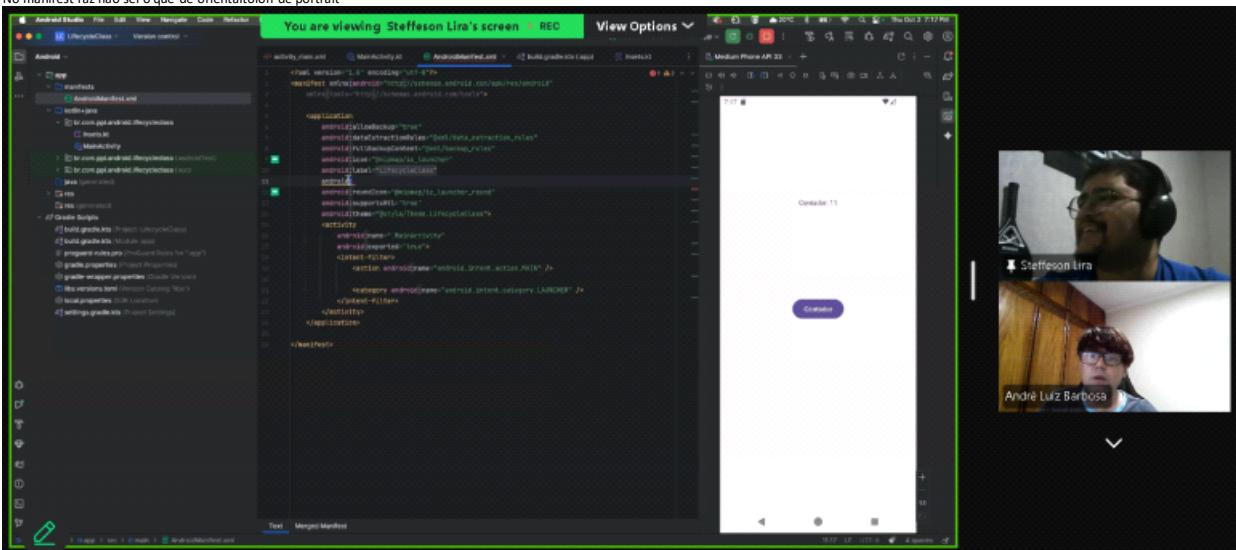


Não entendi o começo da frase



Mas joga na variável que quer

No manifest faz não sei o que de orientação de portrait



Se você quer definir a orientação da tela da sua Activity no Android, isso é feito no arquivo `AndroidManifest.xml`, dentro da tag da Activity. Você pode forçar a orientação para **portrait** (vertical) ou **landscape** (horizontal), dependendo do comportamento desejado.

#### Exemplo: Forçar orientação vertical (portrait)

```
<activity
    android:name=".MainActivity"
    android:screenOrientation="portrait" />
```

#### Exemplo: Forçar orientação horizontal (landscape)

```
<activity
    android:name=".MainActivity"
    android:screenOrientation="landscape" />
```

#### Outras opções disponíveis:

- unspecified: deixa o sistema decidir
- sensor: muda conforme o sensor do dispositivo

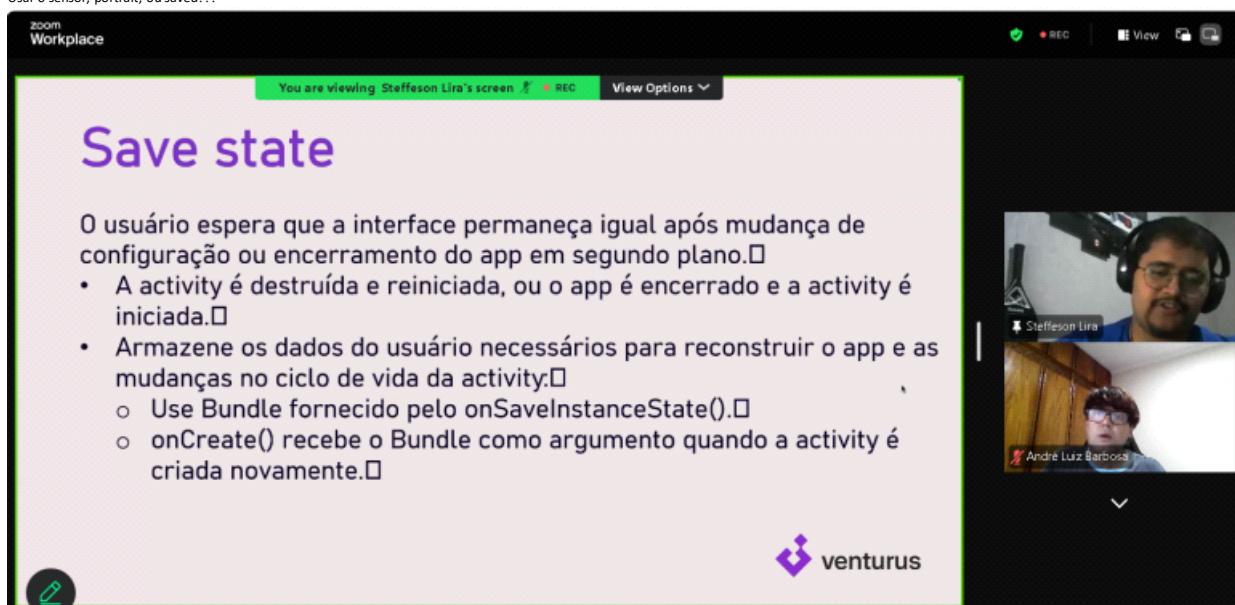
- fullSensor: permite todas orientações, inclusive reverse
  - user: respeita a preferência do usuário

- behind: herda a orientação da Activity anterior

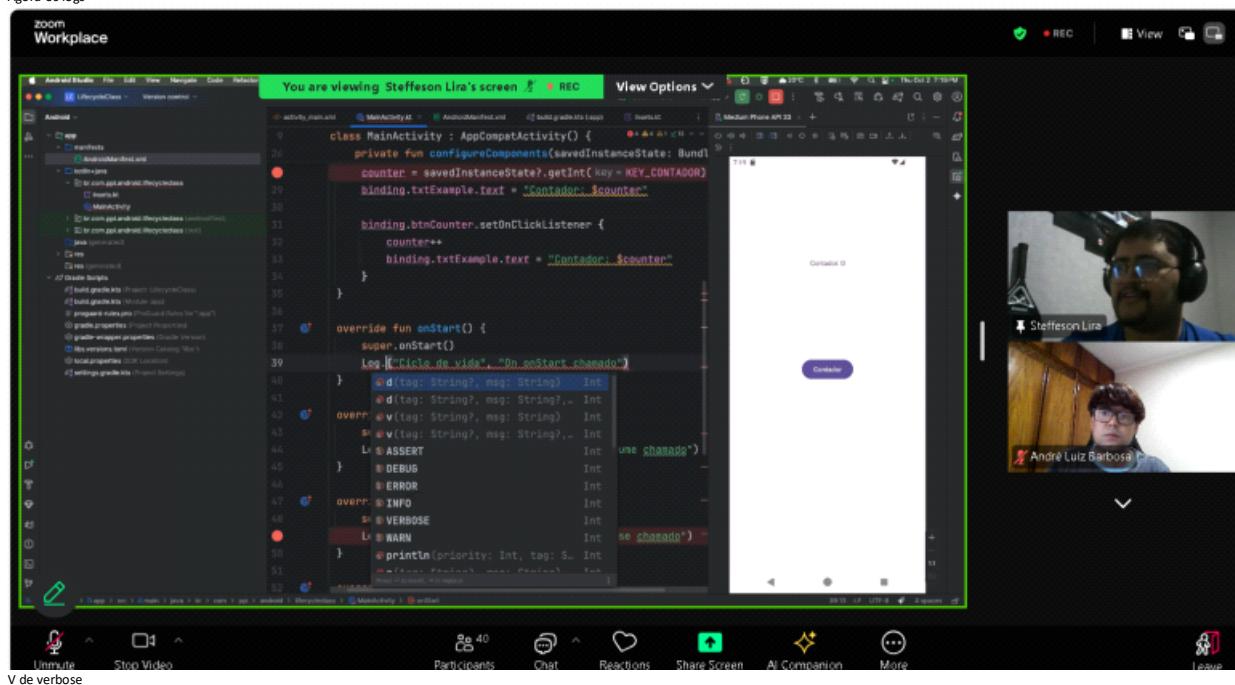
Se você estiver usando múltiplas telas ou precisa que uma delas se adapte dinamicamente, talvez seja melhor deixar como unspecified e controlar via código. Me avisa se quiser que a orientação mude com base em alguma lógica — dá pra fazer isso também!

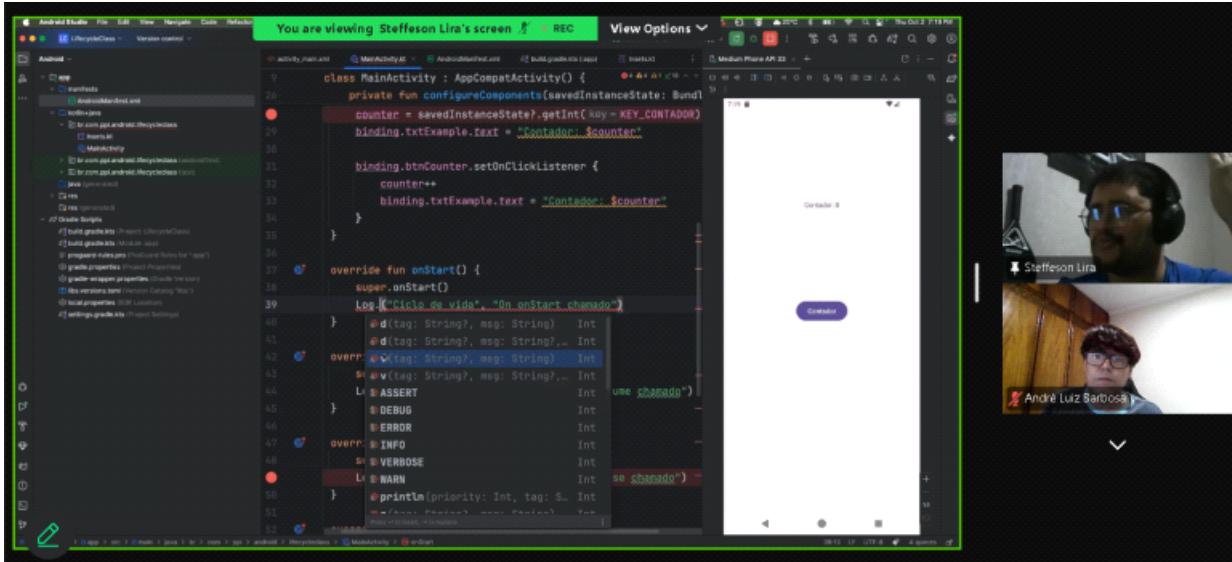
Re <<https://copilot.microsoft.com/chats/e19vsncn99fqSIScgeZsZ>>

Use a sensor portrait you saved???



## Agora os logs



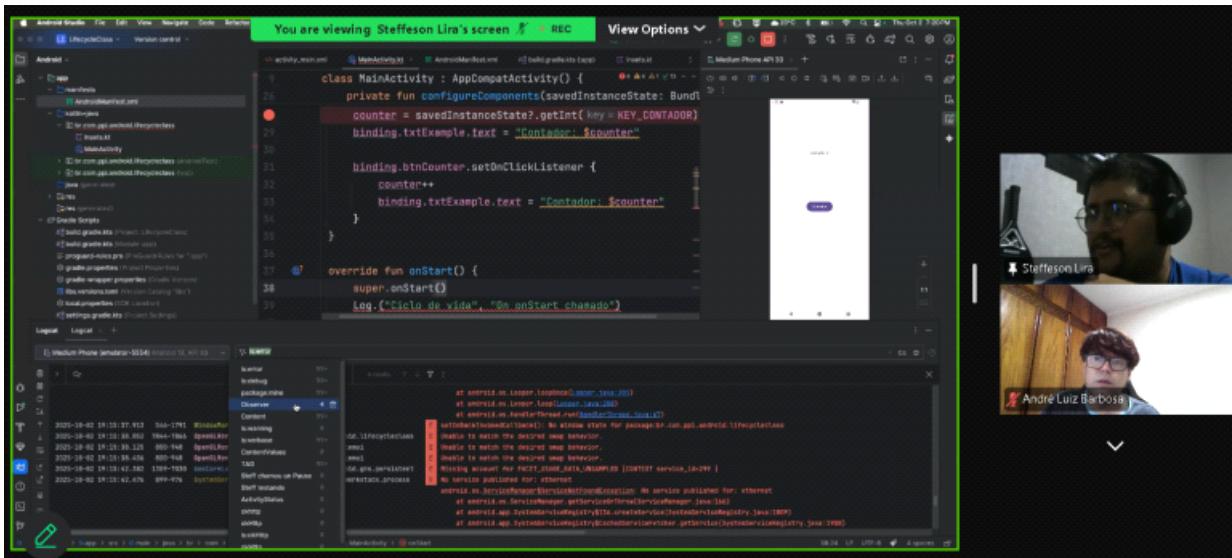
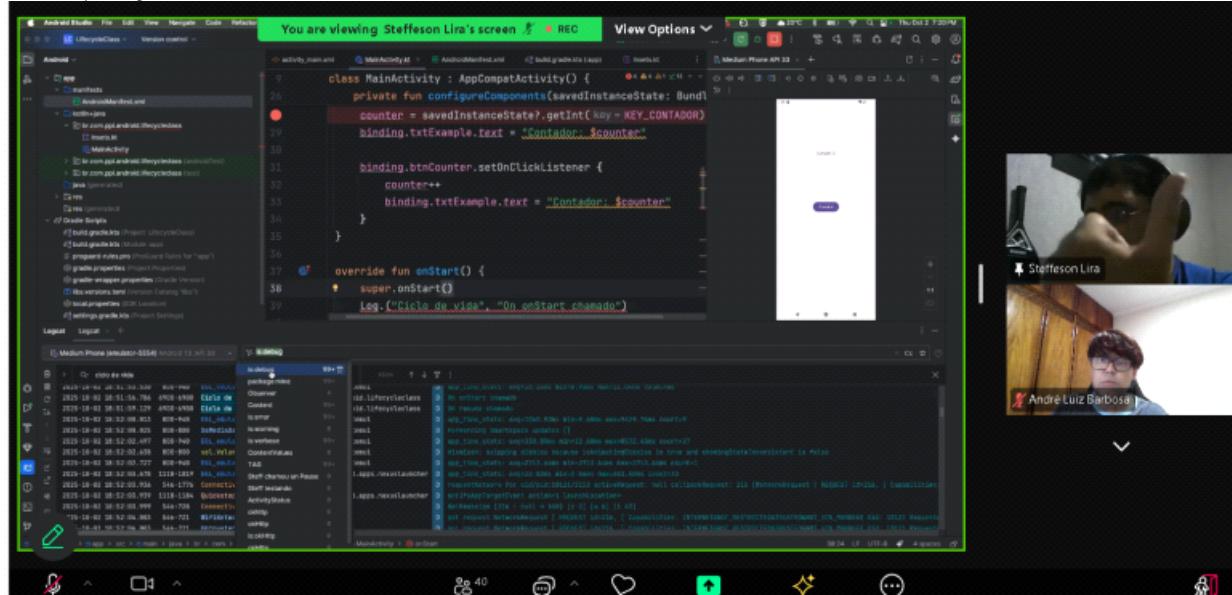


Geralmente coloca o D

Tem o E de erro

De info, informativo passando no device, sistema, de worming tambem

Pode filtrar por debug, error,



You are viewing Steffeson Lira's screen REC View Options

## Registro de logs no Android

- Monitore o fluxo de eventos ou o estado do seu app.
- Use a classe Log embutida ou uma biblioteca de terceiros.
- Exemplo de chamada ao método Log : Log.d(TAG, "Message")

Logical  
Emulator Pixel\_2\_Oreo\_-\_API\_26 - com.example.myapplication (0) • Verbose • MainActivity  
I: Input  
# 2020-05-28 15:19:42.189 5625-5625/com.example.myapplication I/zygote: at void com.example.myapplication.MainActivity.onCreate(android.os.Bundle) (MainActivity.java:8)  
# 2020-05-28 15:19:42.209 5625-5625/com.example.myapplication I/MainActivity: onStart  
# 2020-05-28 15:19:42.295 5625-5625/com.example.myapplication I/MainActivity: onResume  
# 2020-05-28 15:19:52.837 5625-5625/com.example.myapplication I/MainActivity: onPause

**venturus**

You are viewing Steffeson Lira's screen REC View Options

## Escrita de logs

Nível de prioridade	Método do Log
Verbose	Log.v(String, String)
Debug	Log.d(String, String)
Info	Log.i(String, String)
Warning	Log.w(String, String)
Error	Log.e(String, String)

**venturus**

Ciclo de vida do fragment

You are viewing Steffeson Lira's screen REC View Options

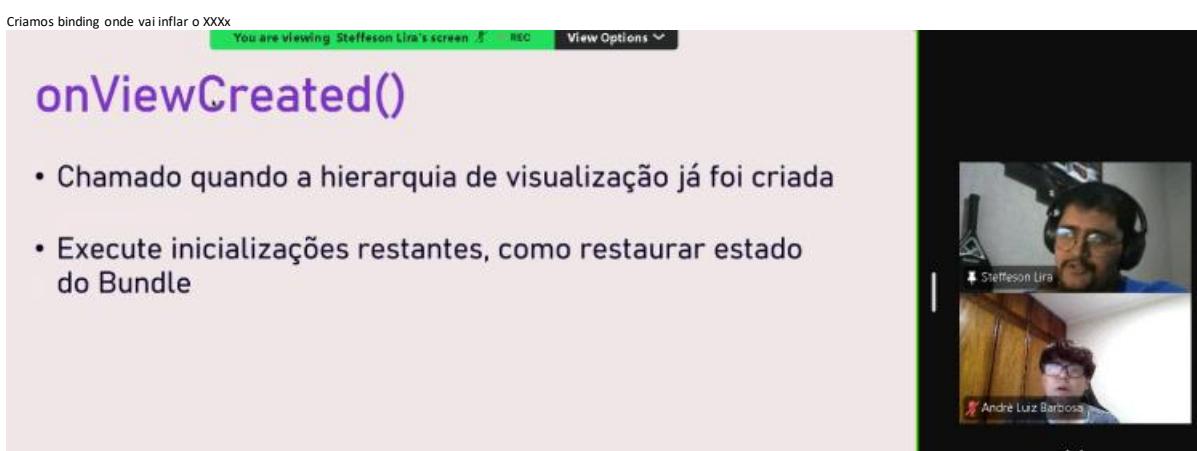
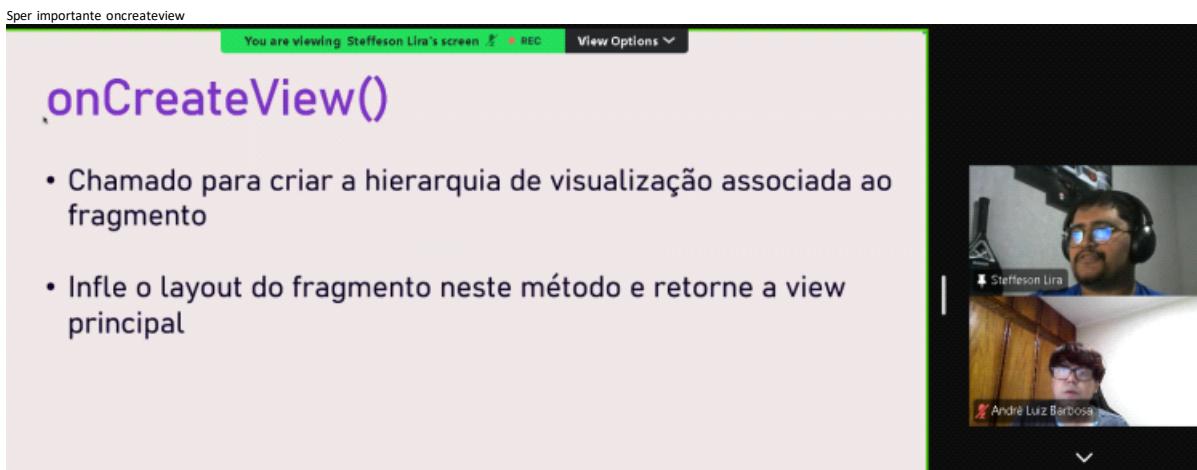
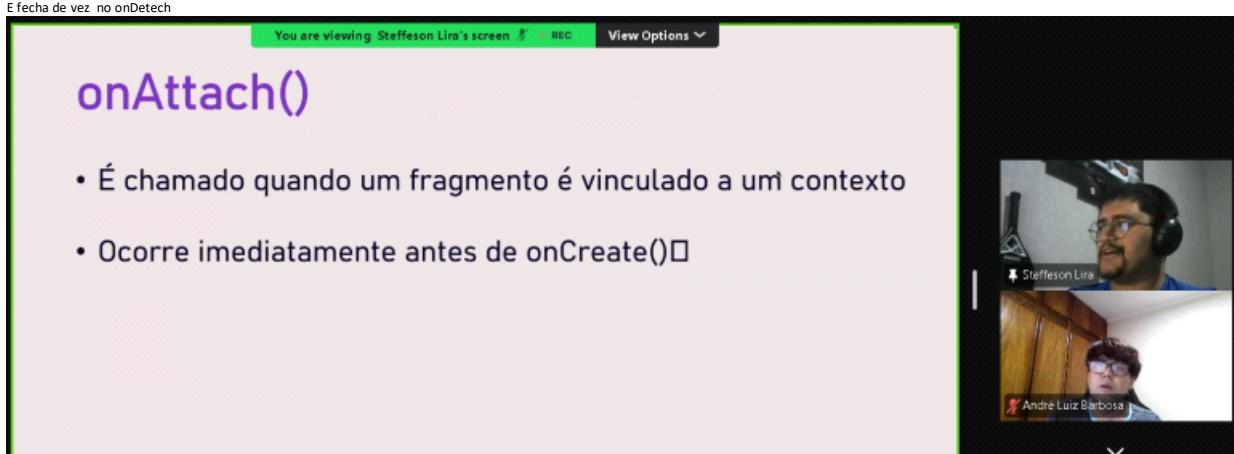
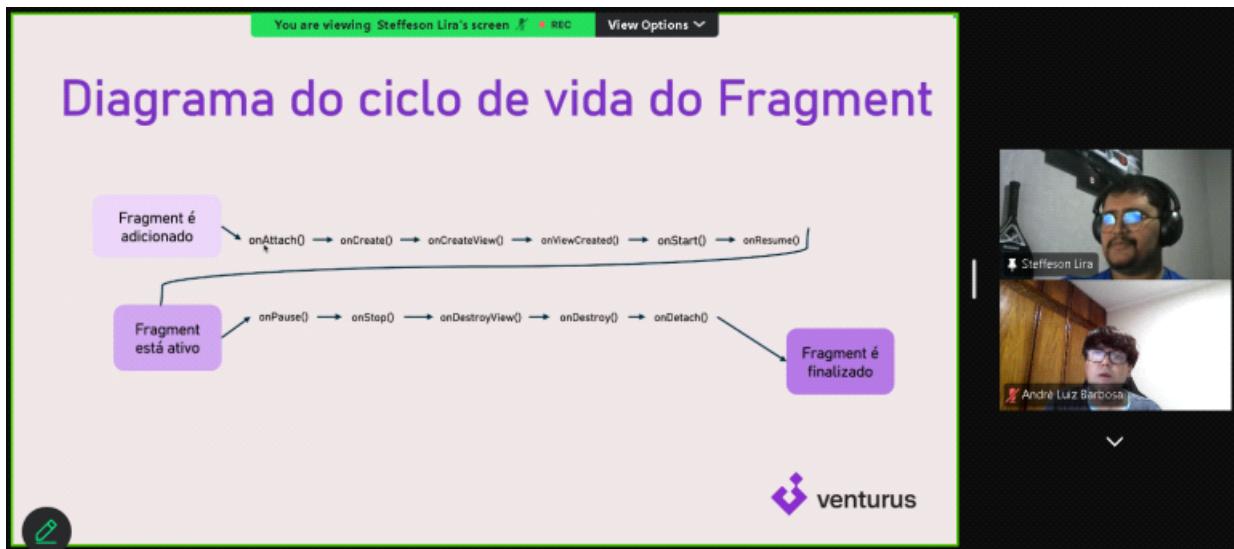
## Fragment states

```

graph TD
    CREATED[CREATED] --> STARTED[STARTED]
    STARTED --> RESUMED[RESUMED]
    RESUMED --> Running[Fragment is running]
    Running --> PAUSED[PAUSED]
    PAUSED --> STOPPED[STOPPED]
    STOPPED --> DESTROYED[DESTROYED]
  
```

**venturus**

Vai mudar um pouco do que esta no slide  
 Single activity  
 Que vai ser am mae das outras telas  
 Fragment é bem parecido com activity, tem os status dele  
 Desde create ate destroyed  
 O ciclo de vida dele é um pouco maior



You are viewing Steffeson Lira's screen REC View Options

# onDestroyView() e onDetach()

- onDestroyView()** É chamado quando a hierarquia de visualização do fragmento é removida.
- onDetach()** é chamado quando o fragmento não está mais anexado ao host.

You are viewing Steffeson Lira's screen REC View Options

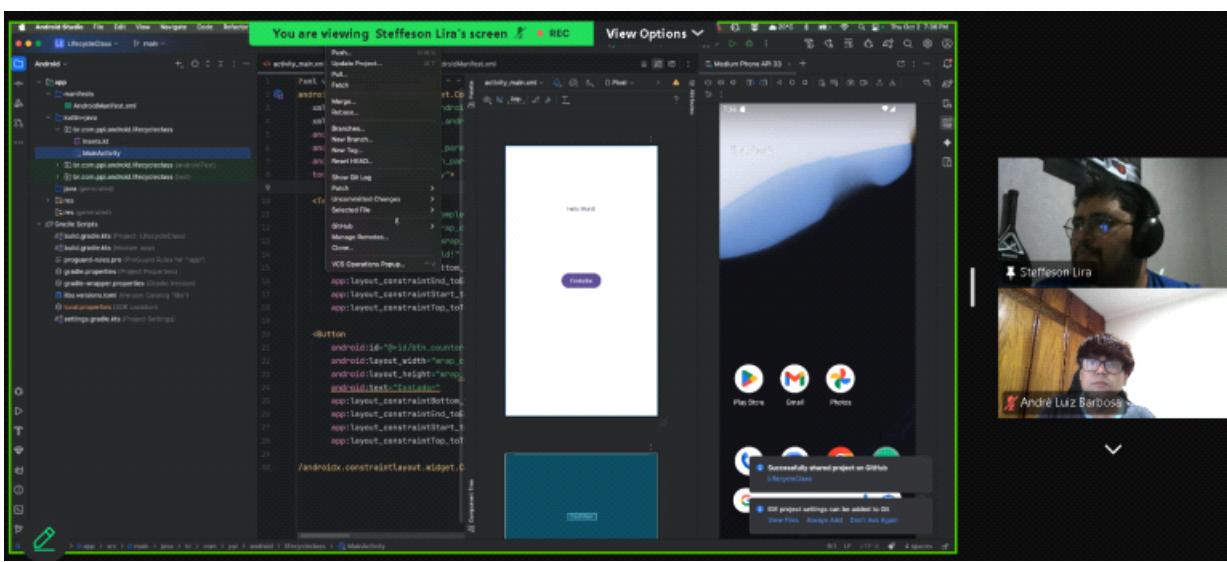
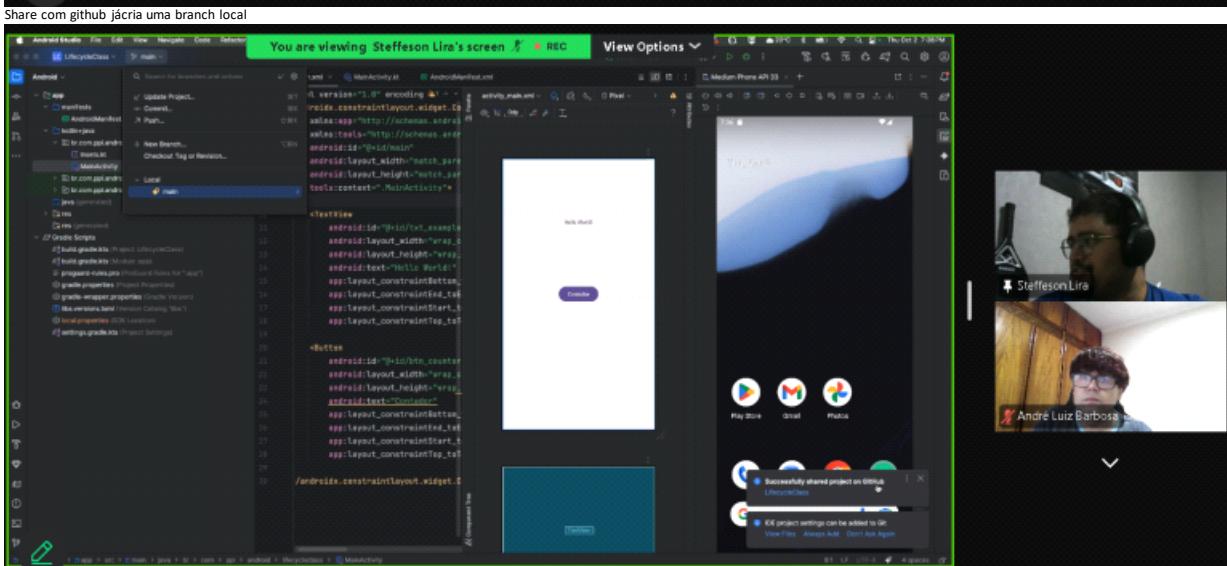
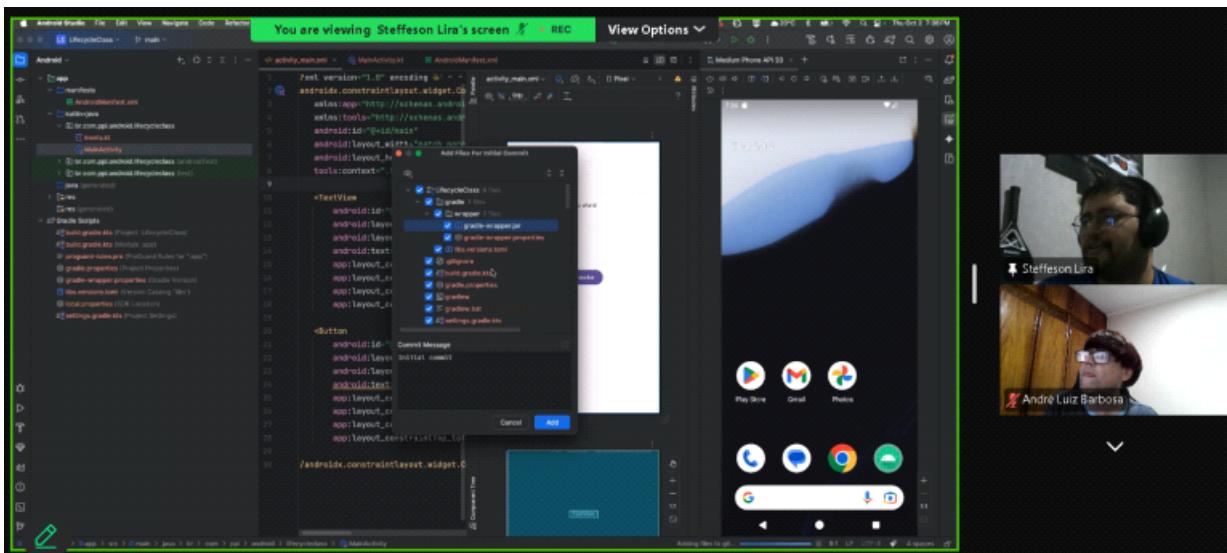
## Resumo dos estados do Fragment

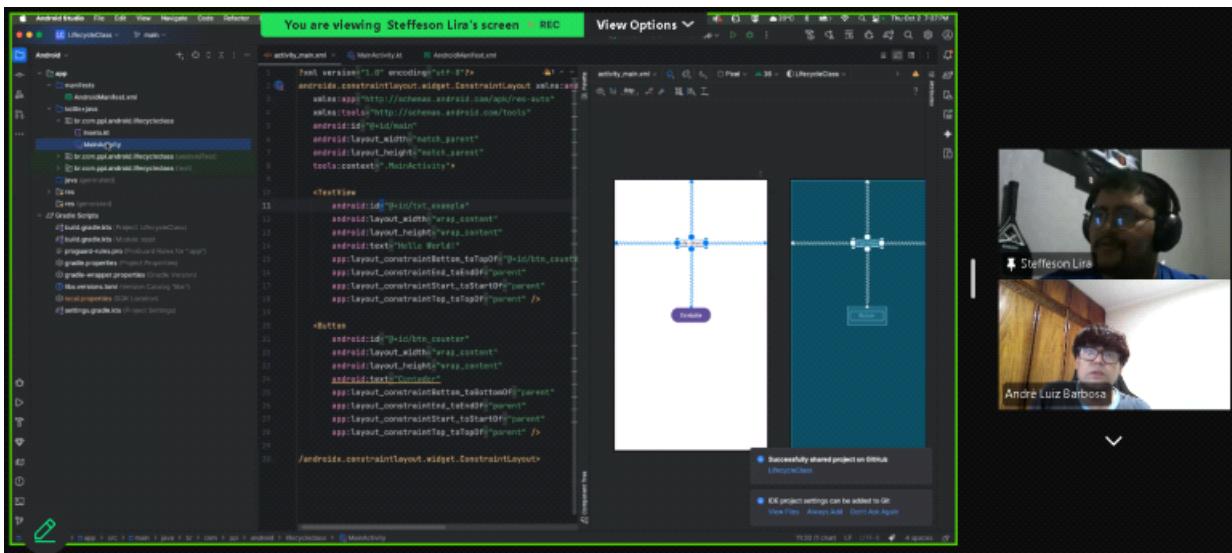
Estado	Callbacks	Description
Inicializado	onAttach()	O fragmento está anexado ao host.
Criado	onCreate(), onCreateView(), onViewCreated()	O fragmento foi criado e o layout está sendo inicializado.
Iniciado	onStart()	O fragmento foi iniciado e está visível.
Ativo	onResume()	O fragmento está com foco de entrada.
Pausado	onPause()	O fragmento não tem mais foco de entrada.
Parado	onStop()	O fragmento não está visível.
Finalizado	onDestroyView(), onDestroy(), onDetach()	O fragmento foi removido do host.

venturus

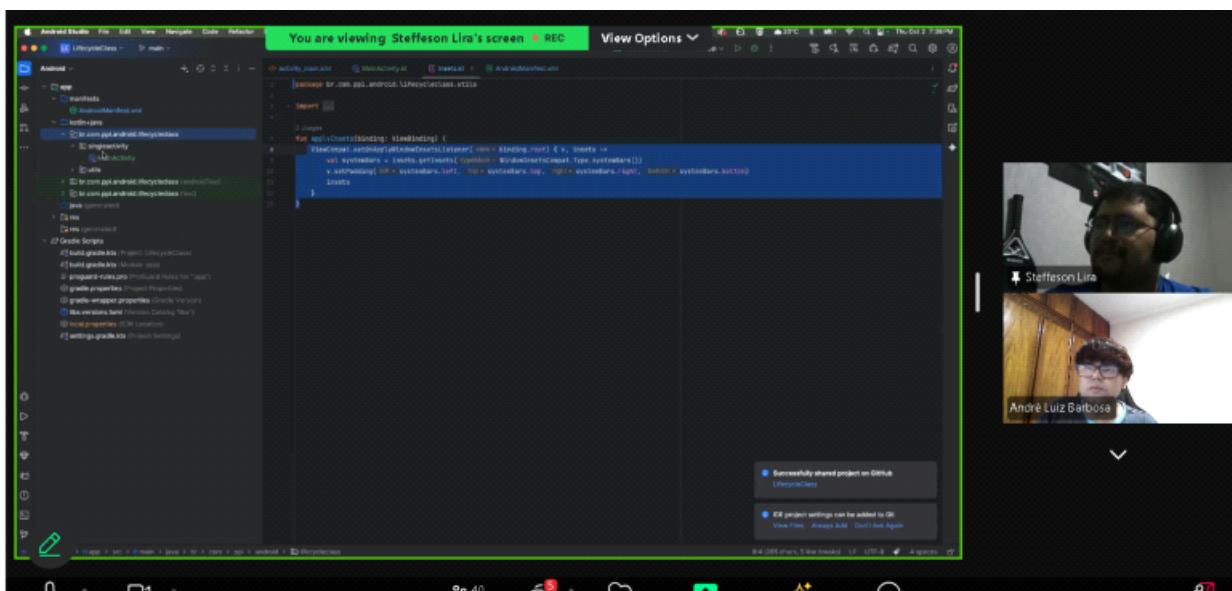
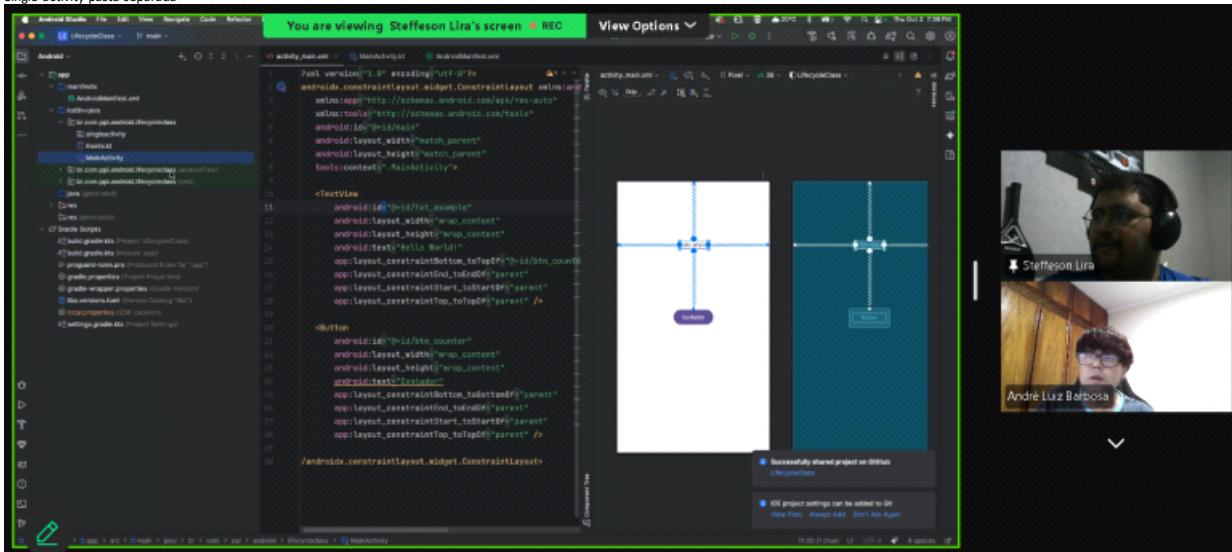
You are viewing Steffeson Lira's screen REC View Options

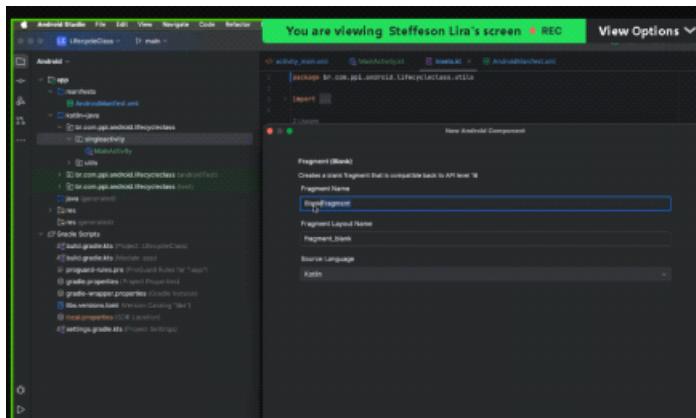
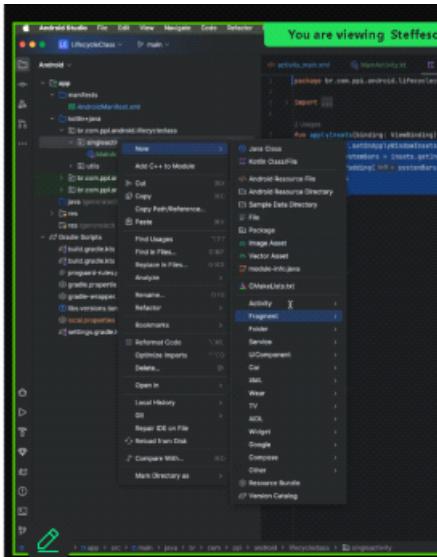
Vai dicionar no repos



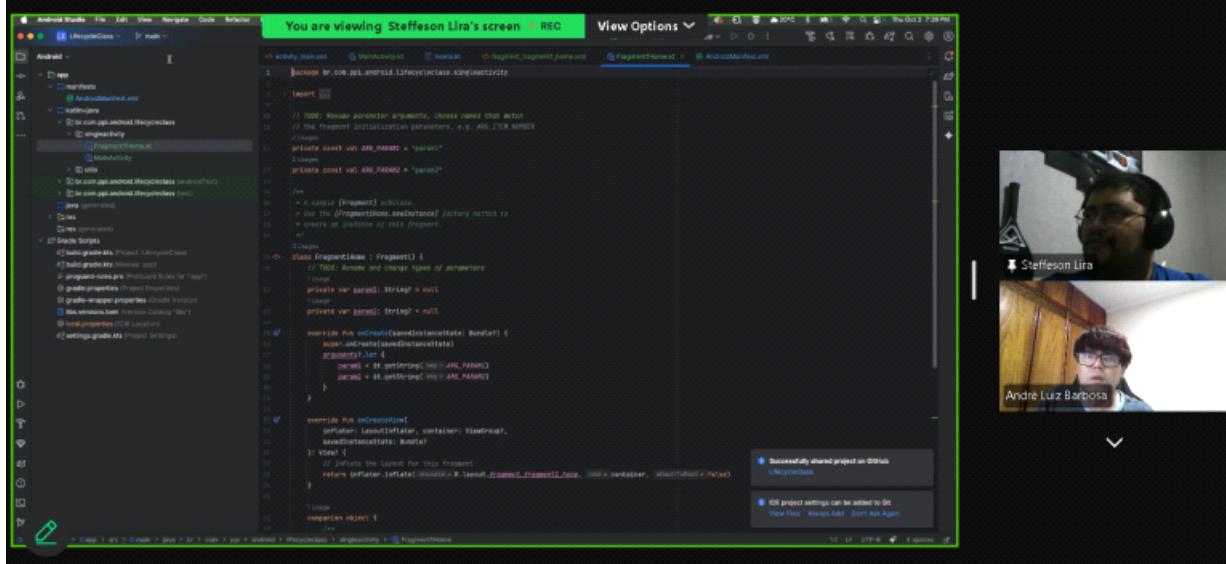


Single activity pasta separada

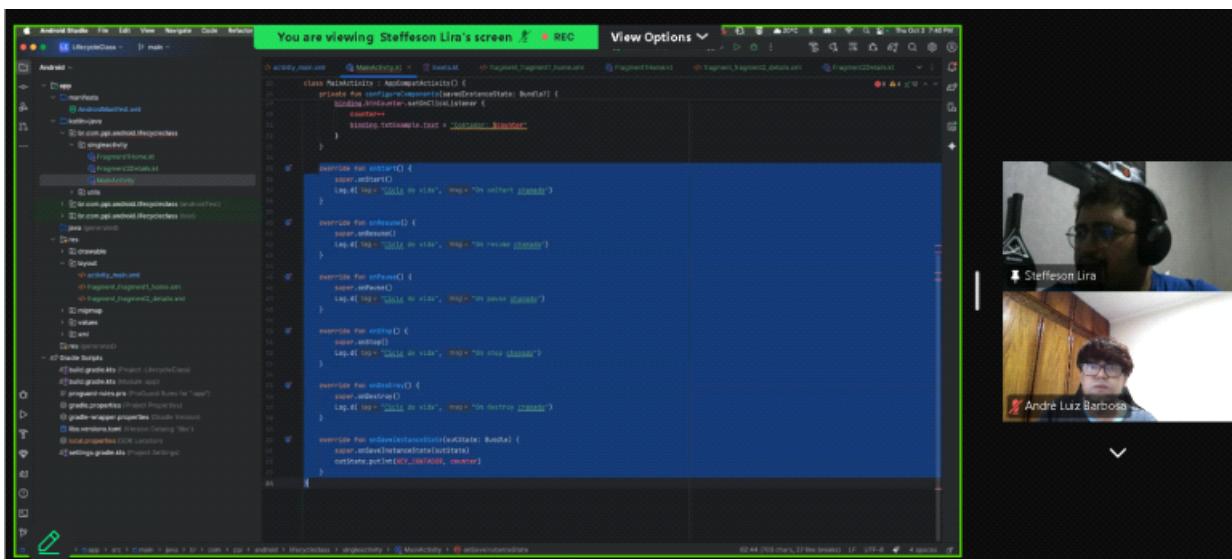
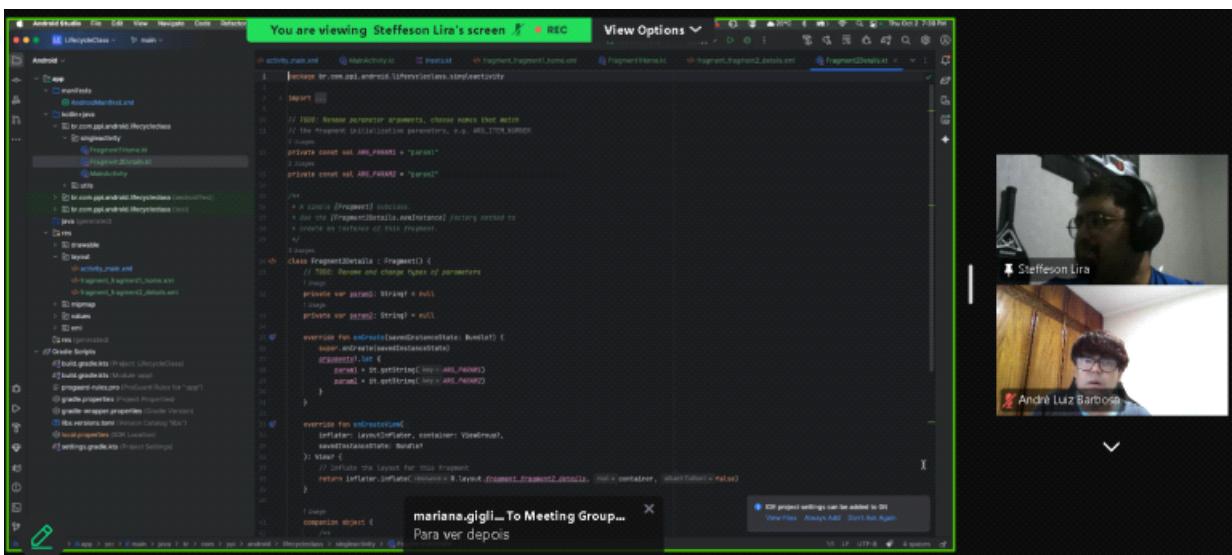
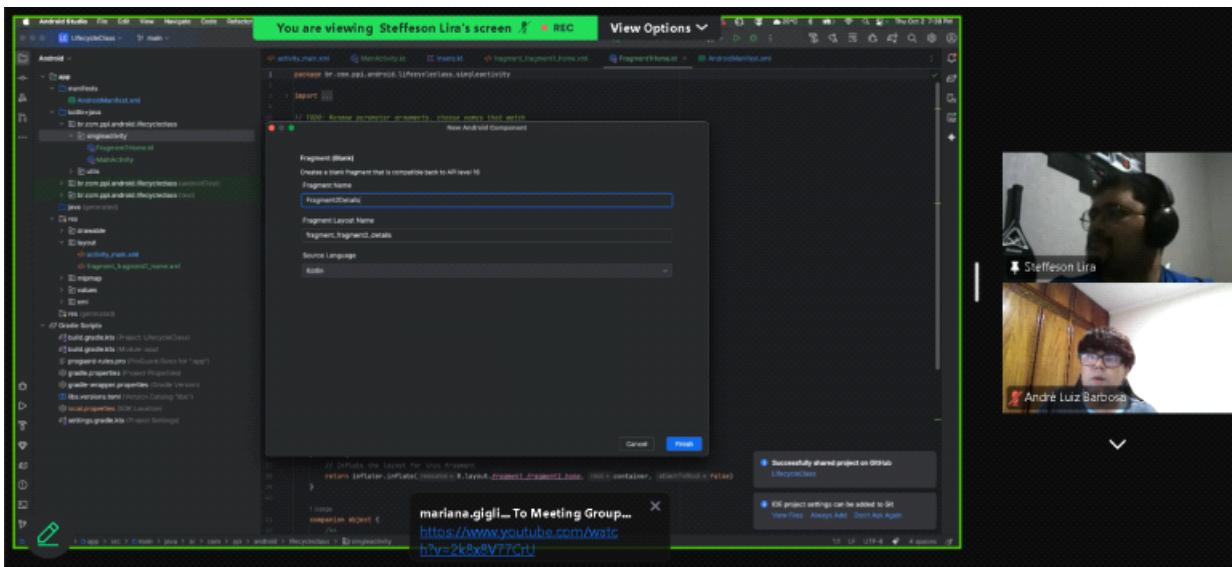




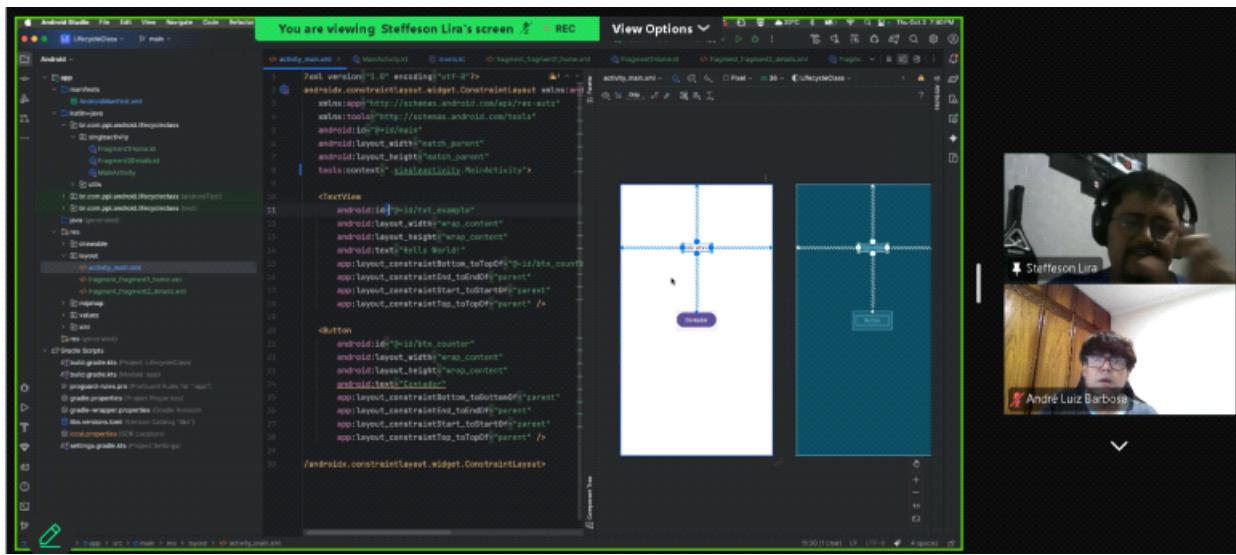
Já add no git



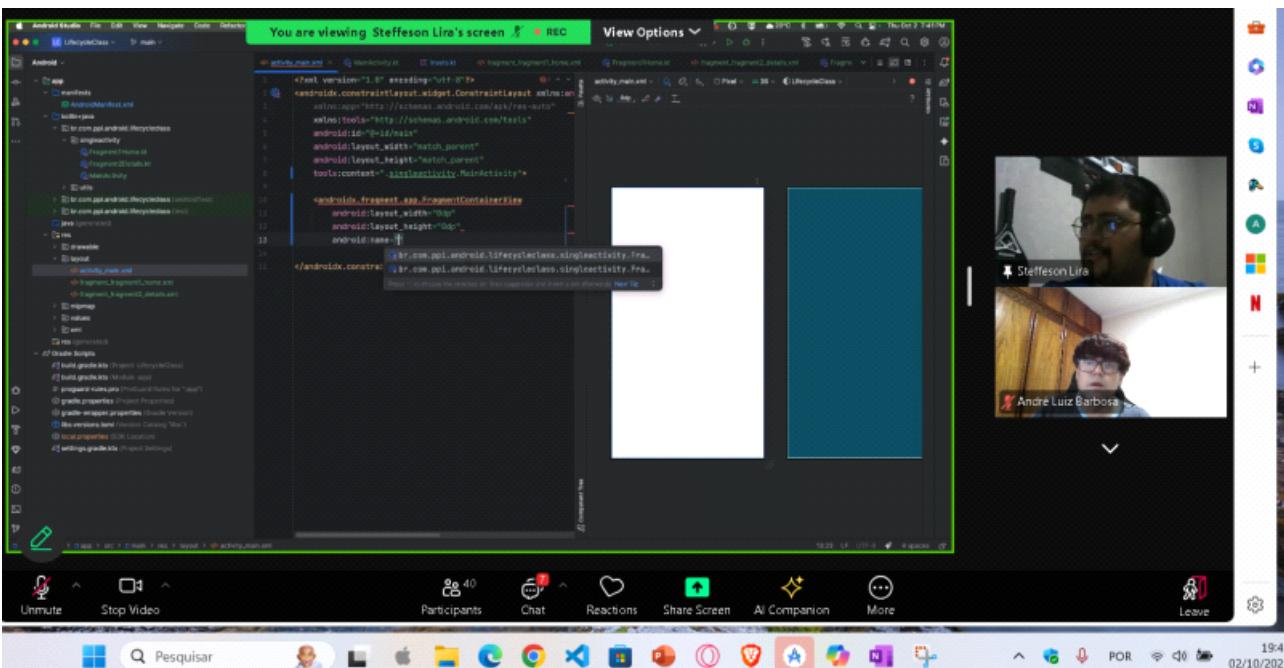
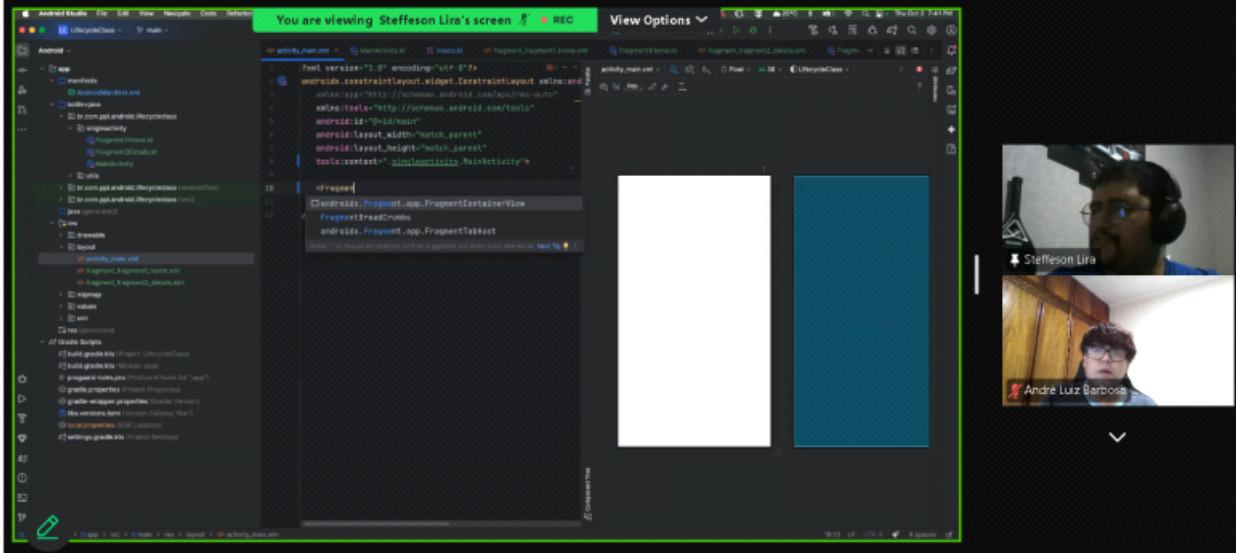
Já cira a classe e layout



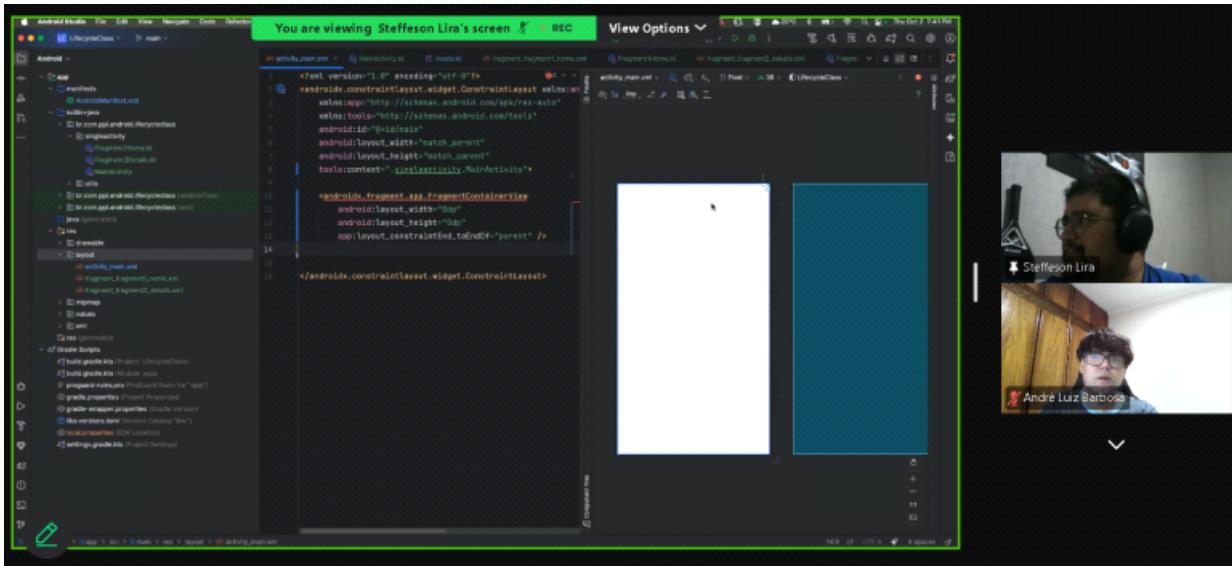
Vai tirar várias partes, deixar praticamente o que é vêm do padrão Container



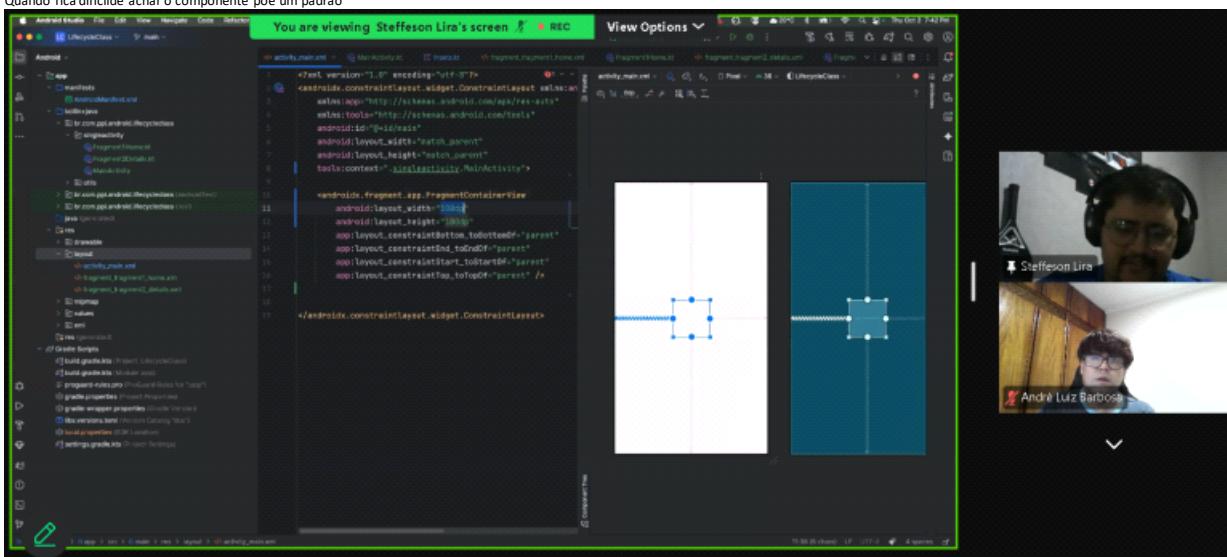
Tirou vários e vaipor container



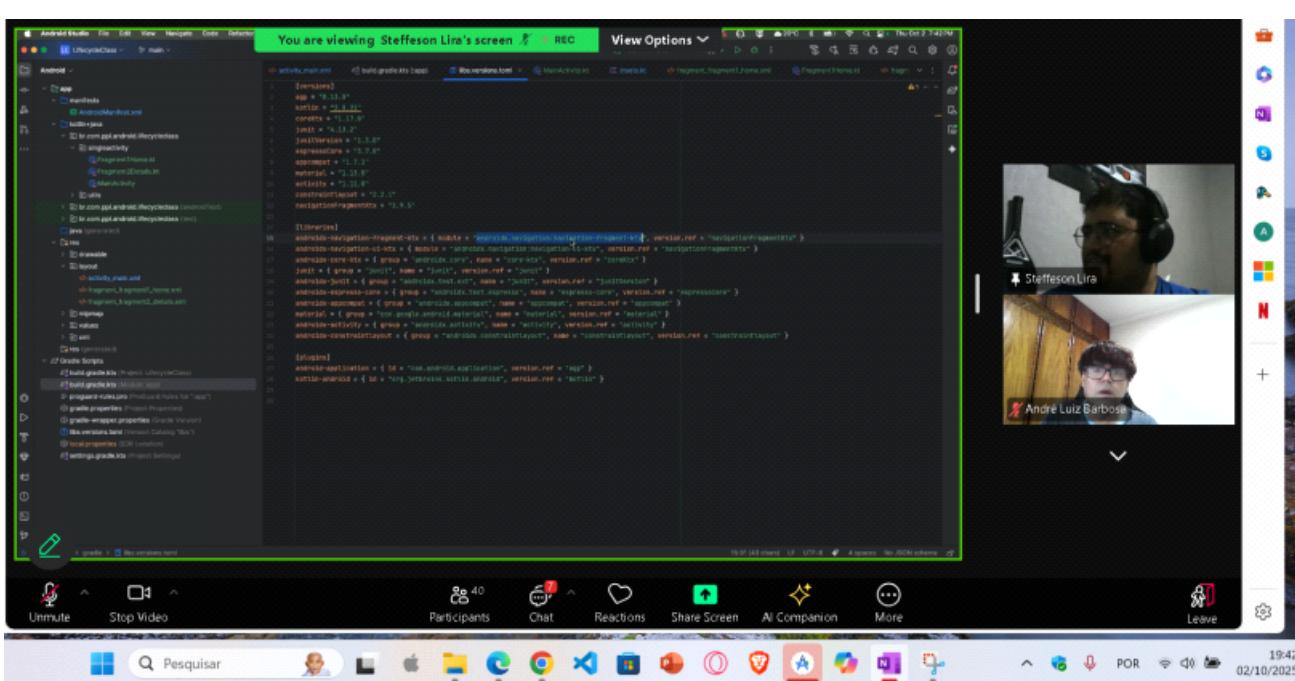
Vai colocar os alinhamentos na tela

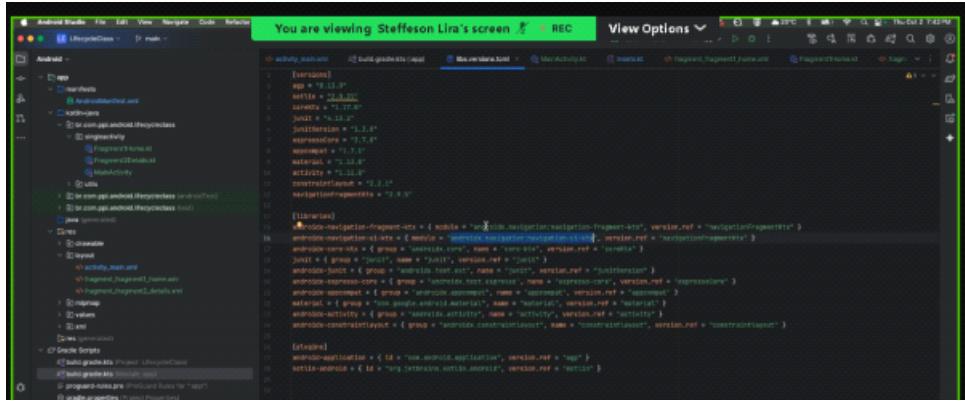


Quando fica dificil de achar o componente põe um padrao

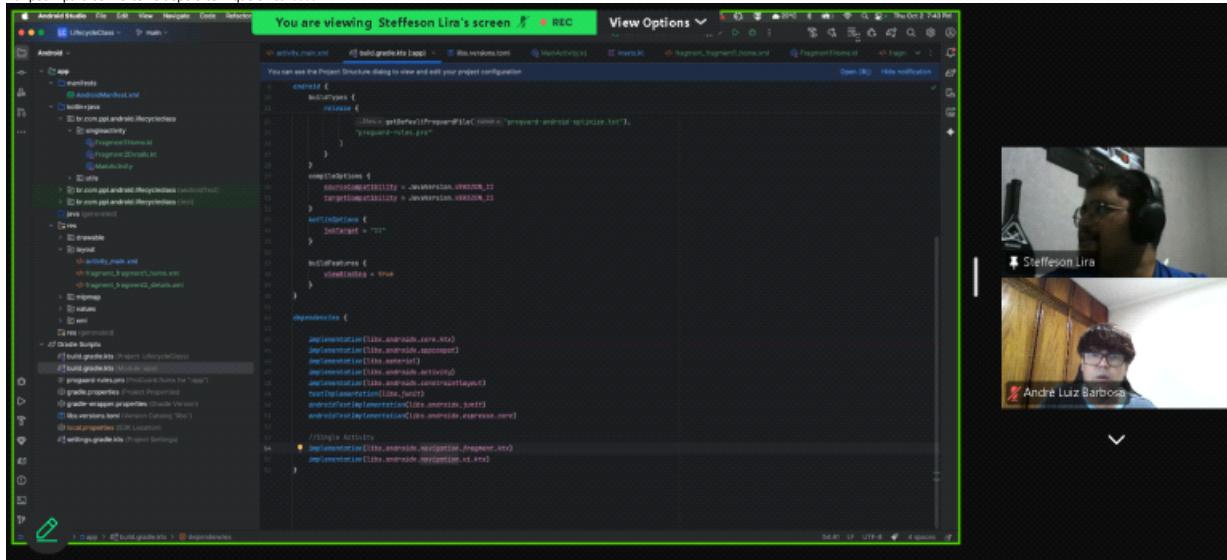


Precisa de um navGraph  
Antes vai no gradle  
Vai importar duas libs

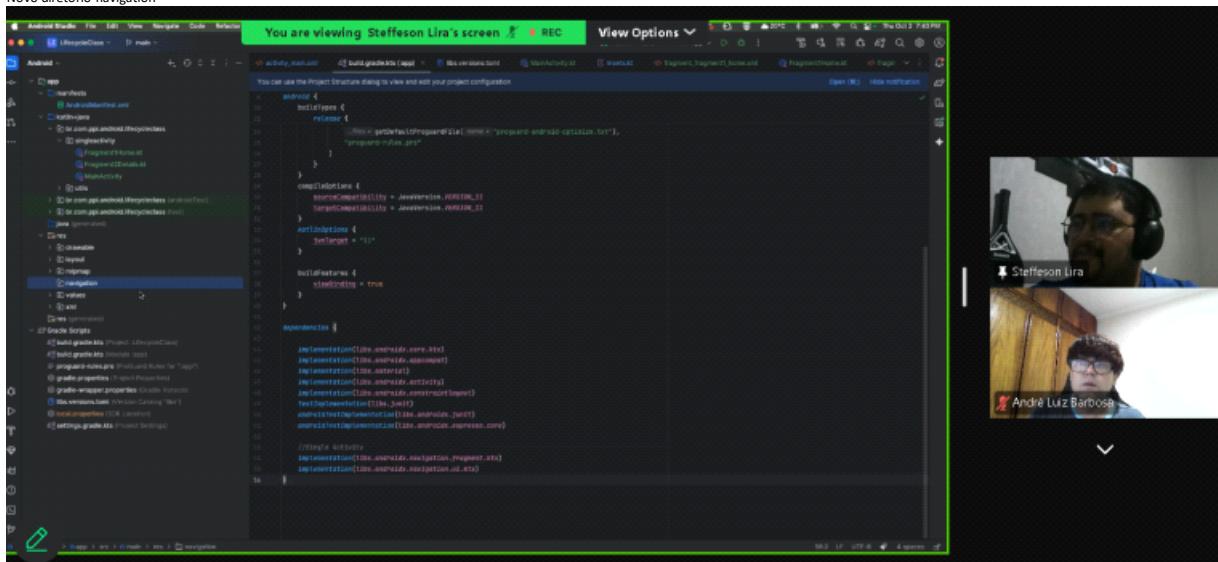




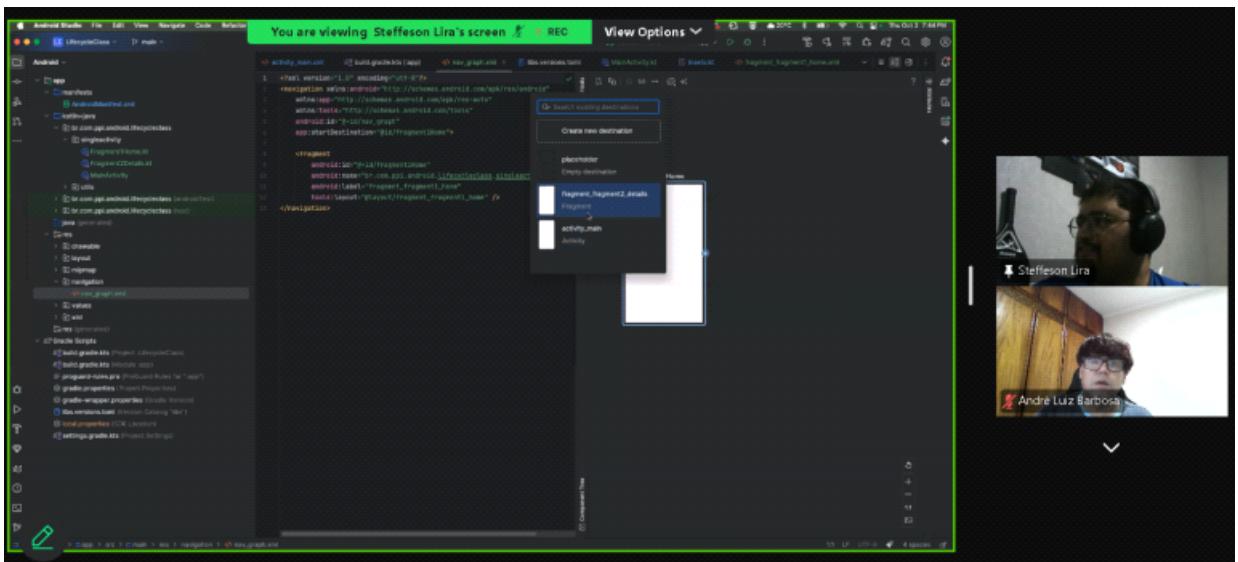
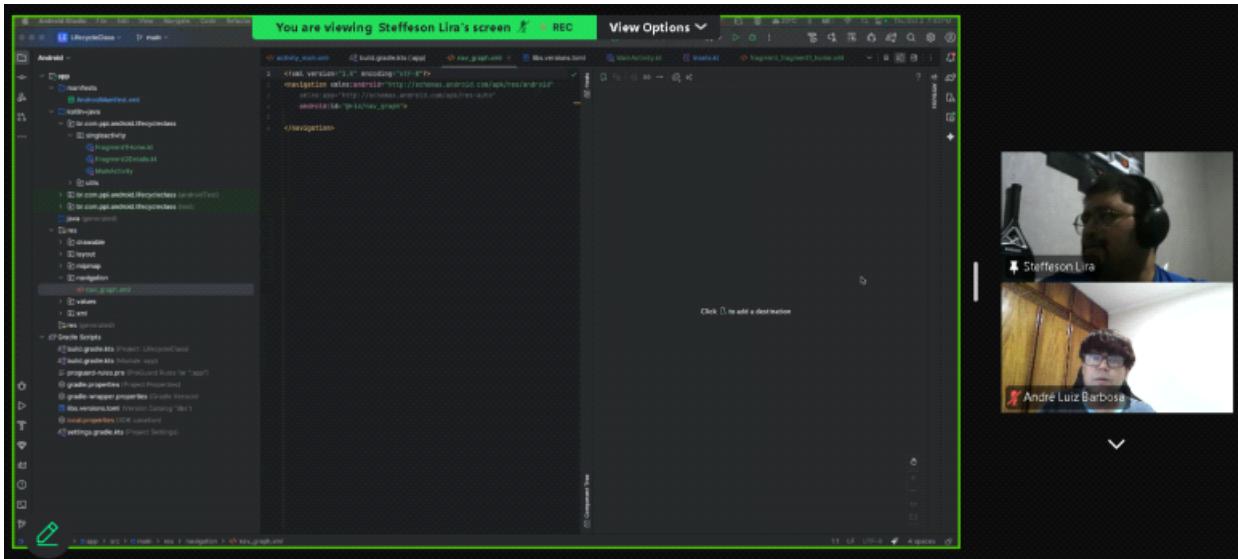
Vai pedir para converter e depois tem que sincar????



**Novo diretório navigation**

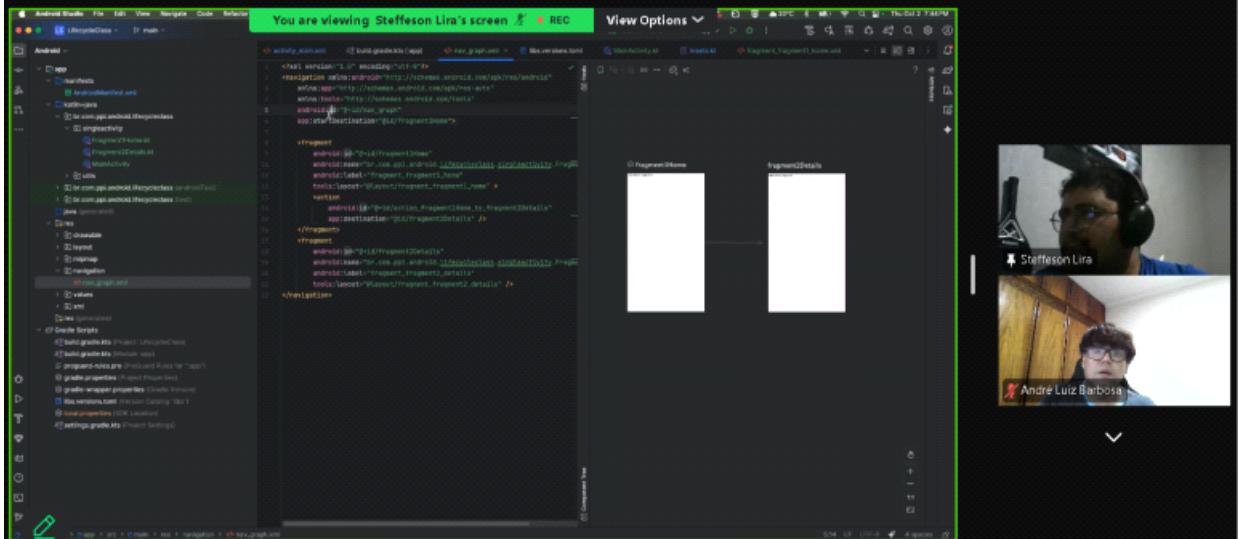


criou o navgraph e vai colocando os fragments

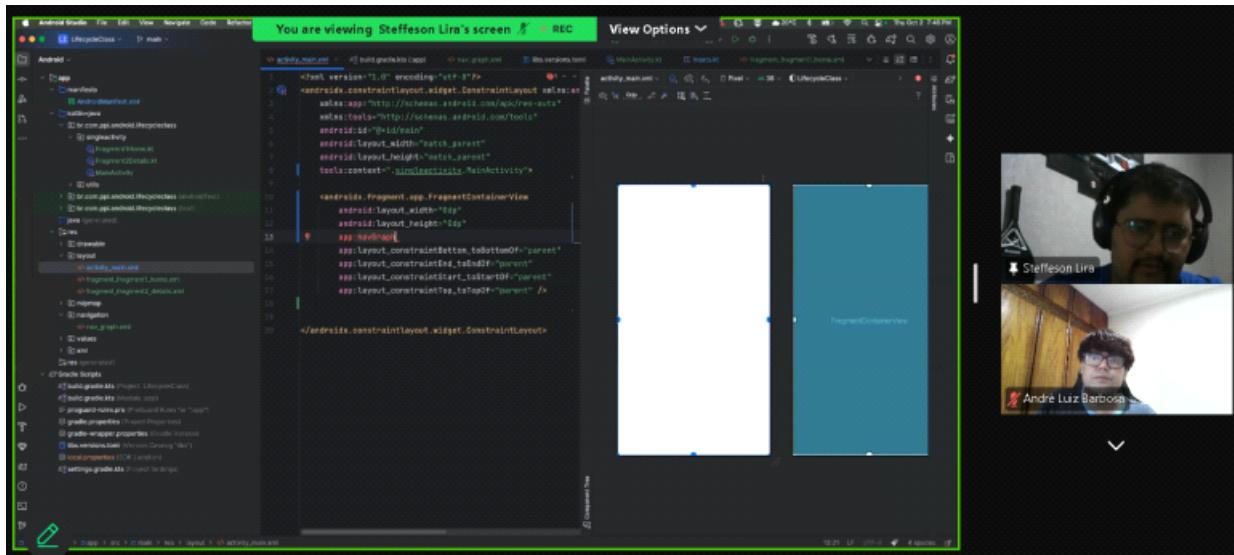
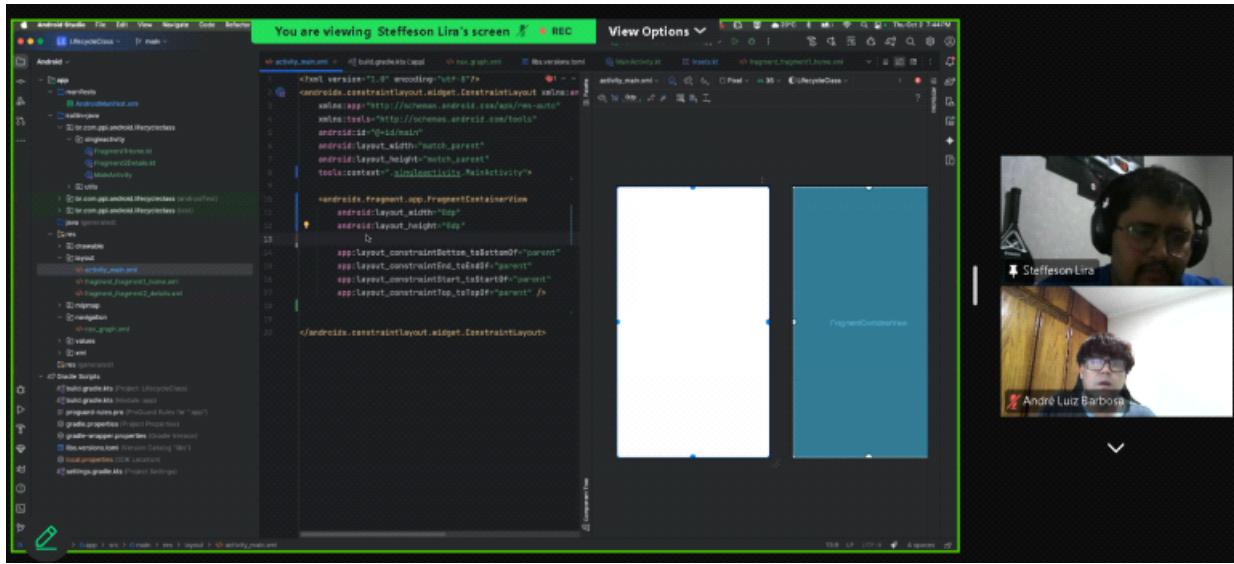


Activity vai se a mãe e por enquanto aqui não vai aparecer a activity

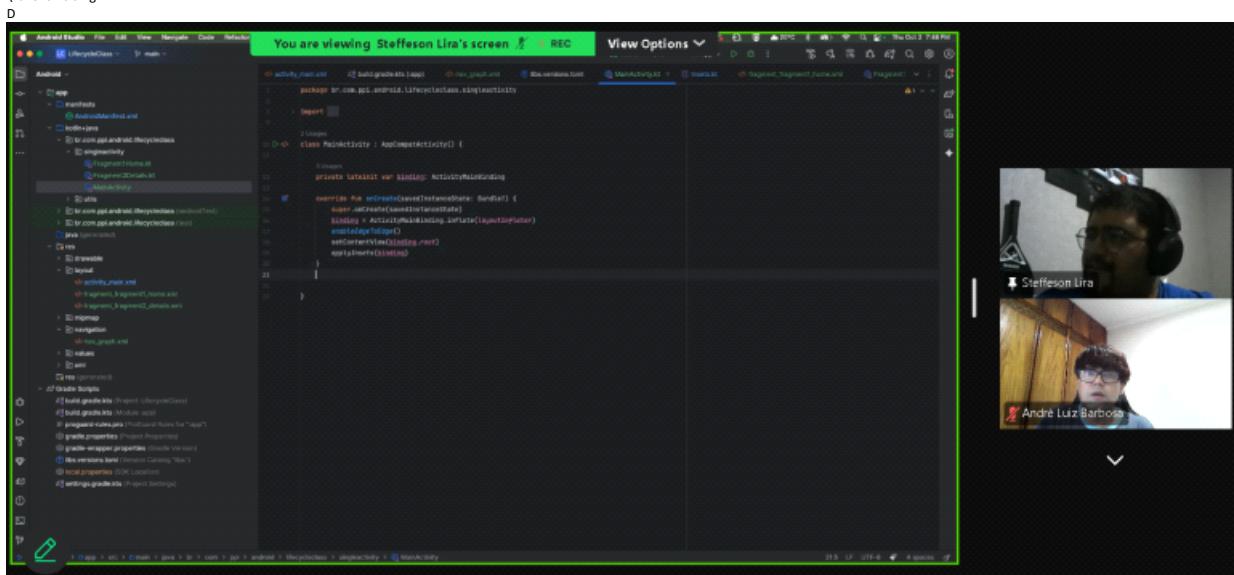
Vai colocar a seta para fazer navegação futura



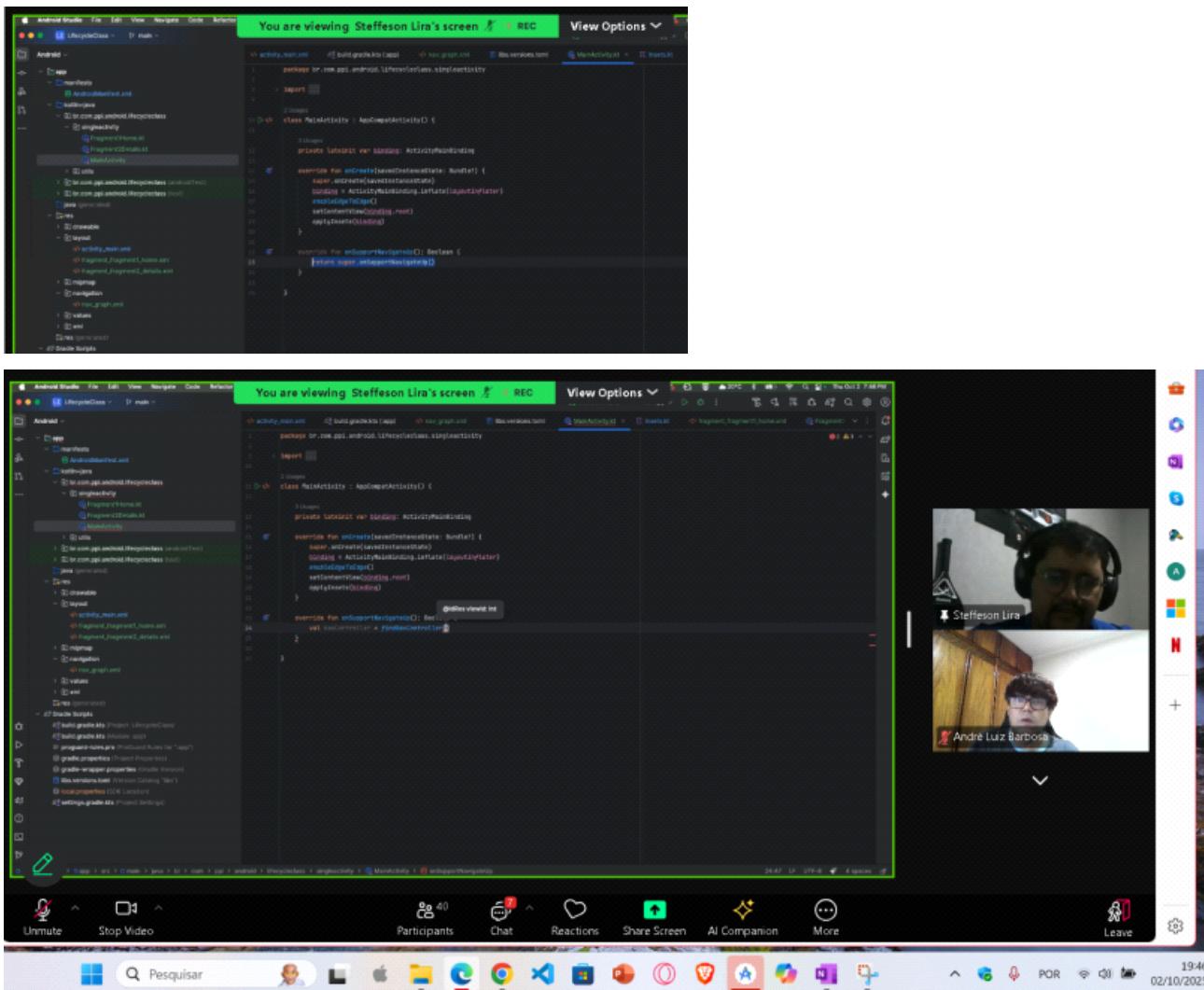
Voltando para o layout



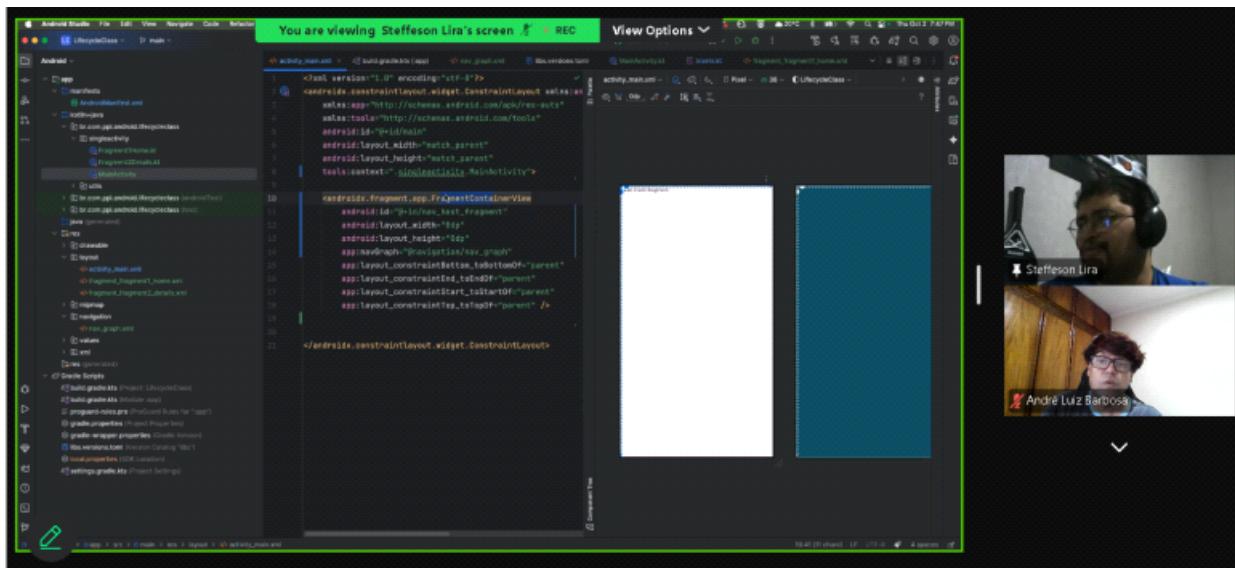
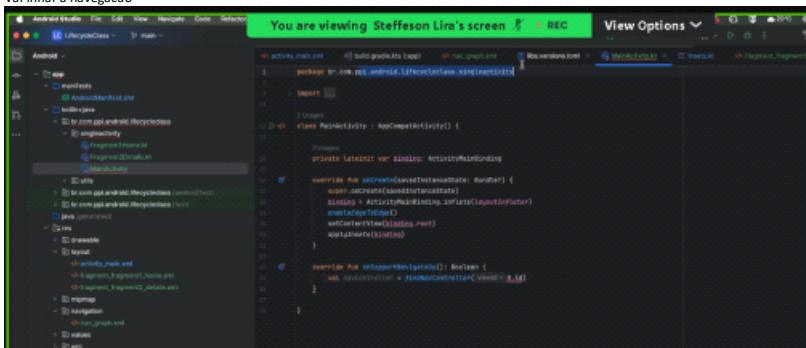
Na activity  
\ vai criar biding



Overriding



Vai inflar a navegacao



You are viewing Steffeson Lira's screen REC View Options

```

Android - LifecycleClass - main - You are viewing Steffeson Lira's screen REC View Options
src/main/java/com/app/android/lifecycletest/singleactivity
private lateinit var binding: ActivityMainBinding
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)
}
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }
}

```

You are viewing Steffeson Lira's screen REC View Options

```

Android - LifecycleClass - main - You are viewing Steffeson Lira's screen REC View Options
src/main/java/com/app/android/lifecycletest/singleactivity
private lateinit var binding: ActivityMainBinding
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)
}
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }
}

```

Com isto faz a criação da activity com os fragmentos

Executando

Iniciou a aplicação mas não tem nada ainda

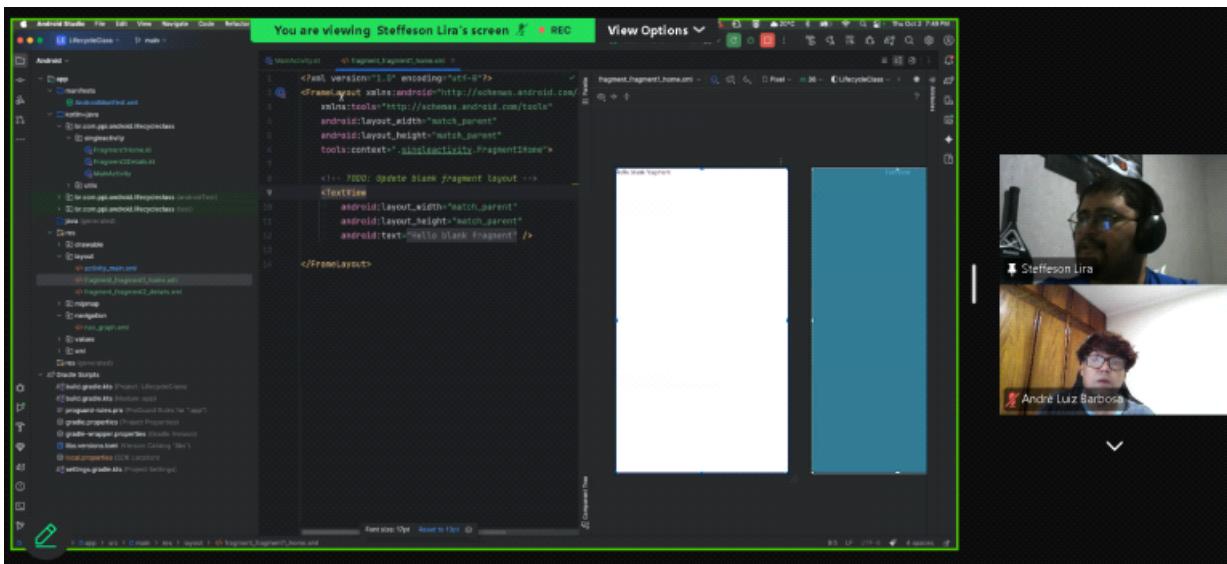
You are viewing Steffeson Lira's screen REC View Options

```

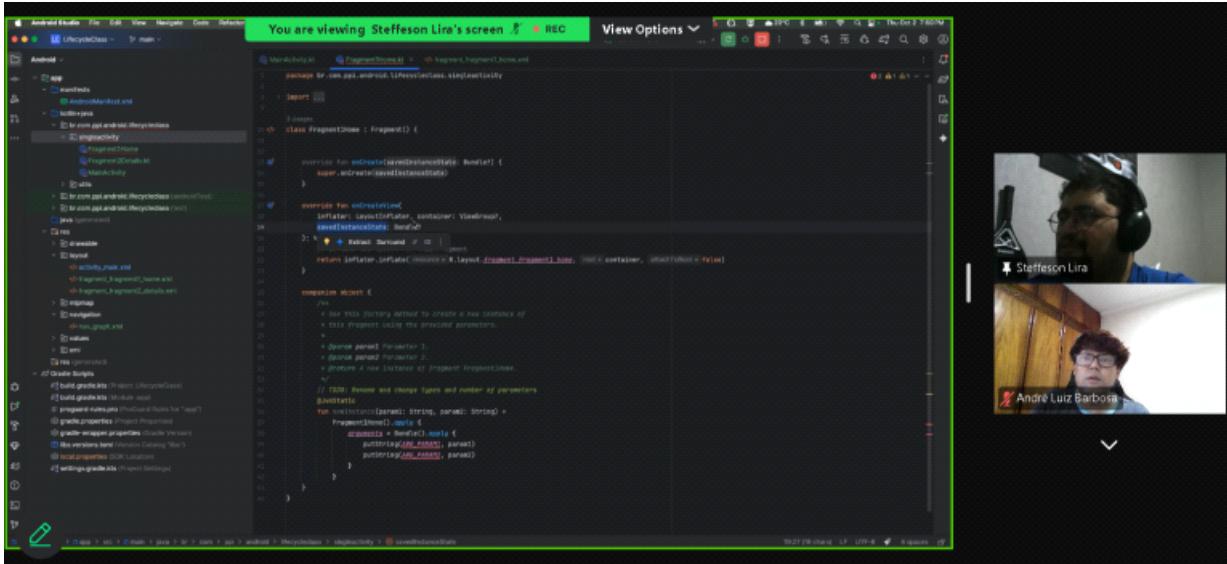
Android - LifecycleClass - main - You are viewing Steffeson Lira's screen REC View Options
src/main/java/com/app/android/lifecycletest/singleactivity
private lateinit var binding: ActivityMainBinding
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)
}
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
    }
}

```

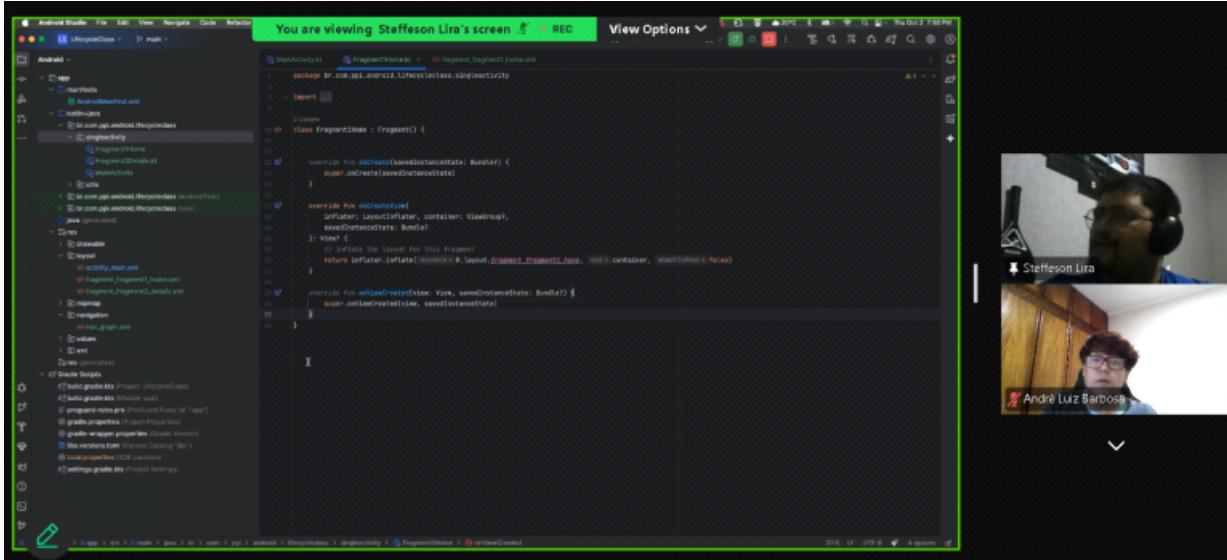
Quando cria vem com framelayoute mas ele só recomenda o constraint layout tanto no fragment quanto na activity



Está tirando vários



E add outros



Tá usnado esta ordem

You are viewing Steffeson Lira's screen  REC View Options

# Resumo dos estados do Fragment

Estado	Callbacks	Description
Inicializado	onAttach()	O fragmento está anexado ao host.
Criado	onCreate(), onCreateView(), onViewCreated()	O fragmento foi criado e o layout está sendo inicializado.
Iniciado	onStart()	O fragmento foi iniciado e está visível.
Ativo	onResume()	O fragmento está com foco de entrada.
Pausado	onPause()	O fragmento não tem mais foco de entrada.
Parado	onStop()	O fragmento não está visível.
Finalizado	onDestroyView(), onDestroy(), onDetach()	O fragmento foi removido do host.



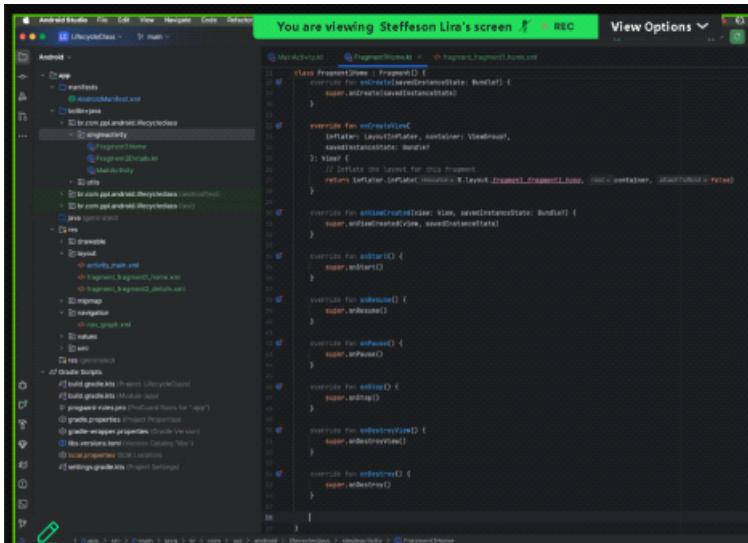
Steffeson Lira



André Luiz Barbosa

 venturus

```
public class MainActivity extends AppCompatActivity implements FragmentListener {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
  
    @Override  
    public void onActivityCreated(@NonNull Fragment fragment, @Nullable Bundle savedInstanceState) {  
        if (fragment instanceof HomeFragment) {  
            HomeFragment homeFragment = (HomeFragment) fragment;  
            homeFragment.setListener(this);  
        }  
    }  
  
    @Override  
    public void onAttach(Context context) {  
        super.onAttach(context);  
    }  
  
    @Override  
    public void onDetach() {  
        super.onDetach();  
    }  
  
    @Override  
    public void onCreateView(LayoutInflater inflater, ViewGroup container, @Nullable Bundle savedInstanceState) {  
        View view = inflater.inflate(R.layout.fragment_home, container, false);  
        ButterKnife.bind(this, view);  
        return view;  
    }  
}
```



```
class FragmentClass : Fragment() {
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_fragment, container, false)
    }

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

    override fun onStart() {
        super.onStart()
    }

    override fun onResume() {
        super.onResume()
    }

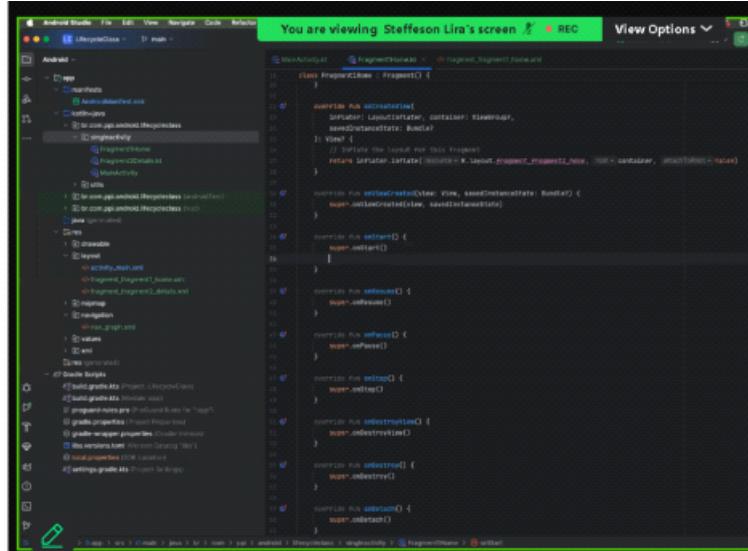
    override fun onPause() {
        super.onPause()
    }

    override fun onStop() {
        super.onStop()
    }

    override fun onDestroyView() {
        super.onDestroyView()
    }

    override fun onDestroy() {
        super.onDestroy()
    }
}
```

Por fim o onDestruct



```
class FragmentClass : Fragment() {
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_fragment, container, false)
    }

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

    override fun onStart() {
        super.onStart()
    }

    override fun onResume() {
        super.onResume()
    }

    override fun onPause() {
        super.onPause()
    }

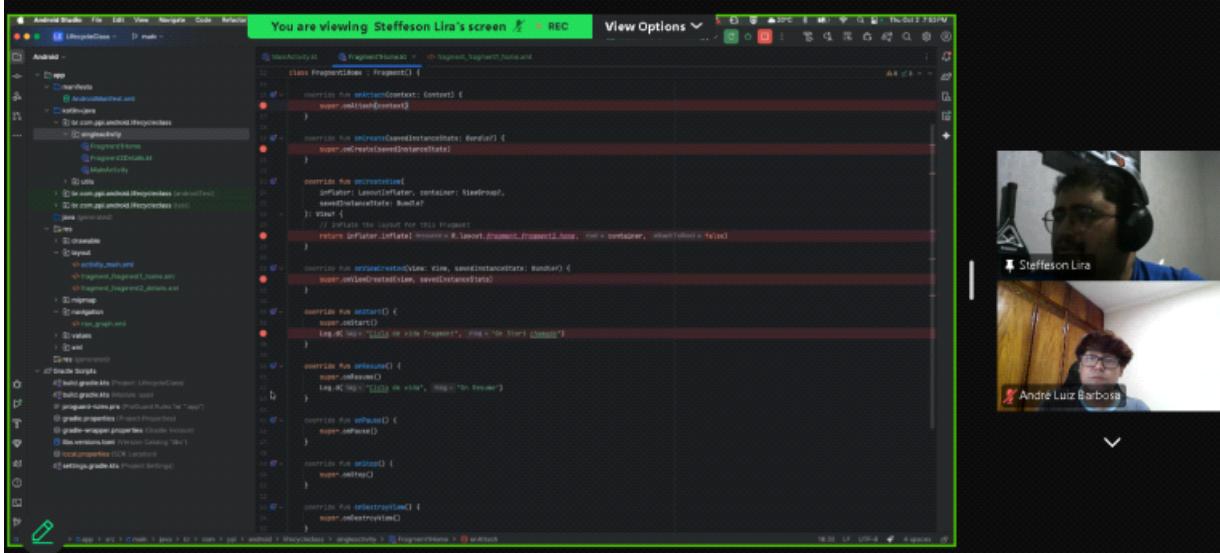
    override fun onStop() {
        super.onStop()
    }

    override fun onDestroyView() {
        super.onDestroyView()
    }

    override fun onDestroy() {
        super.onDestroy()
    }
}
```

Ctrl+shift+y para ver se tem o historico do log??? Windos + Z?

Vai ver o funcionamento pelo debug



```
class FragmentClass : Fragment() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
    }

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

    override fun onStart() {
        super.onStart()
    }

    override fun onResume() {
        super.onResume()
    }

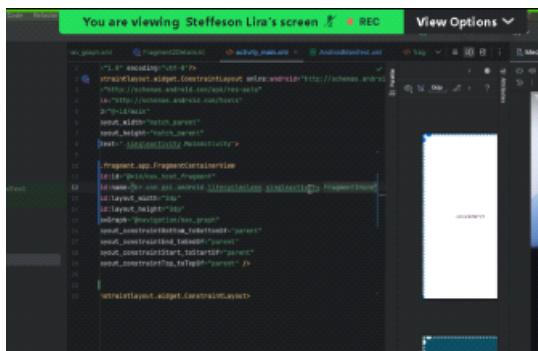
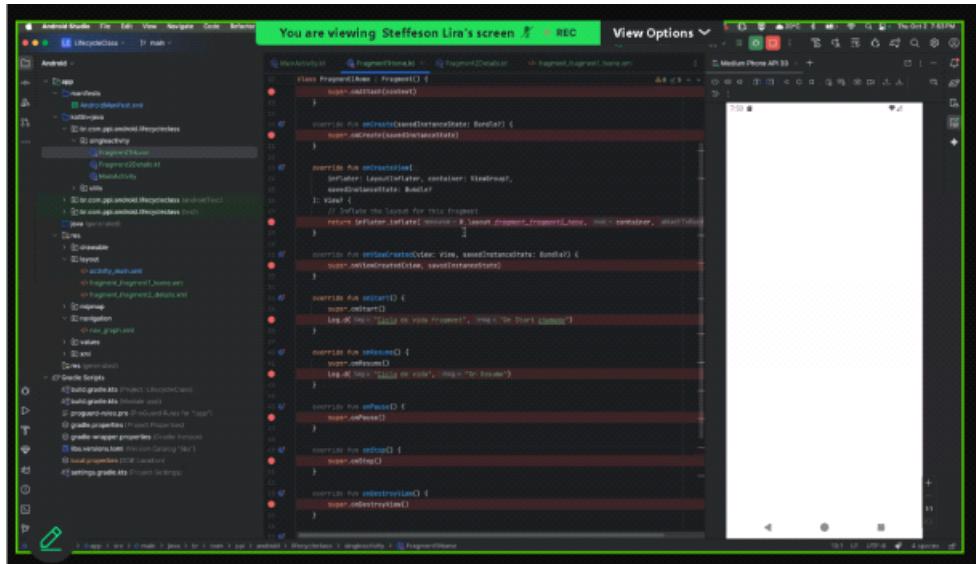
    override fun onPause() {
        super.onPause()
    }

    override fun onStop() {
        super.onStop()
    }

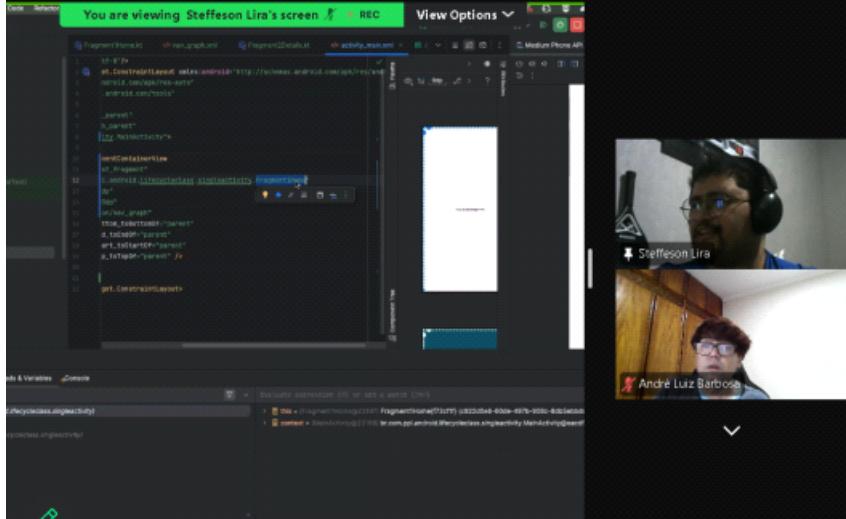
    override fun onDestroyView() {
        super.onDestroyView()
    }

    override fun onDestroy() {
        super.onDestroy()
    }
}
```

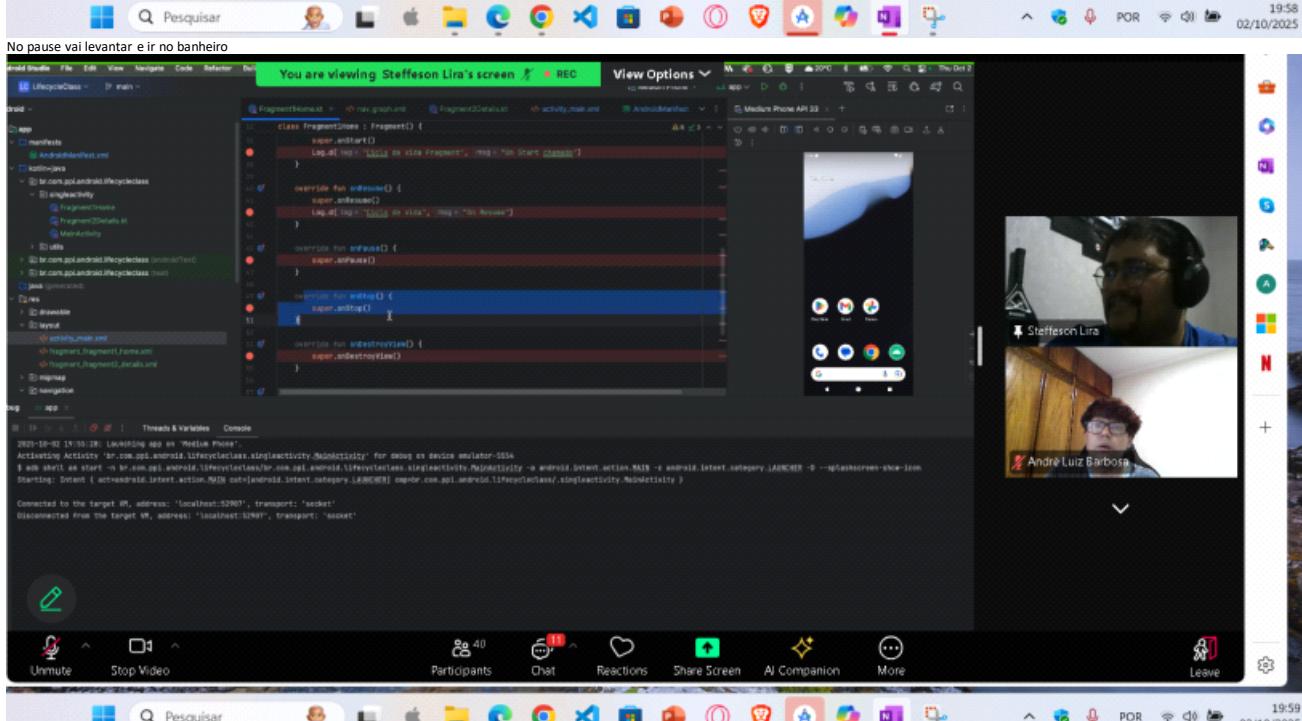
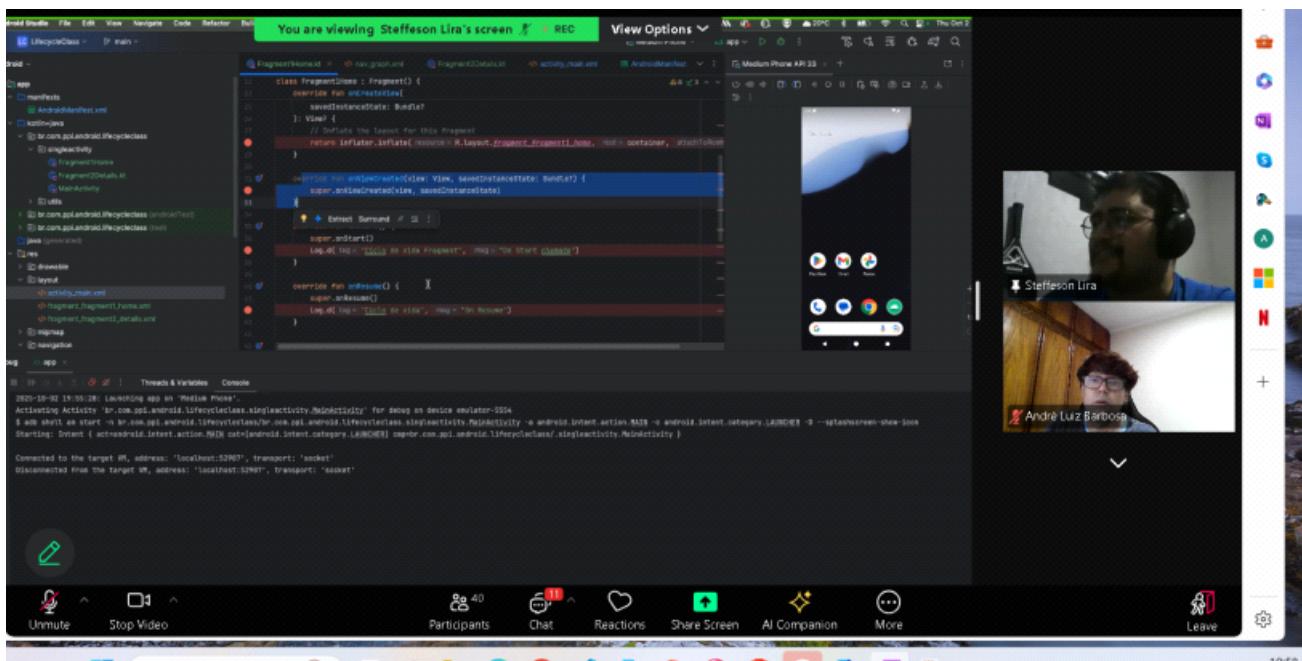
não chamou a activity o que ocorreu?

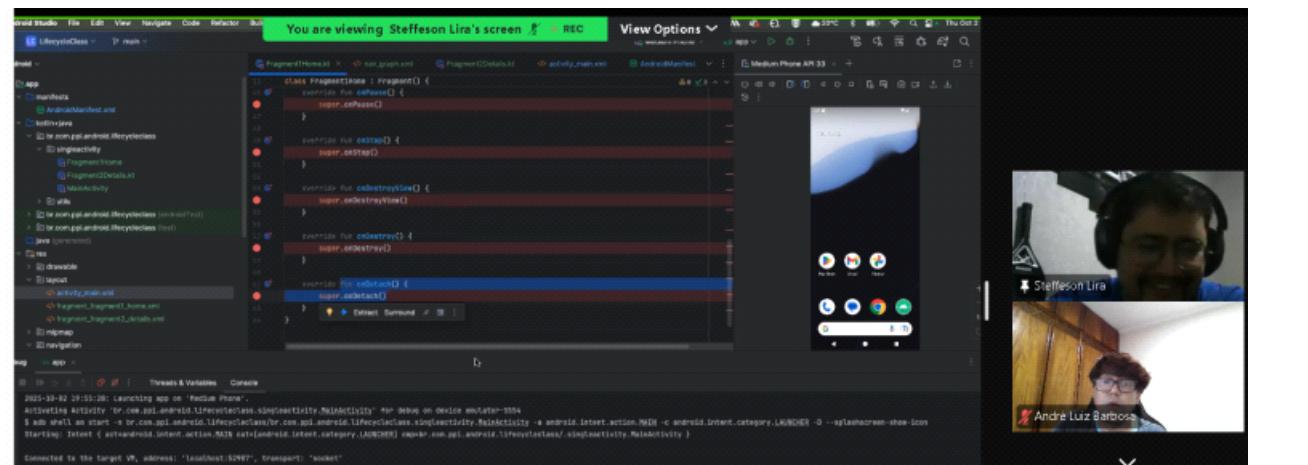
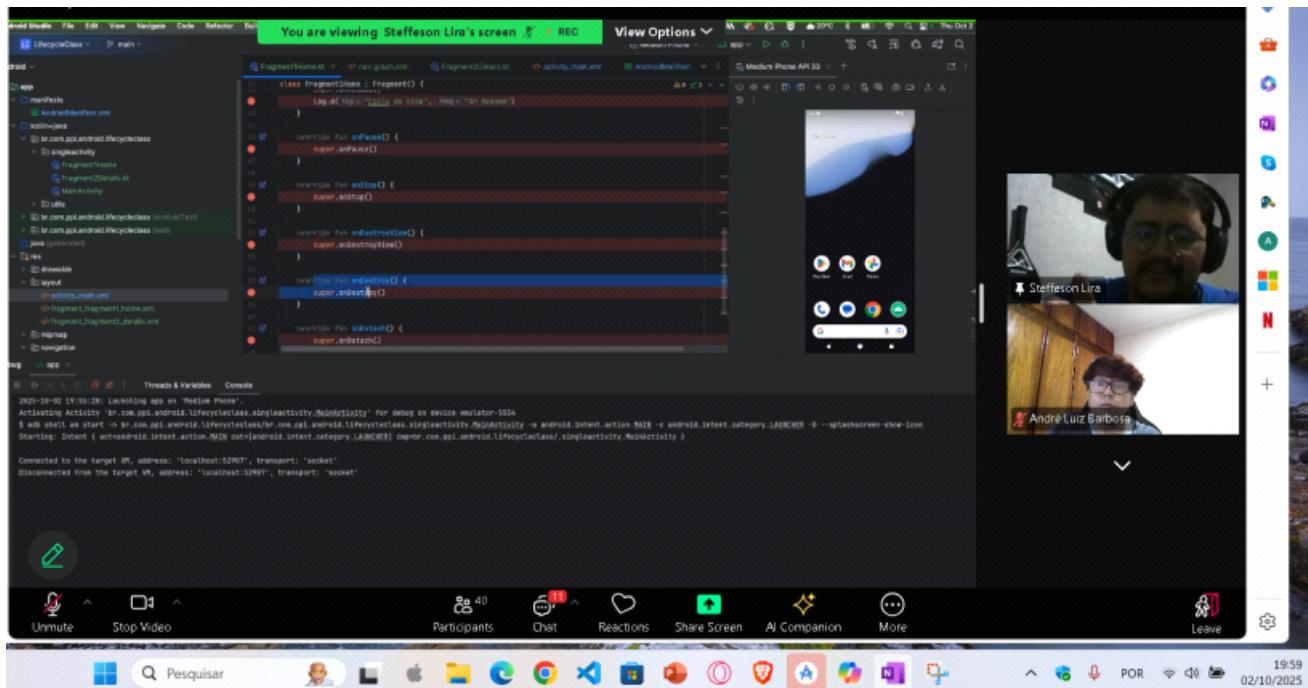


Tem que colocar o nome e referenciar o fragment

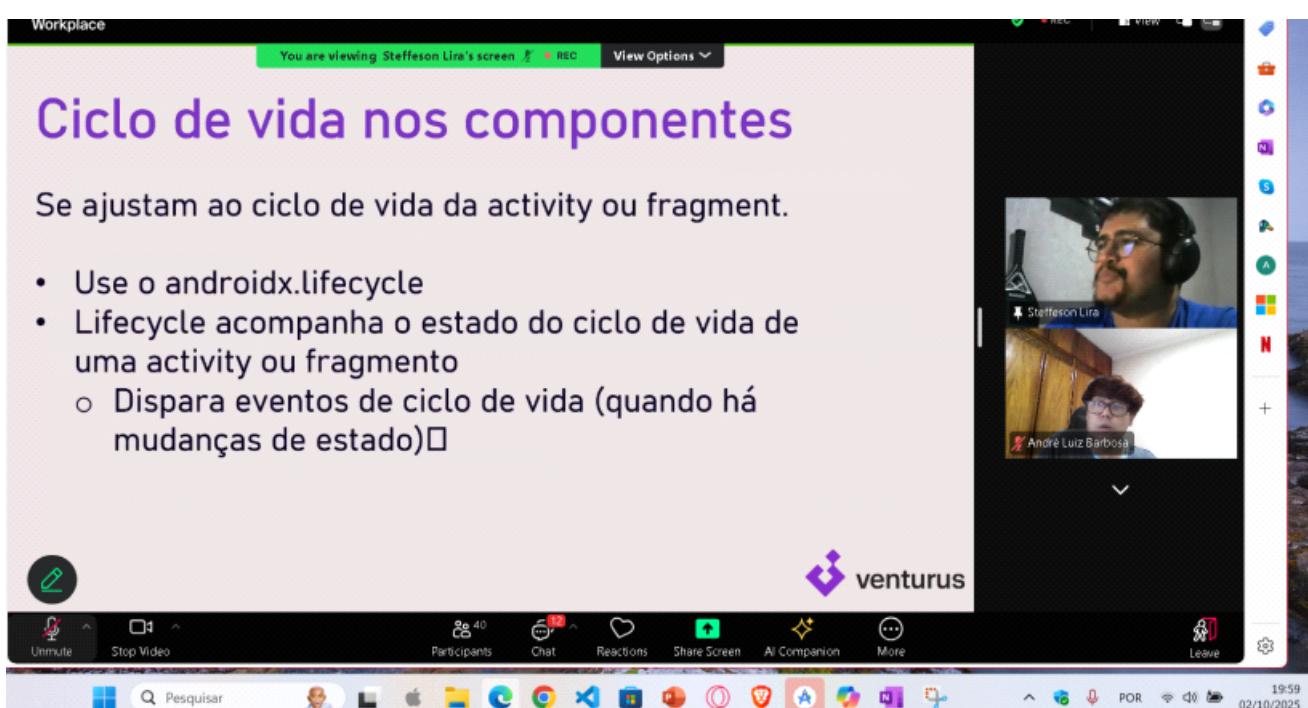


Fez analogia do debug com tomar um café da manha e garcom servindo o café





No onDetch sai da cafeteria



LifecycleOwner

- Interface que indica que esta classe possui um ciclo de vida
- Implementadores devem implementar o método `getLifecycle`

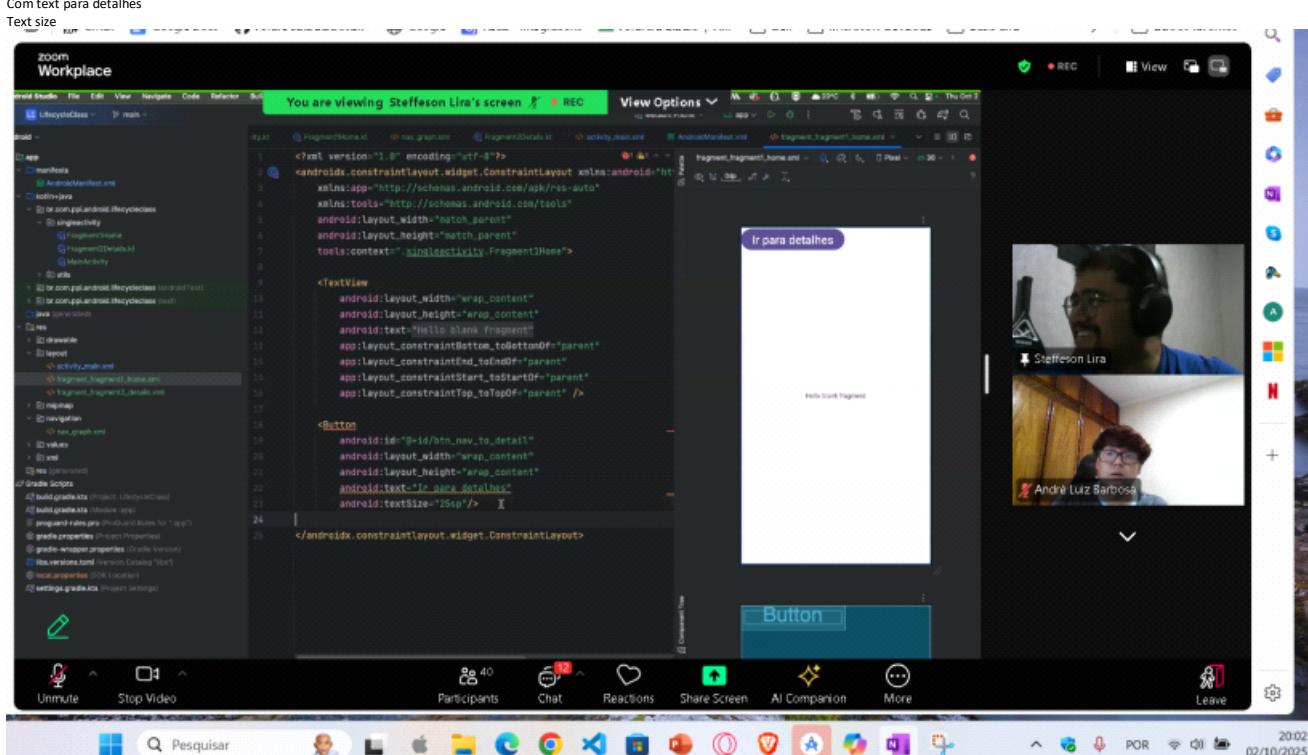
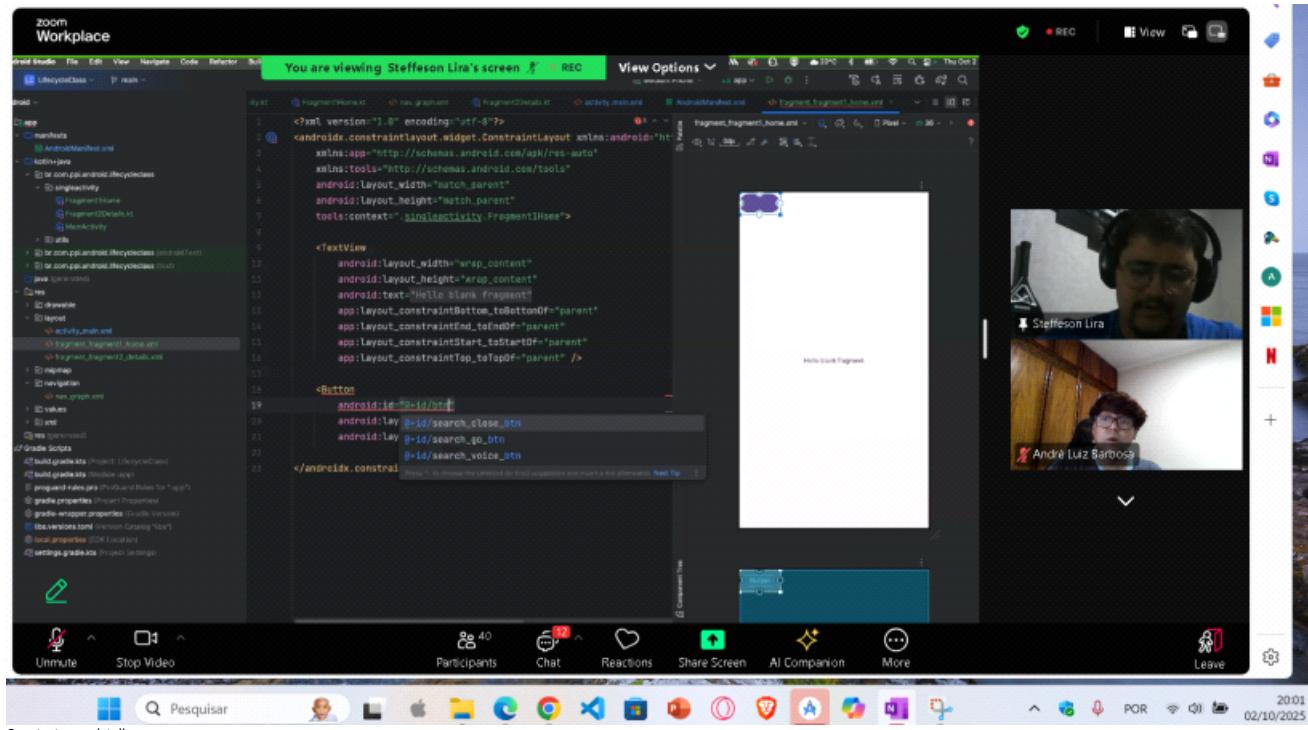
Exemplos: Fragment e AppCompatActivity são implementações de LifecycleOwner

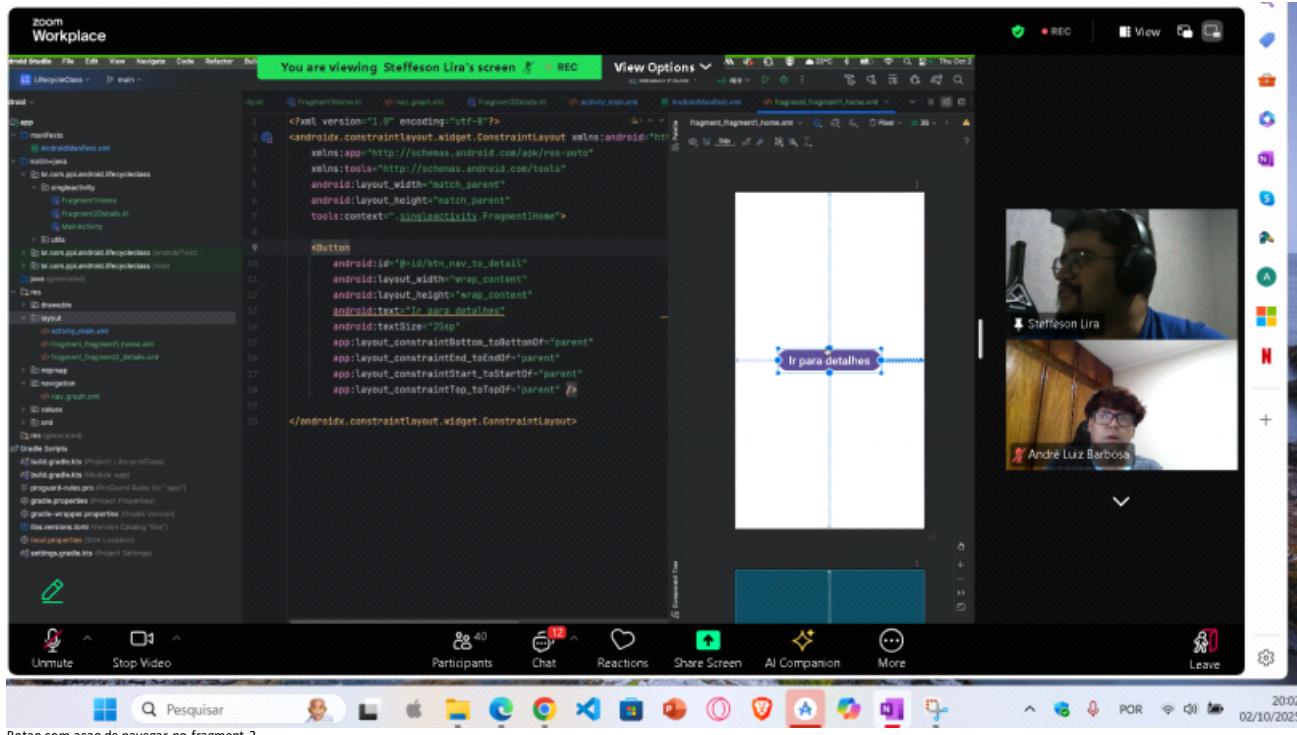
LifecycleObserver

Implemente a interface LifecycleObserver:

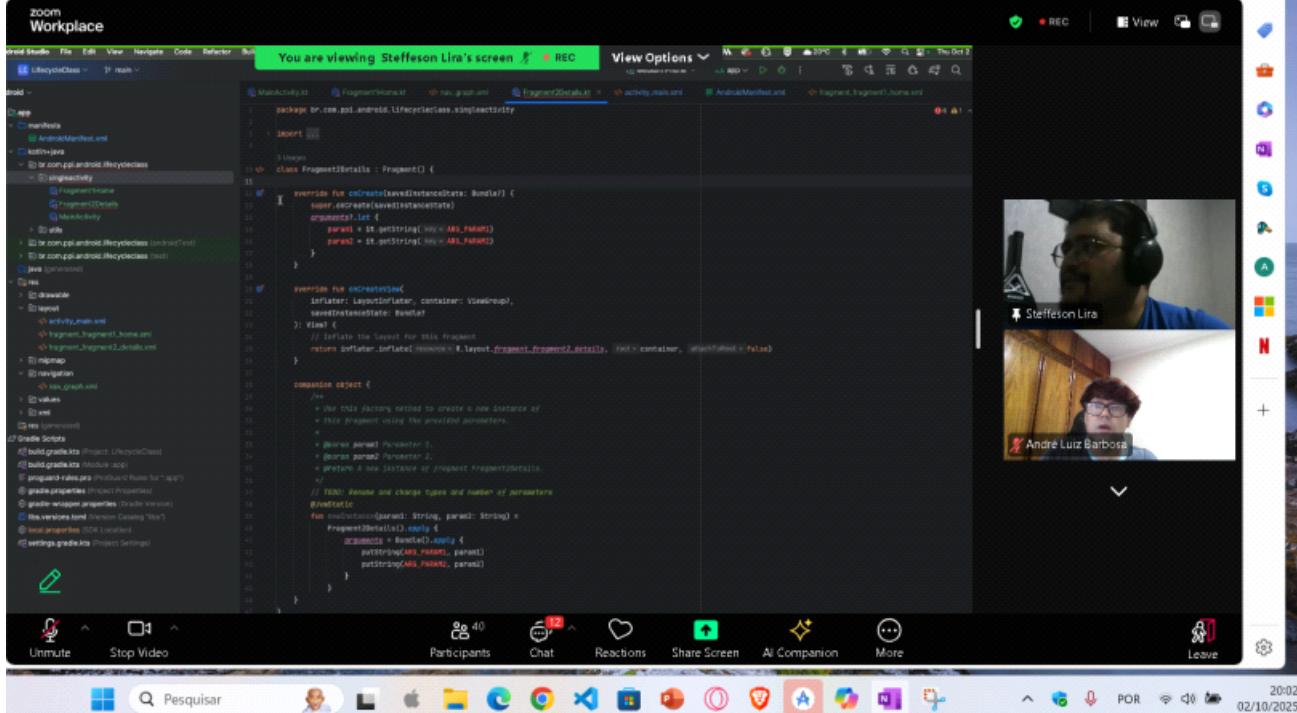
```
class MyObserver : DefaultLifecycleObserver{  
    override fun onStart(owner: LifecycleOwner)  
    }  
...  
Adicione o observador ao ciclo de vida:  
viewLifecycleOwner.lifecycle().addObserver(MyObserver())
```

Explикando no código os observadores  
Ботао para mudança de tela  
Para ver estados e tudo o mais da tela

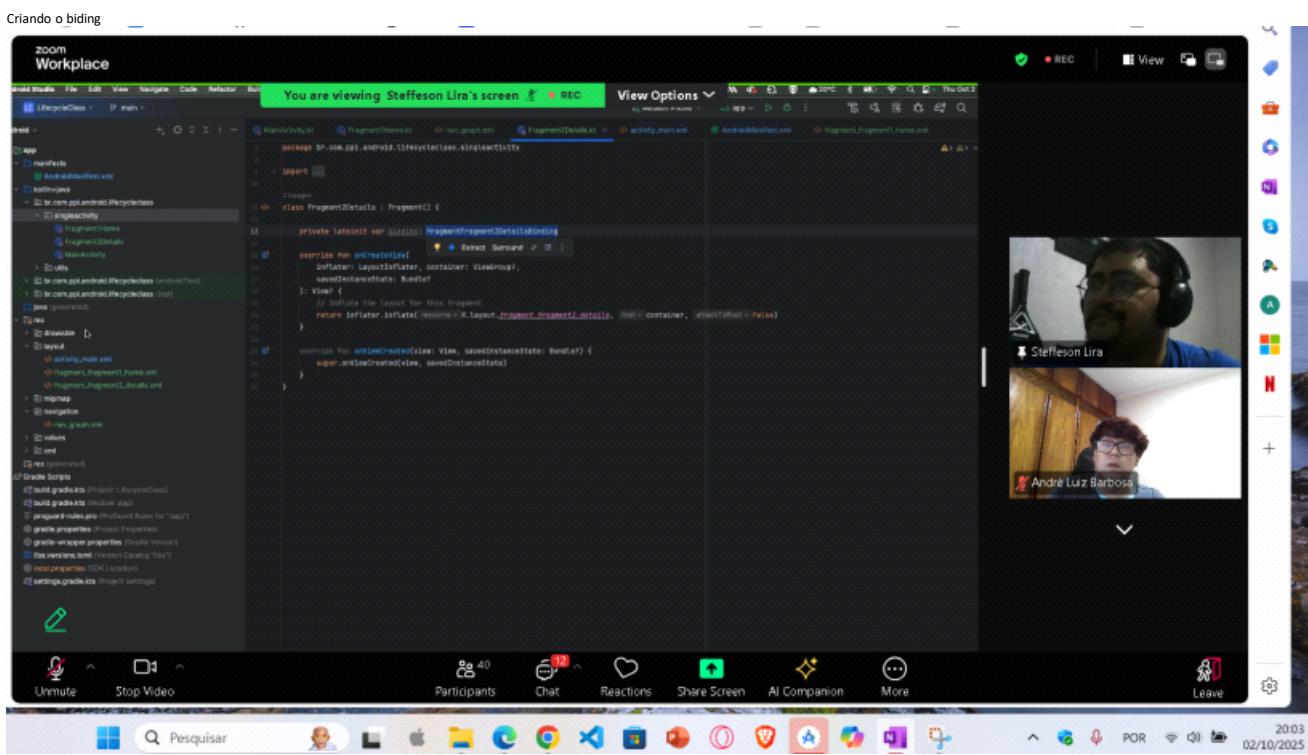
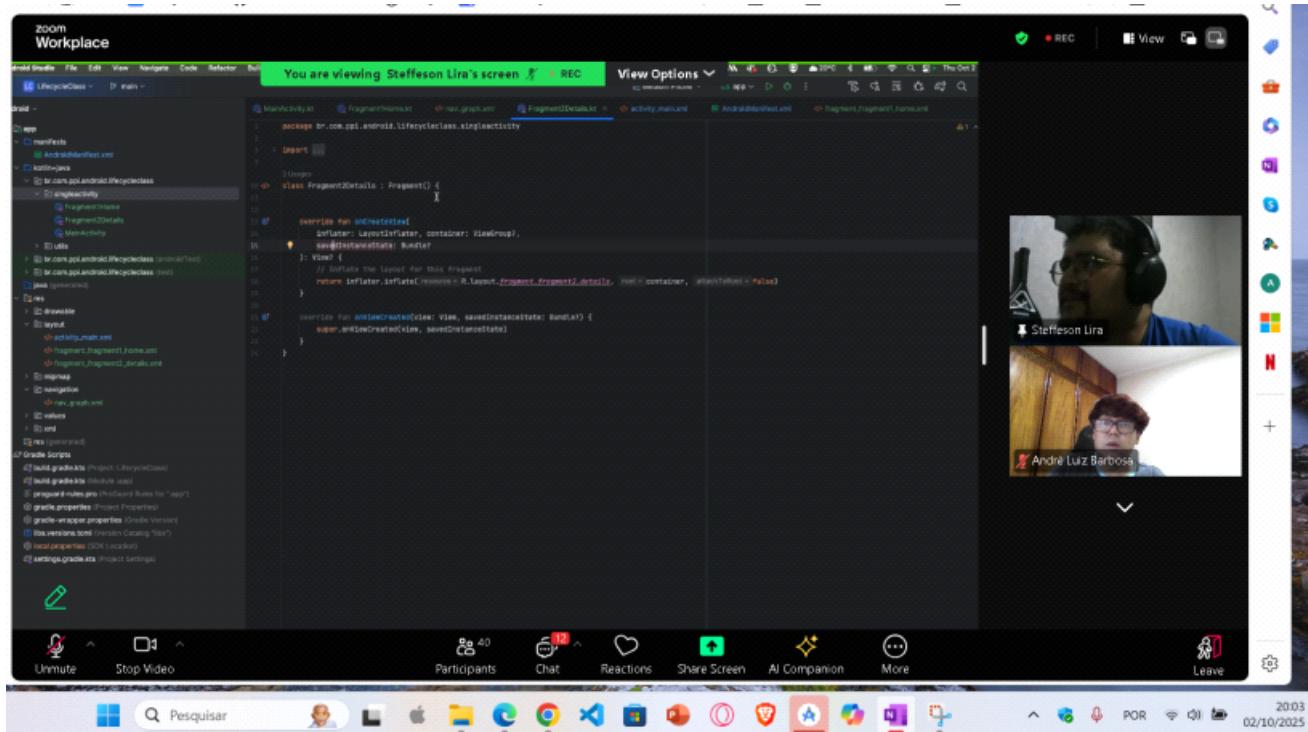




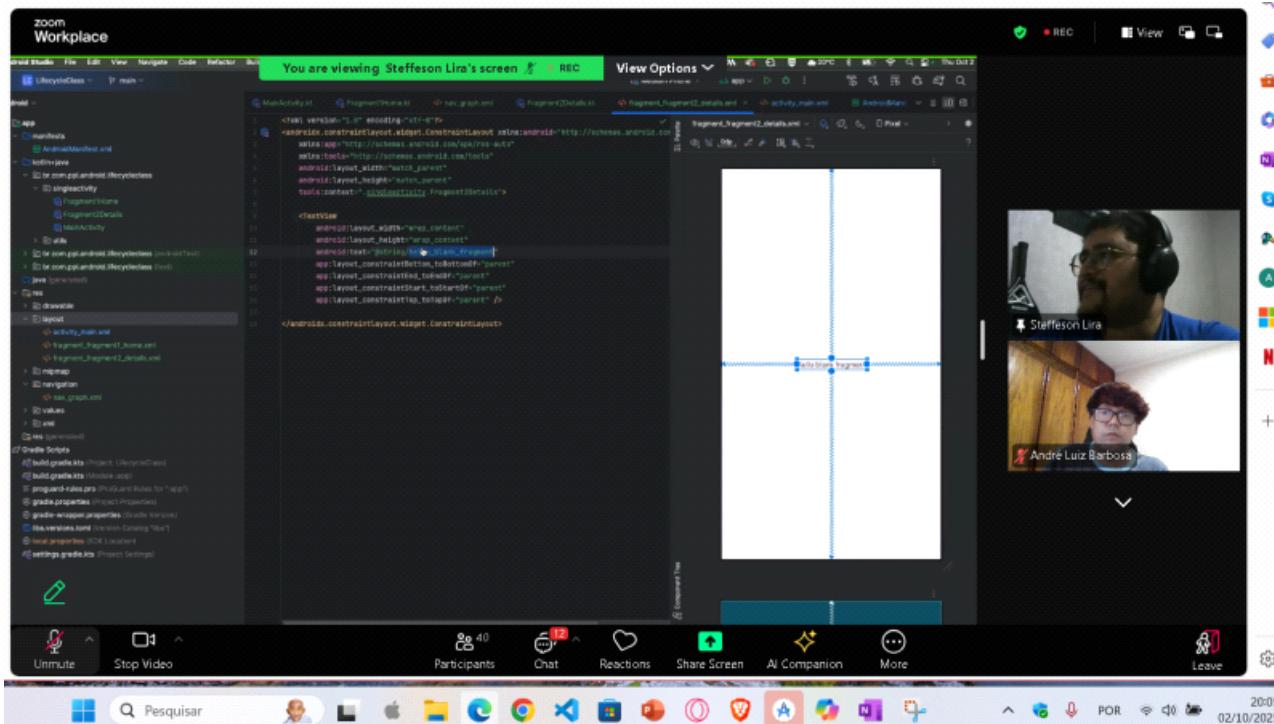
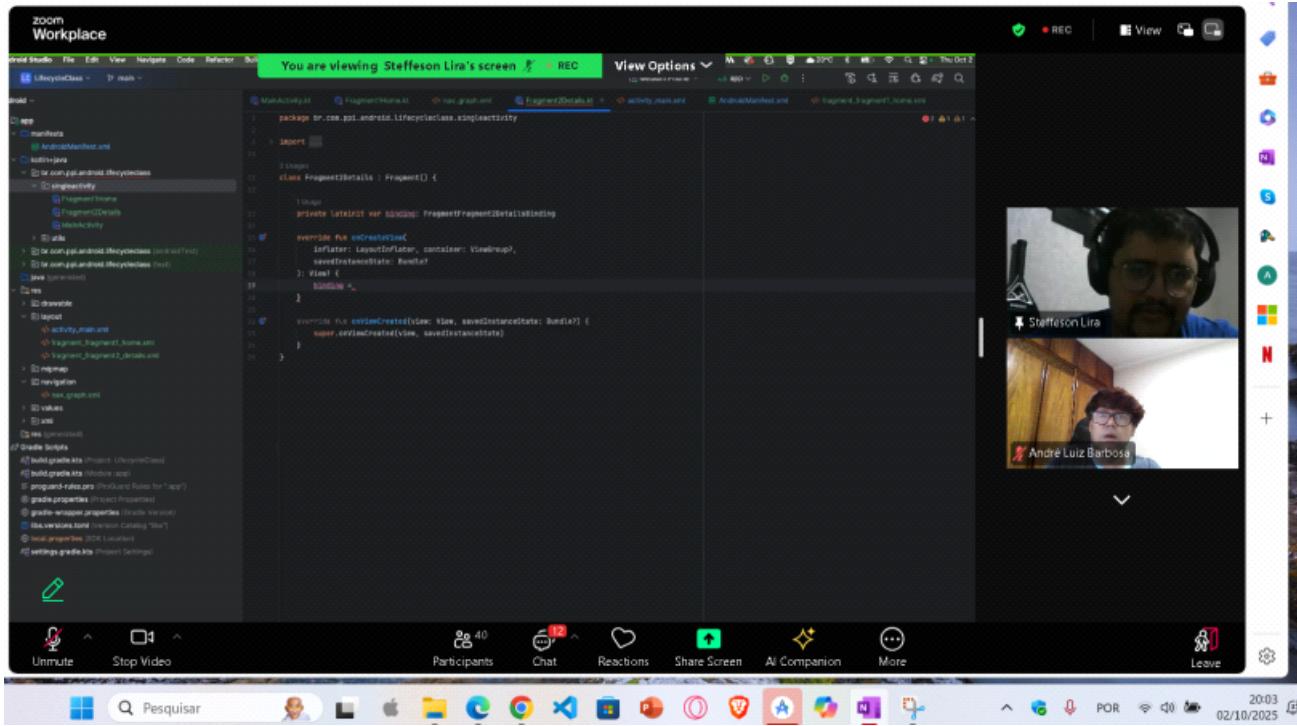
Bota o com ação de navegar no fragment 2

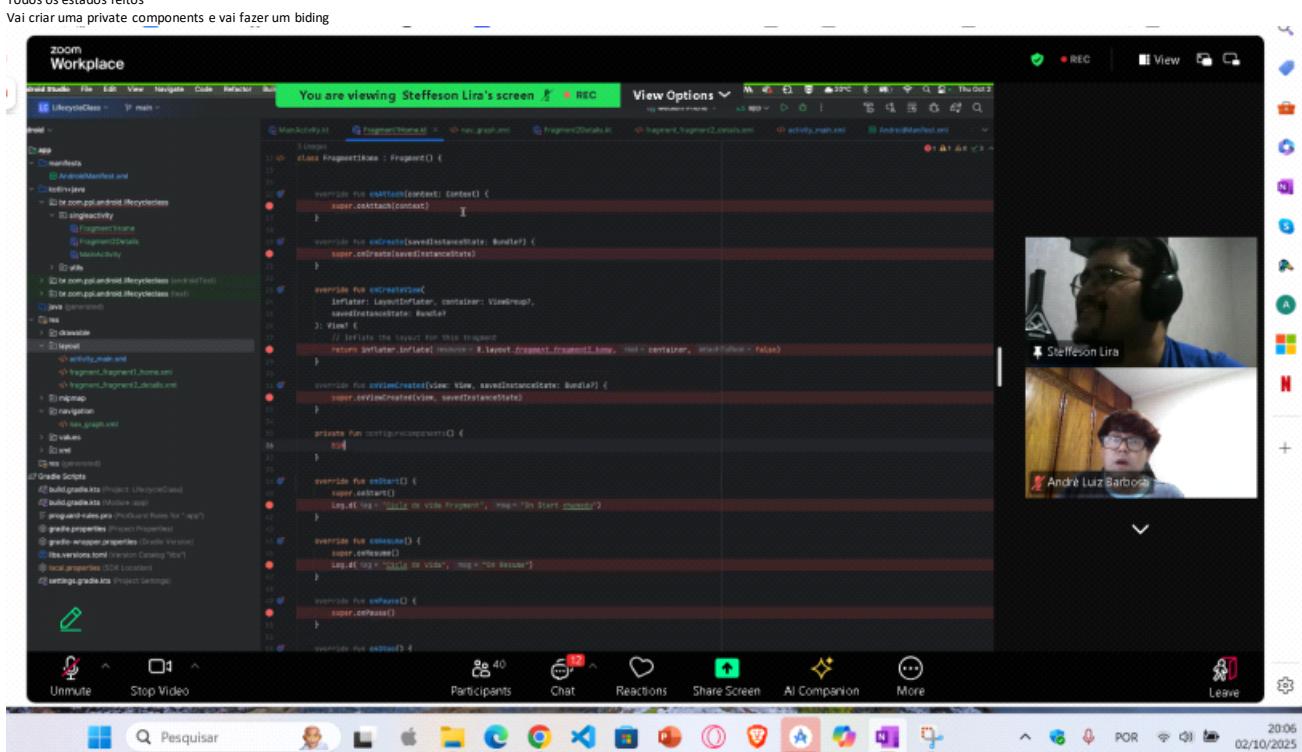
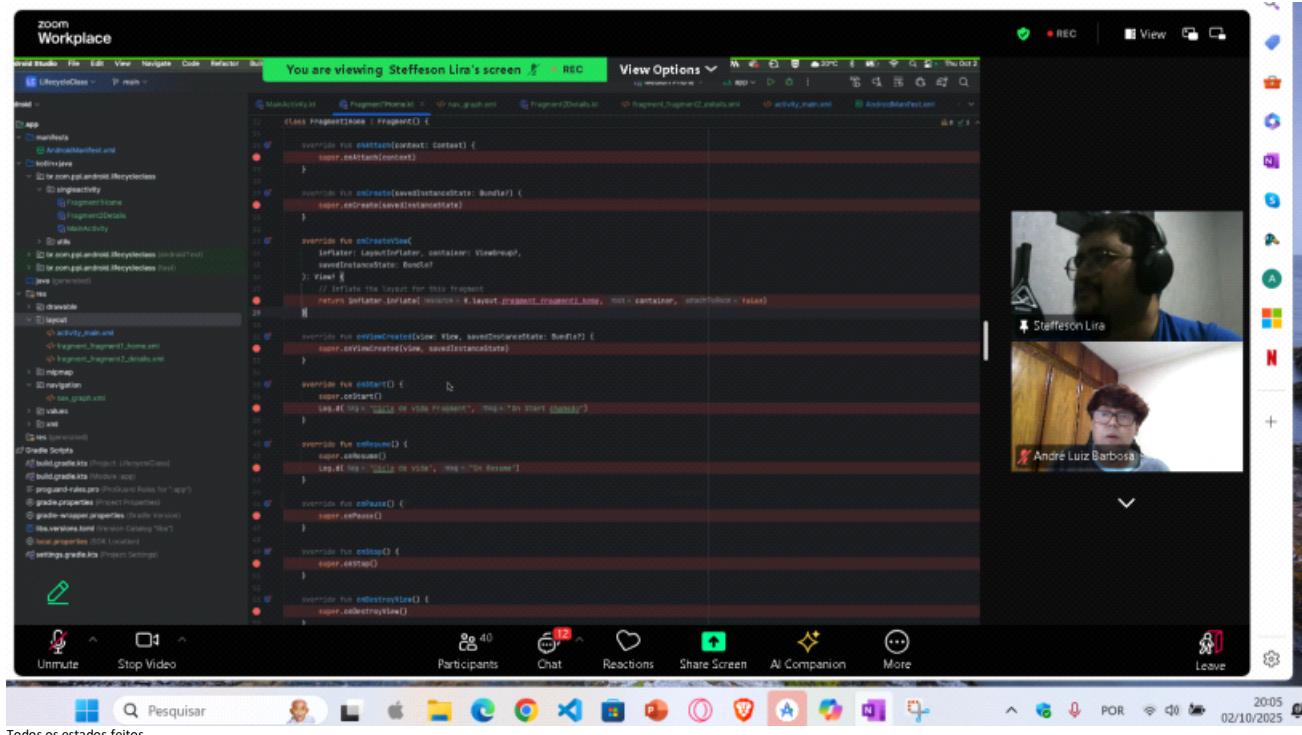


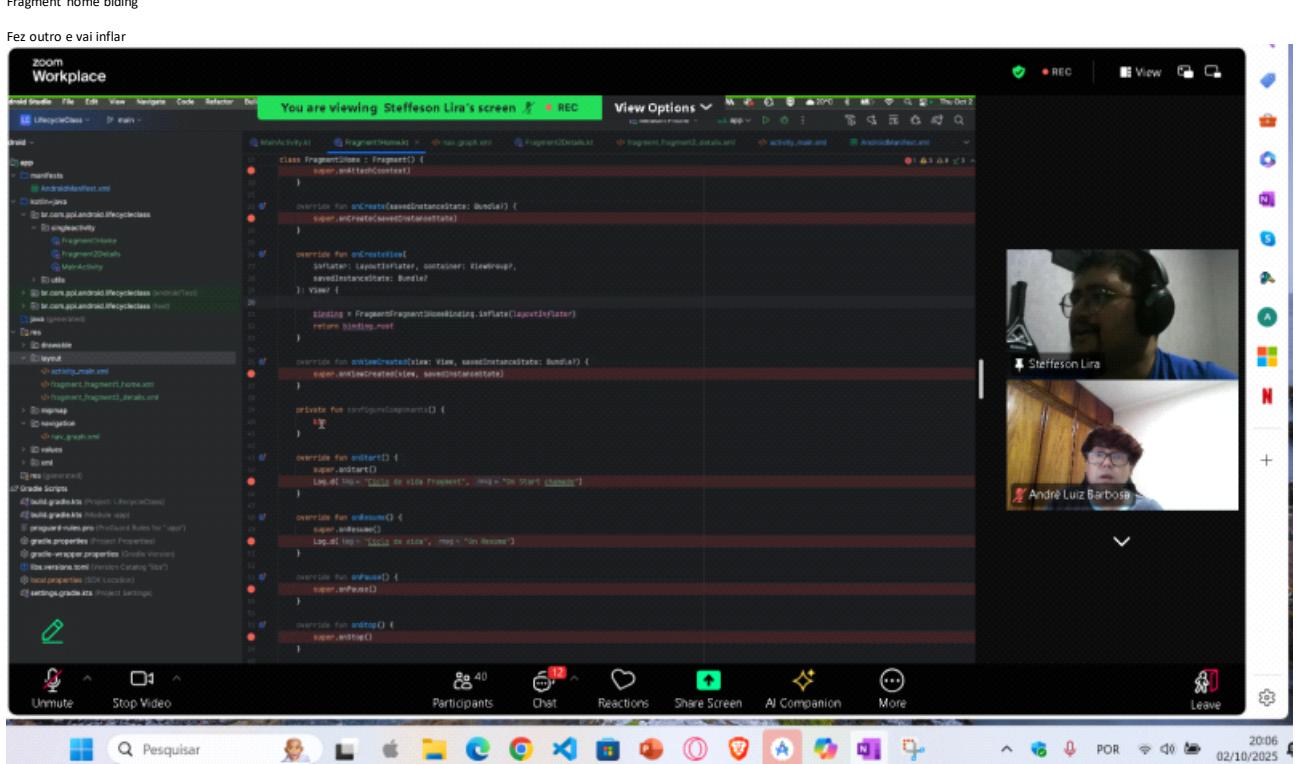
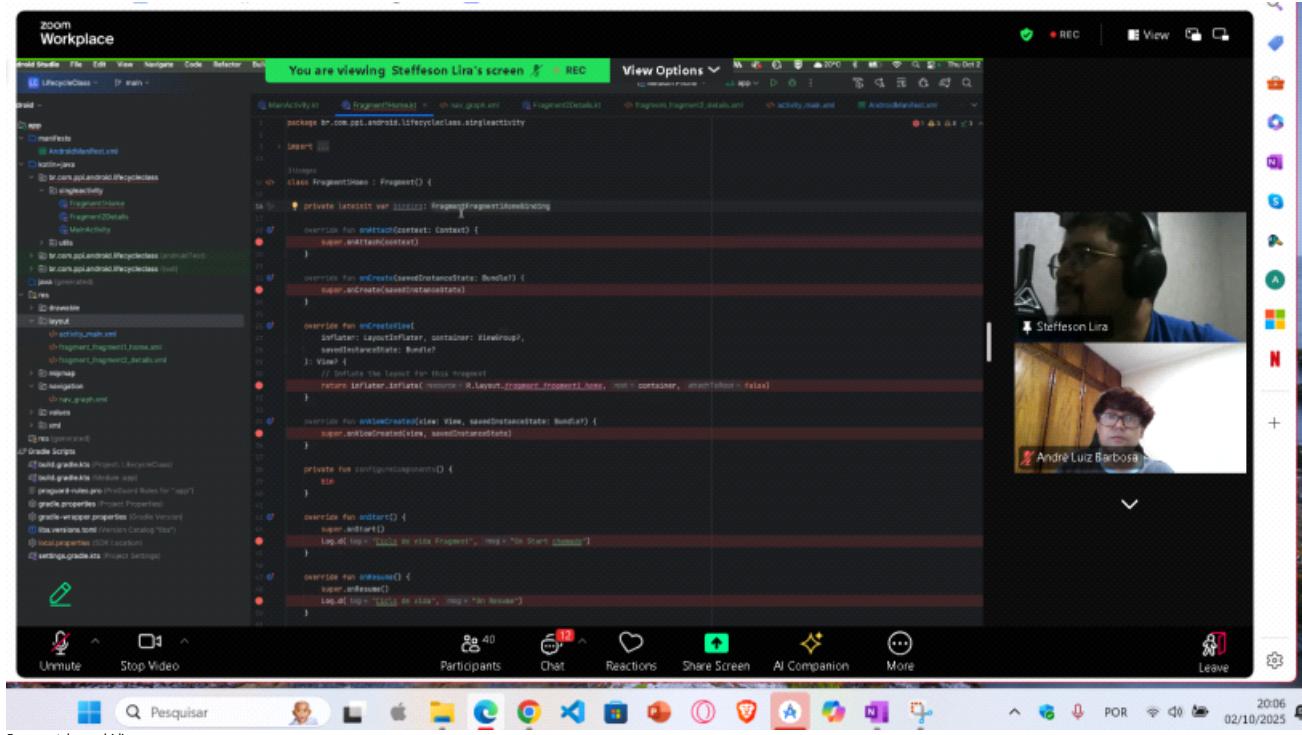
Vai tirar um onCreate de deixar um  
O professor recomenda deixar apenas estes dois

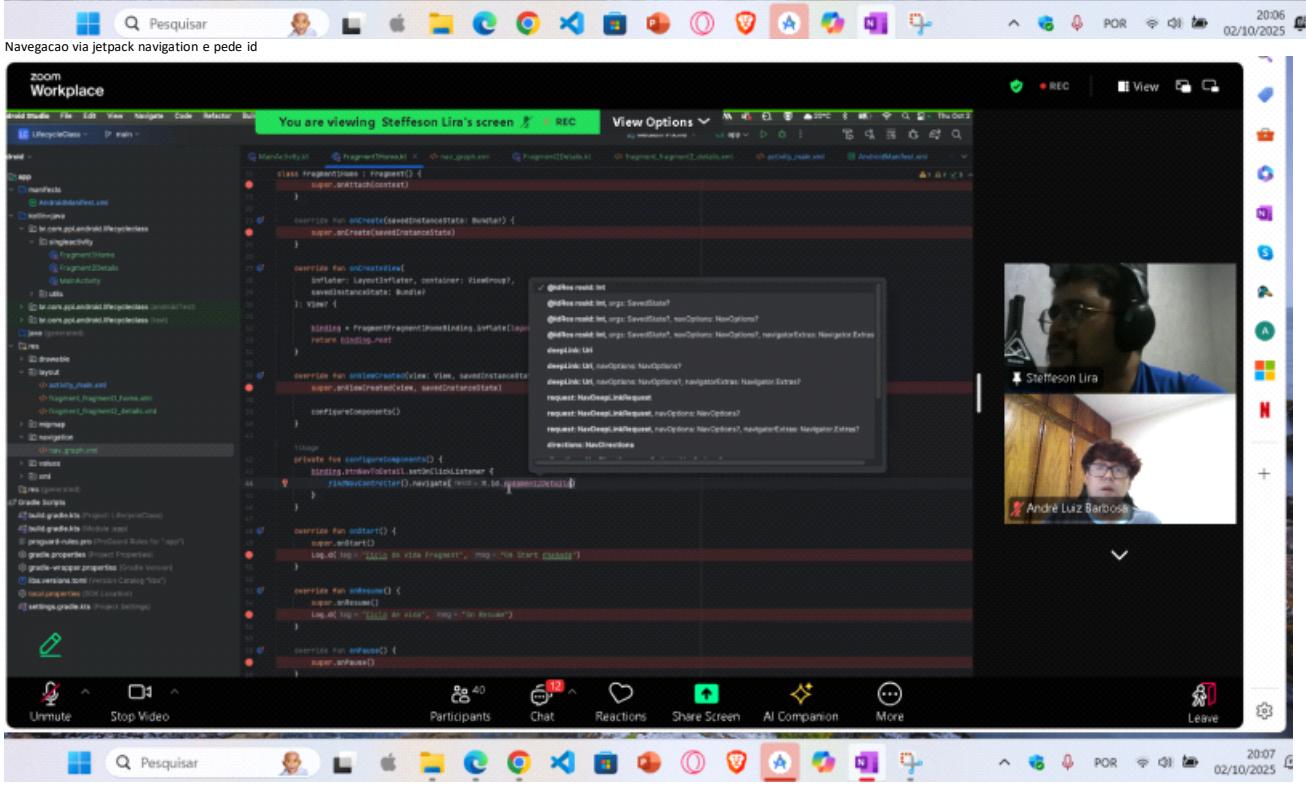
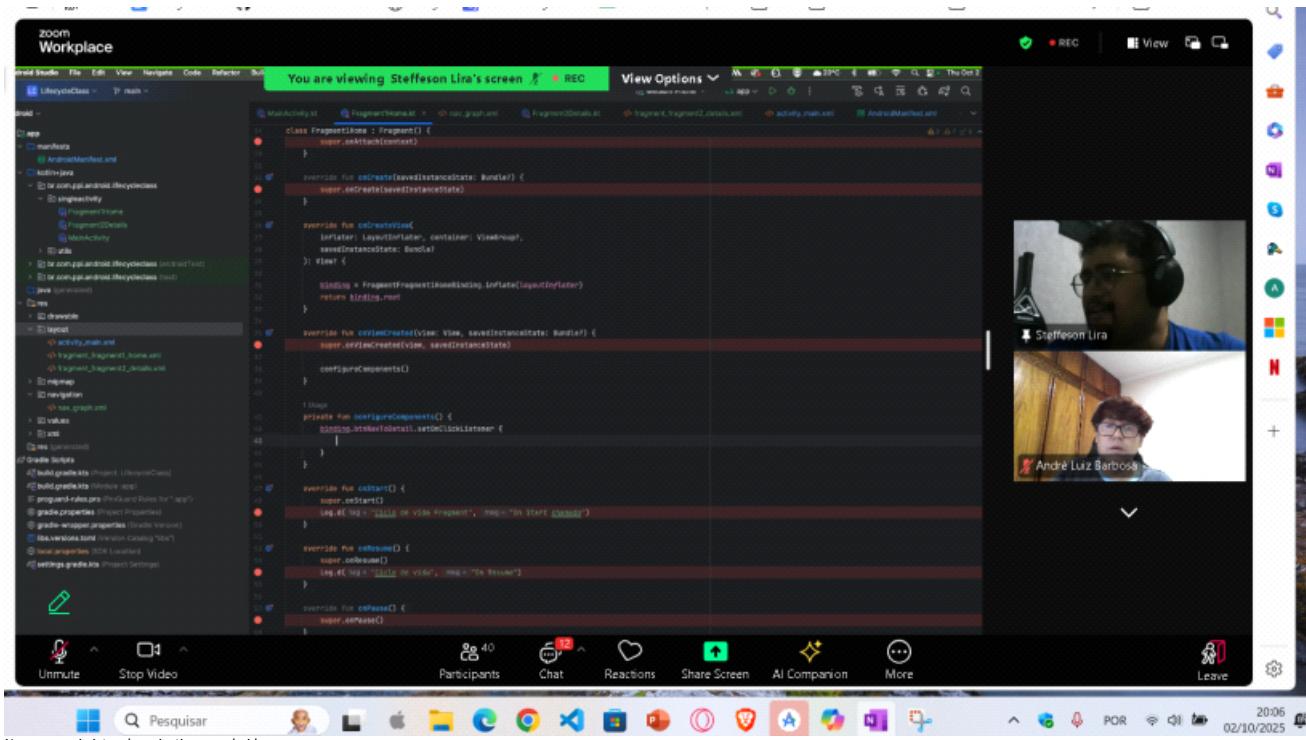


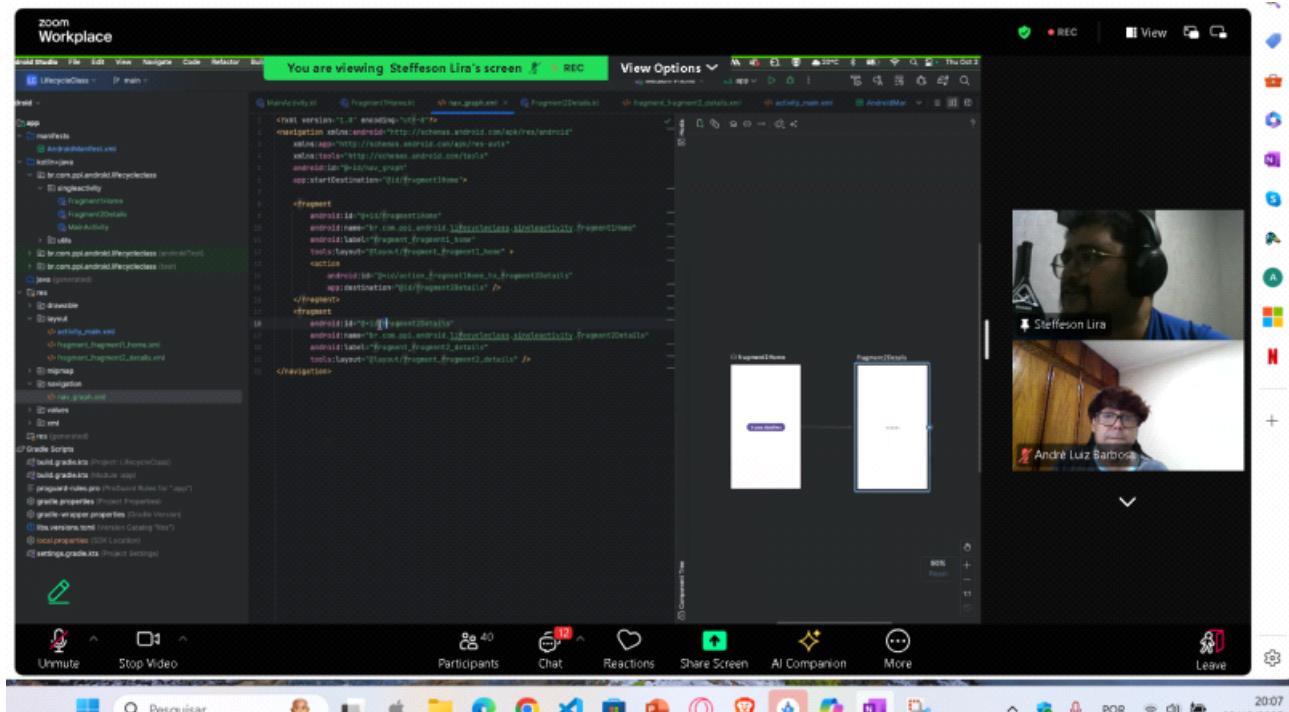
Para achar o nome do biding ir sempre no ???binding receive



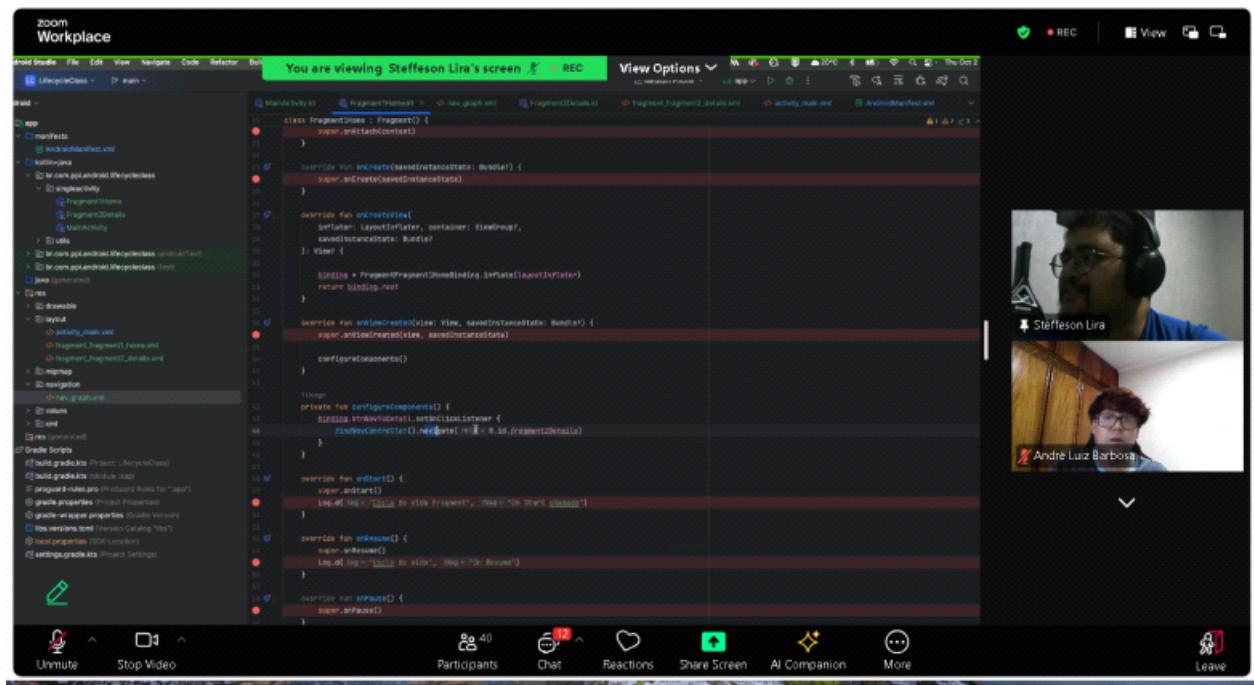
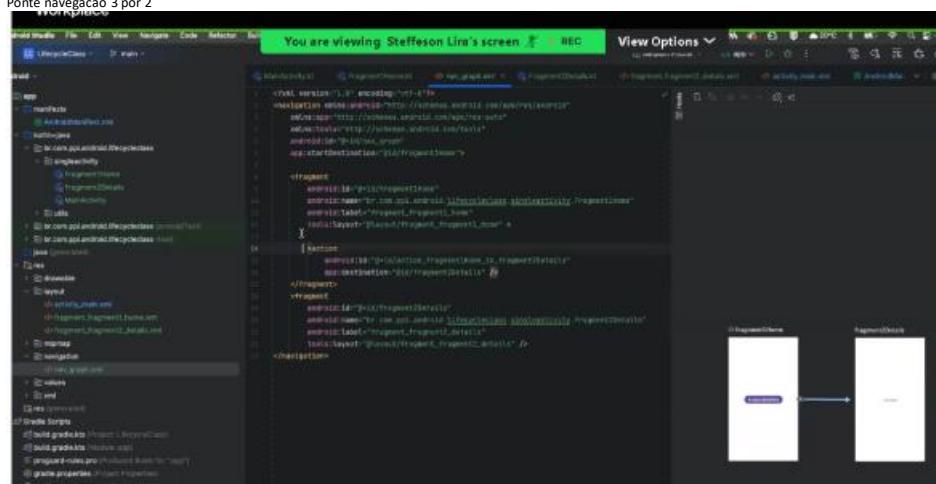


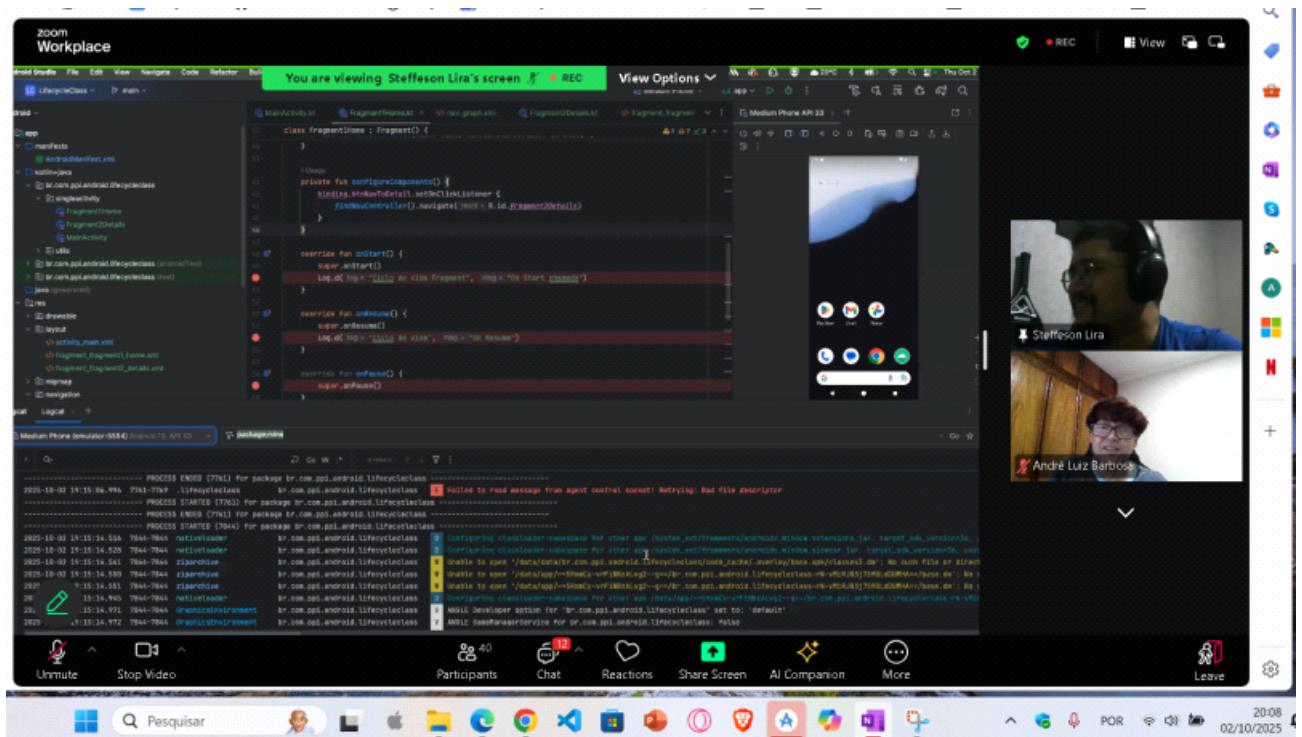
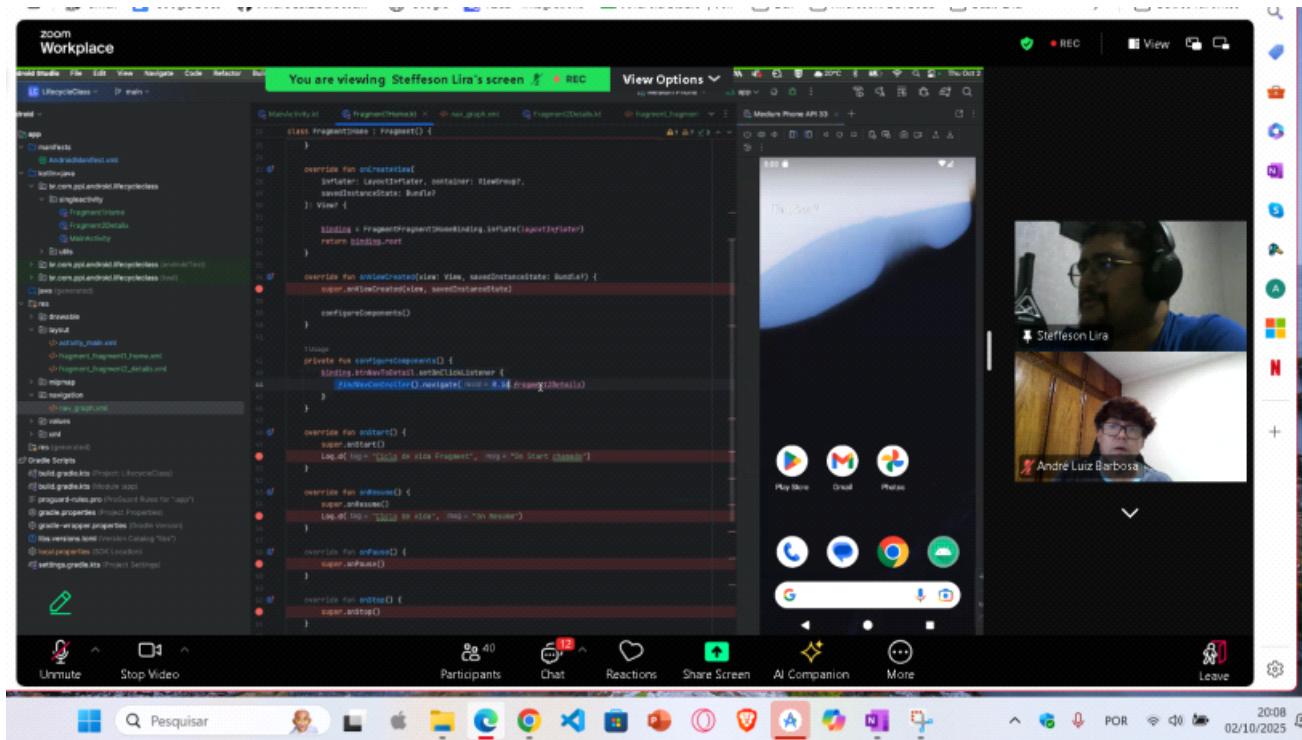


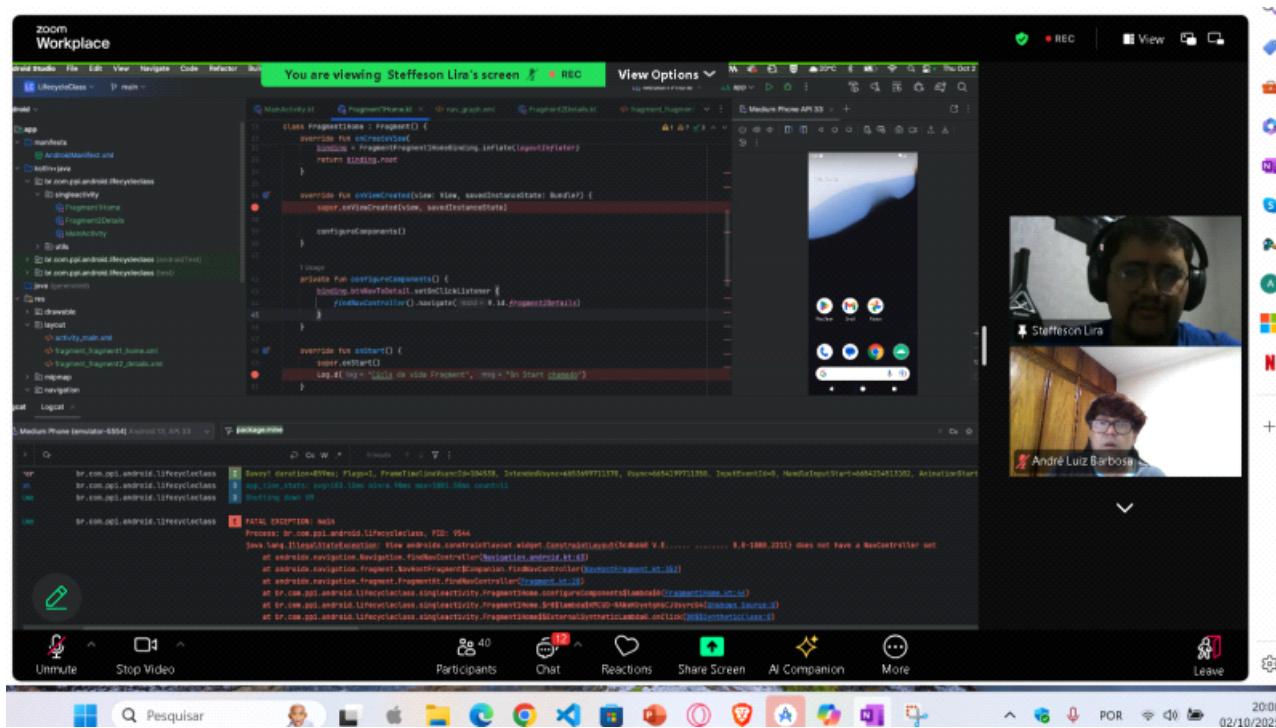
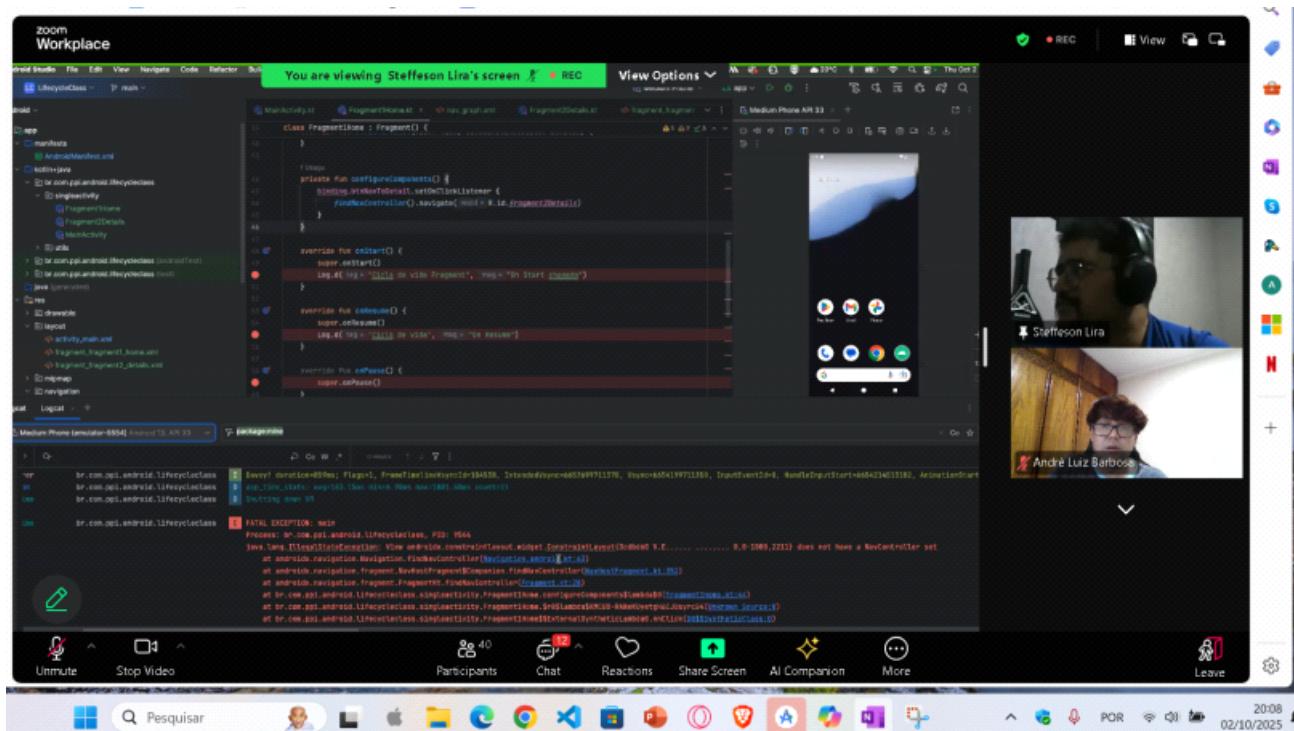


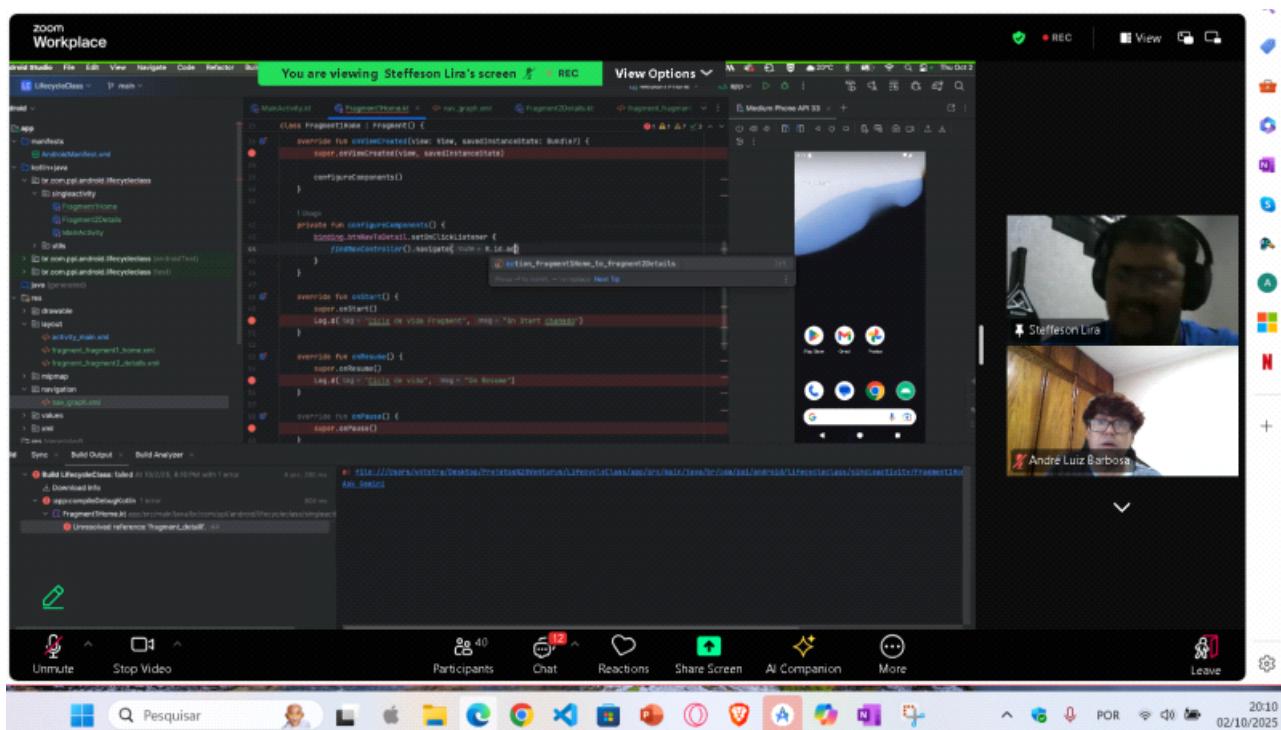
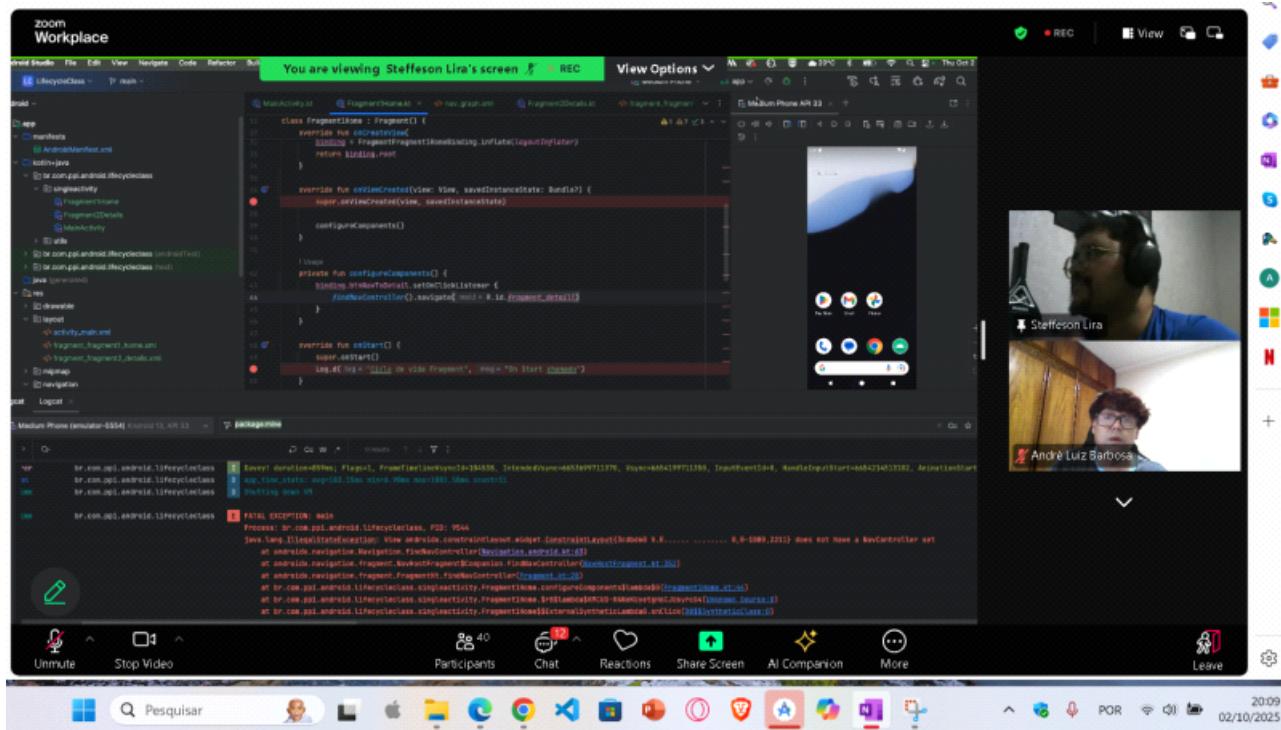


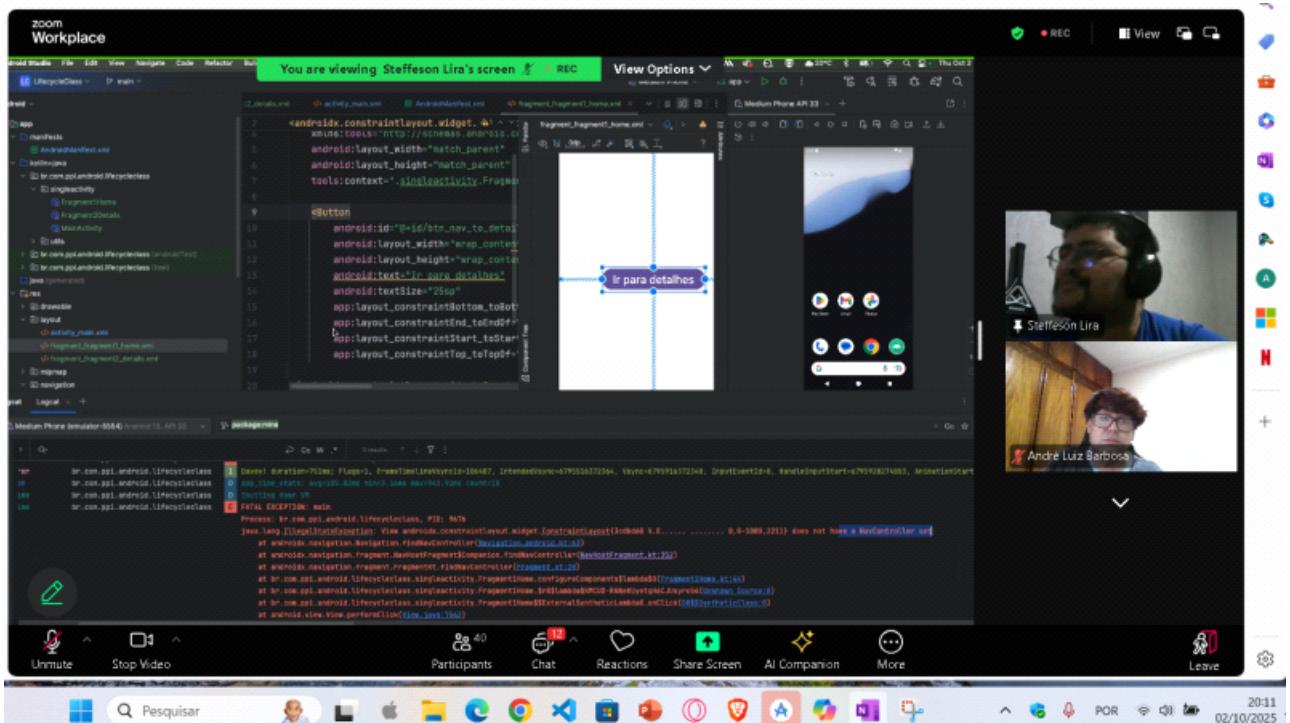
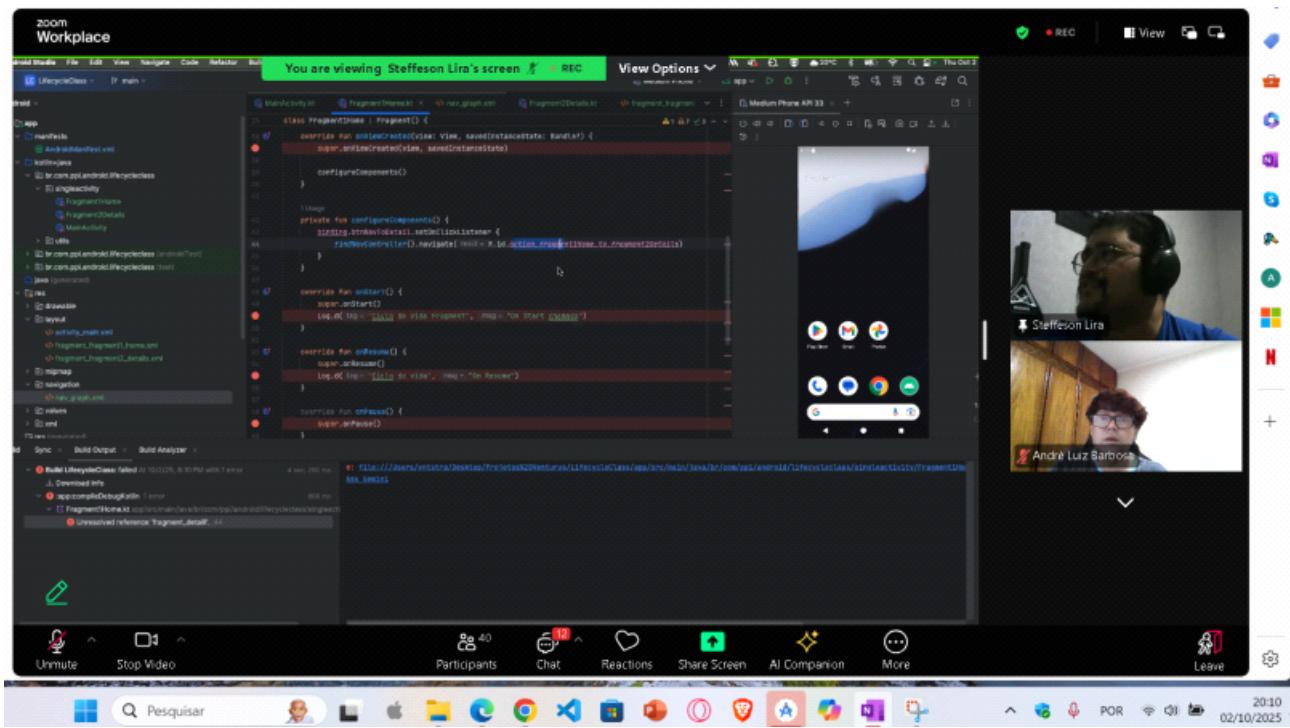
Ponte navegação 3 por 2

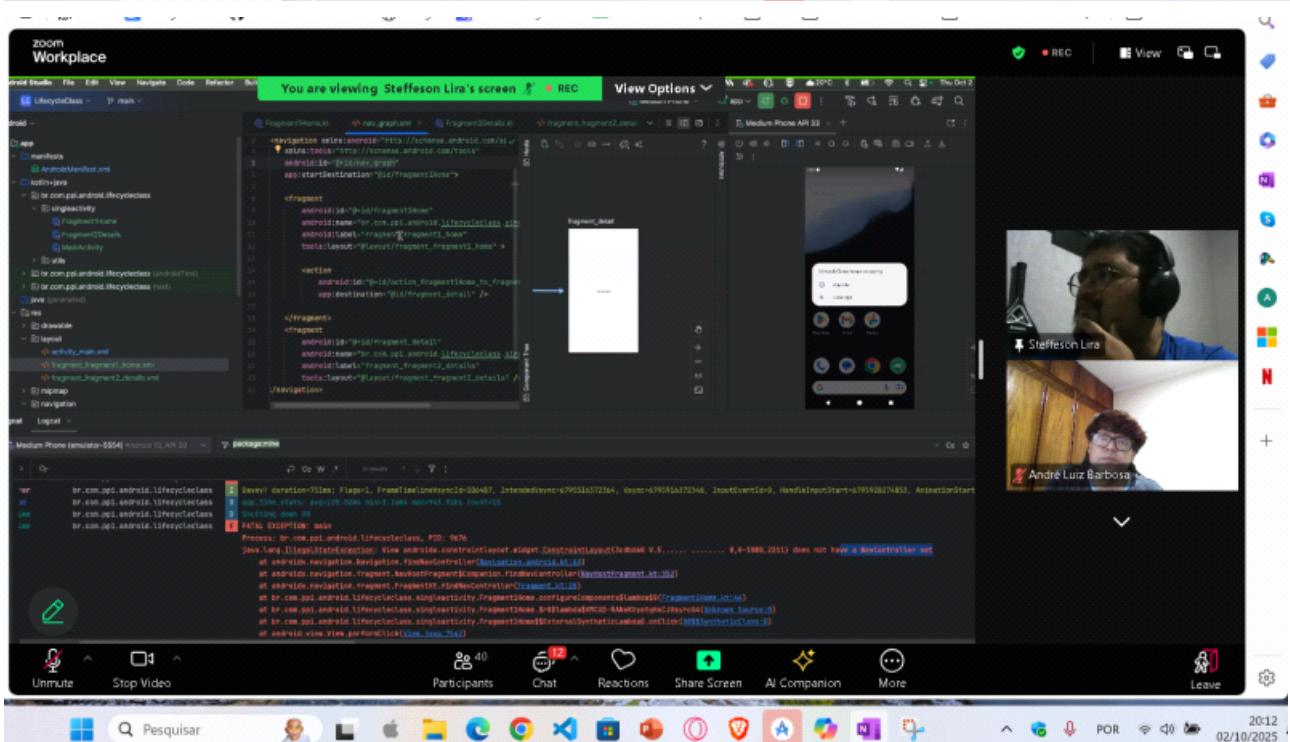
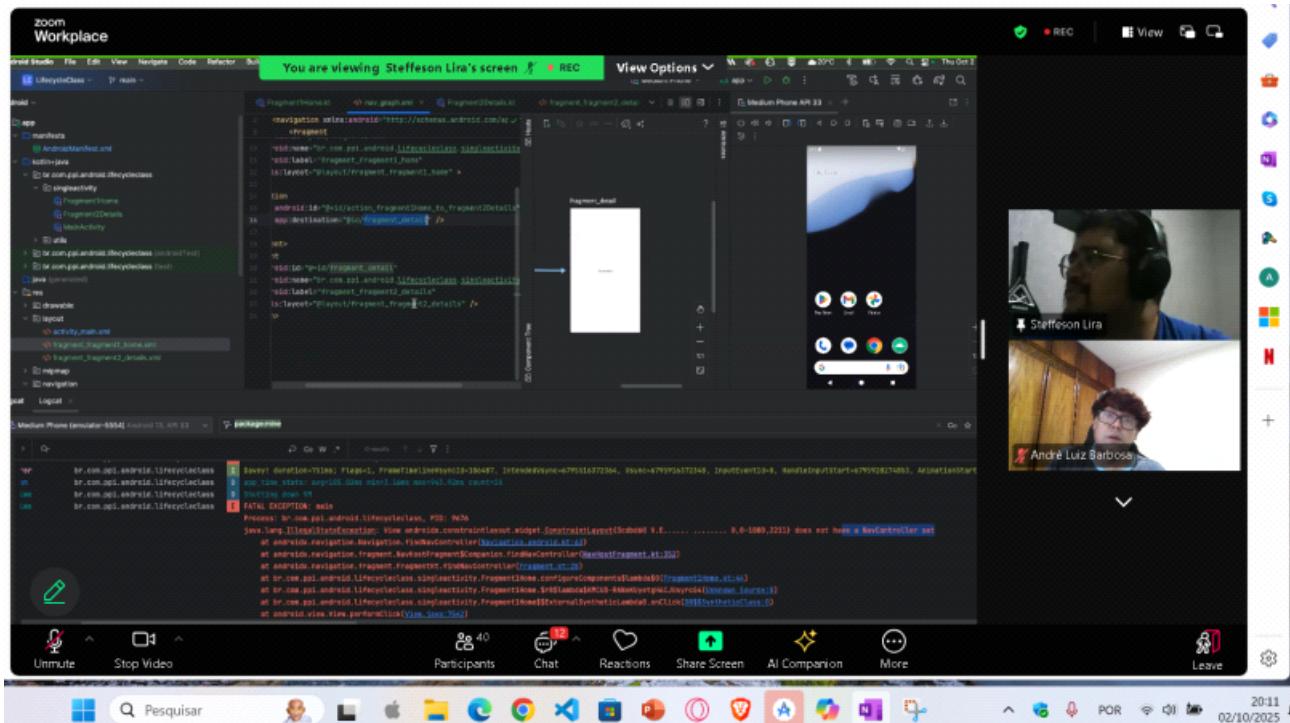




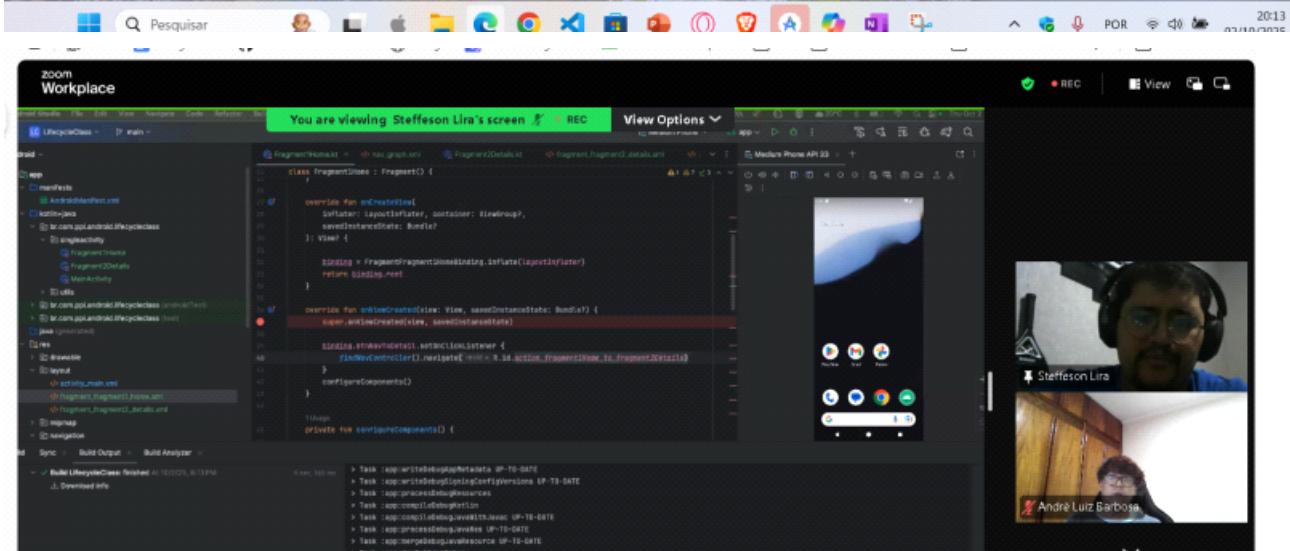
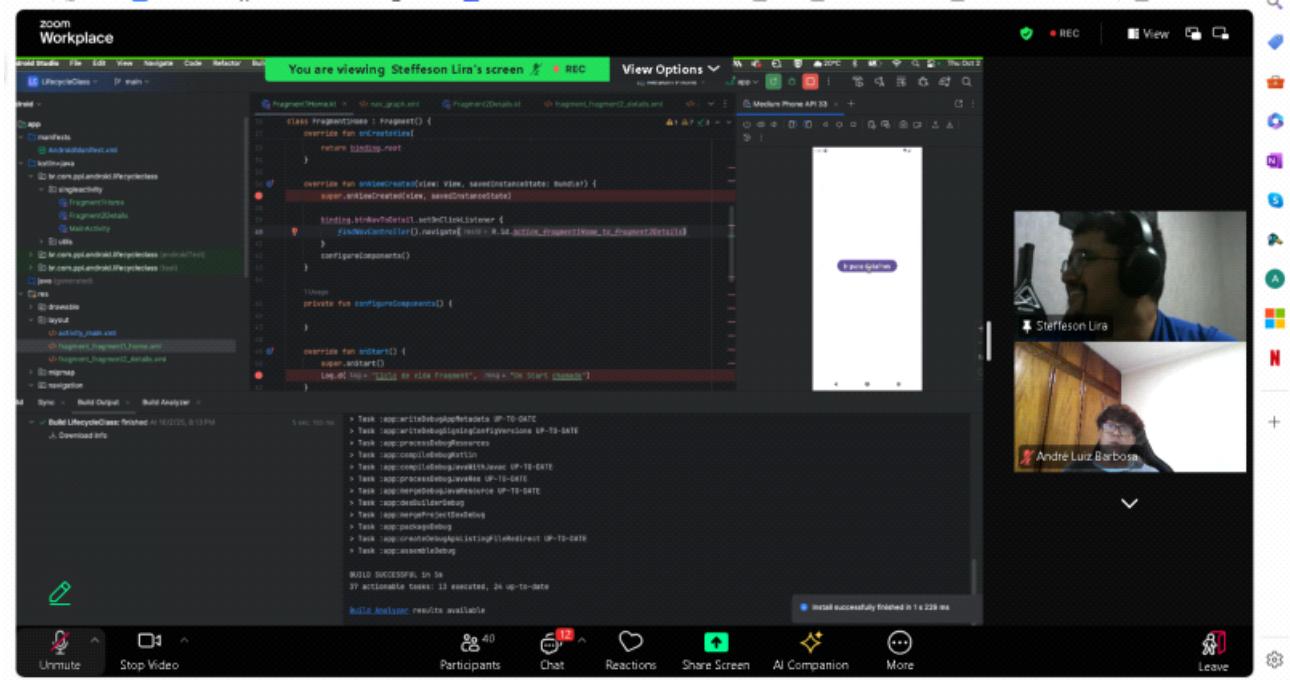
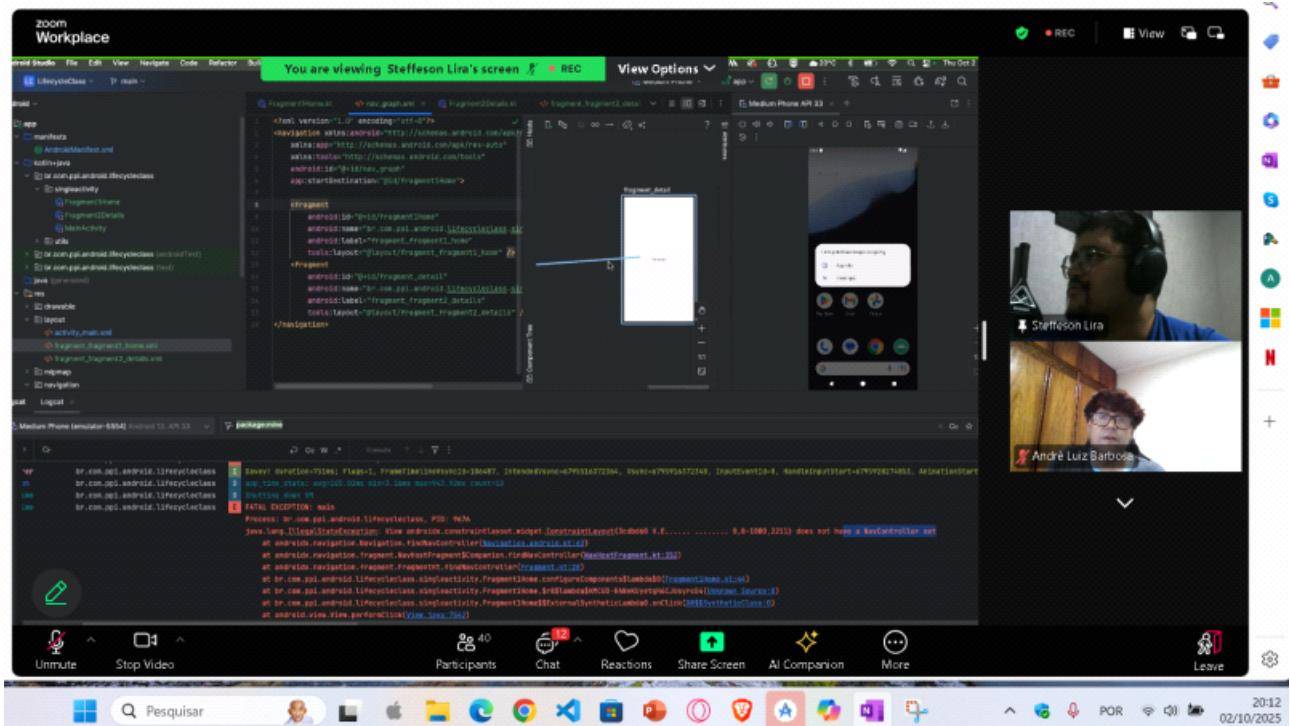


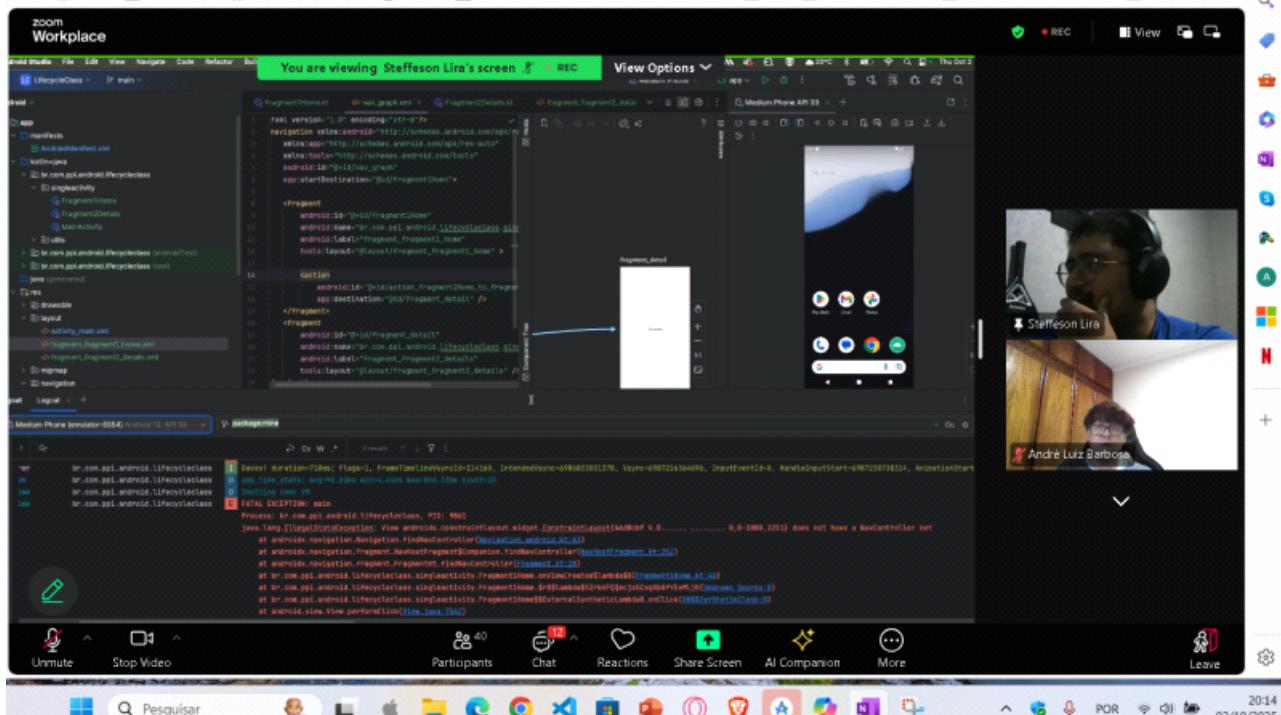
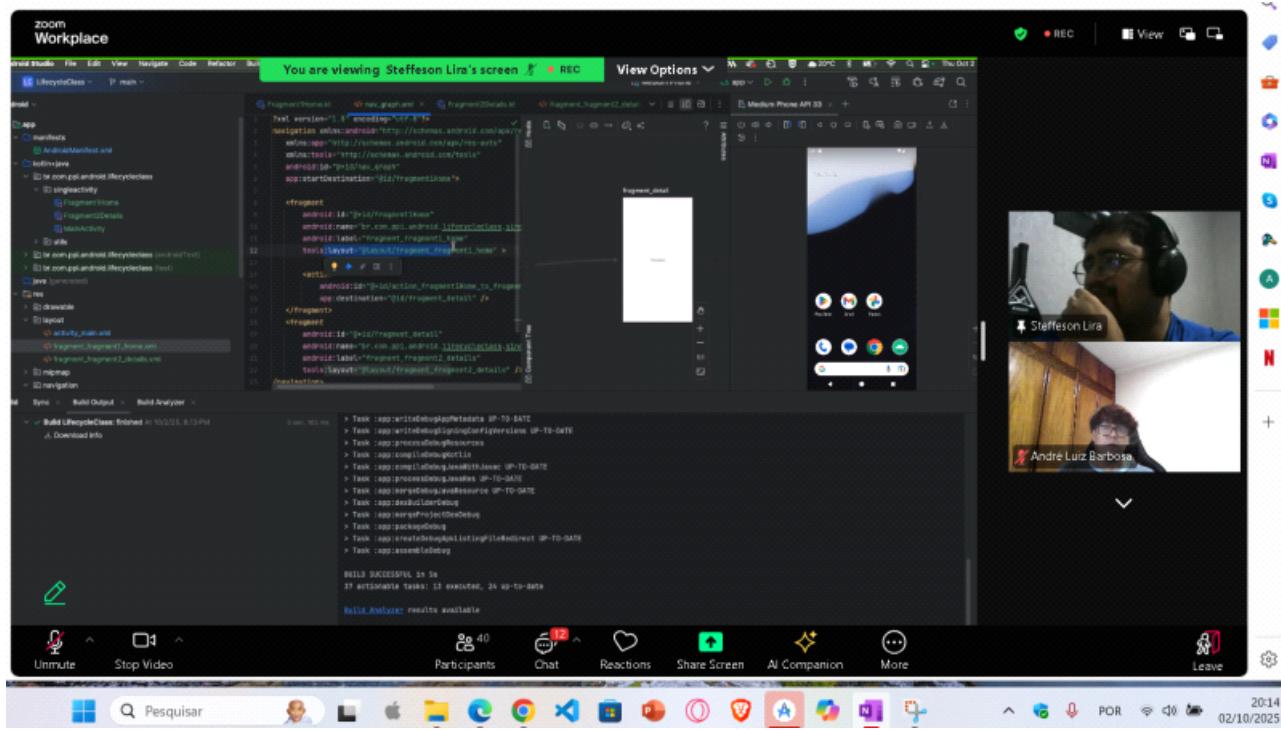




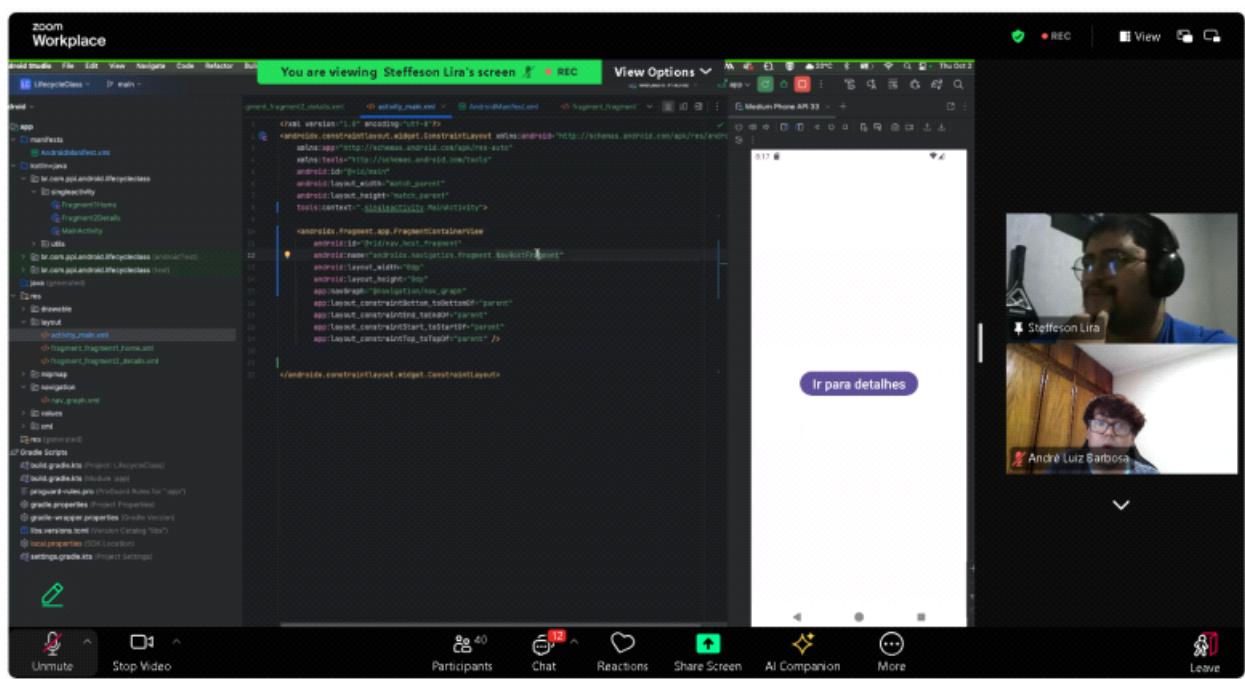
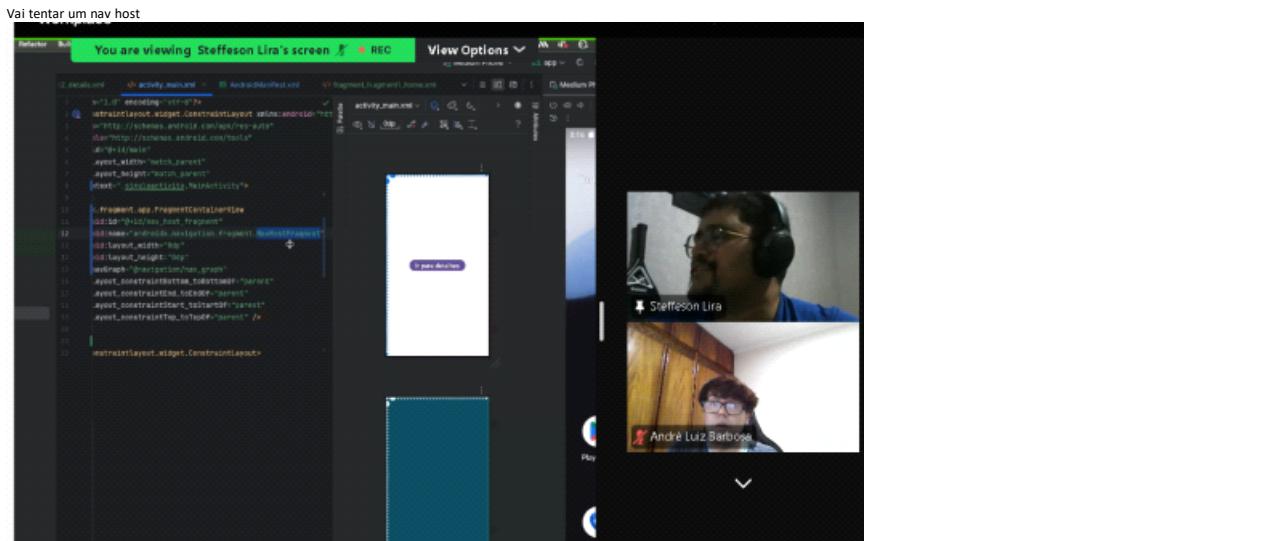
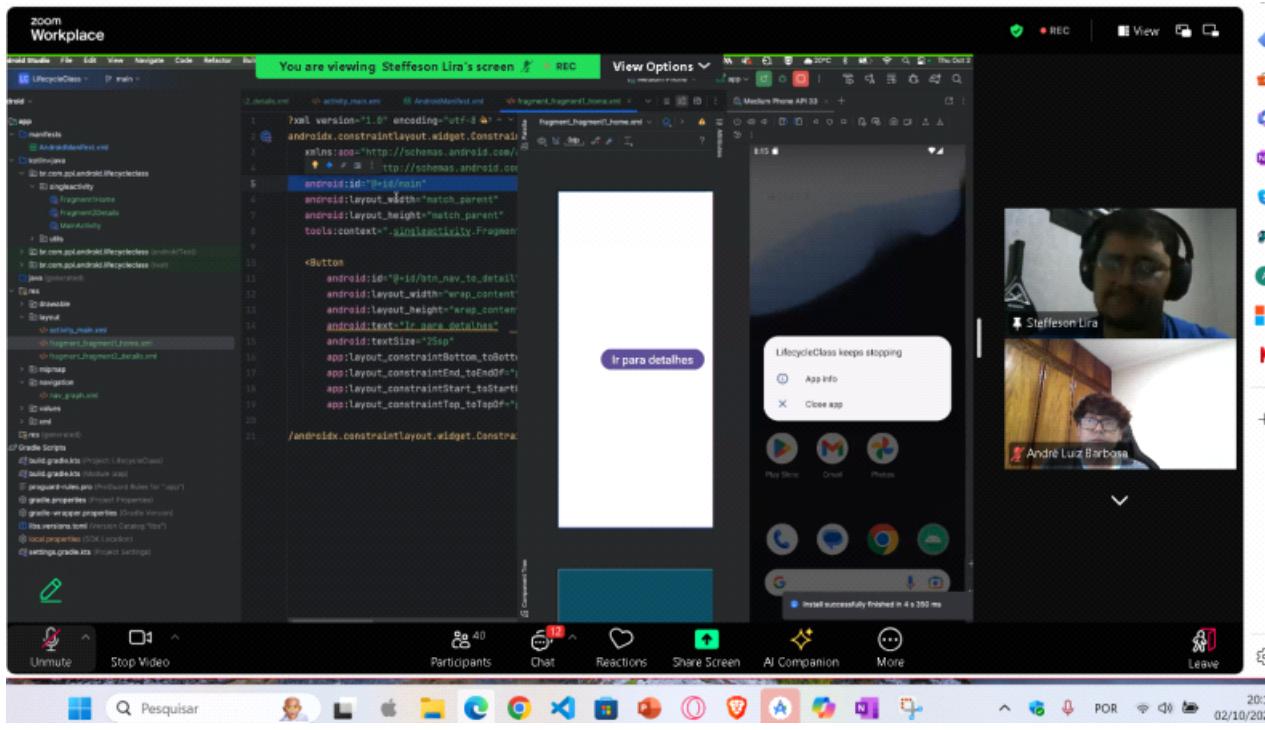


Até aqui não descobre o que fez dar crash no app





Id faltando no 2?



Erro identificado  
Sim, Andre — agora que vi o Logcat, o erro está bem claro:

## Erro identificado:

java.lang.IllegalStateException: NavController.setGraph() must be called before calling getGraph()

## O que isso significa?

Você está tentando acessar o NavController.getGraph() **antes de ter chamado setGraph()** — ou seja, o gráfico de navegação (navGraph) ainda não foi atribuído ao NavController quando você tentou usá-lo.

## Como corrigir

Você precisa garantir que o NavController tenha um gráfico de navegação antes de chamar getGraph() ou qualquer operação que dependa dele. Aqui está um exemplo de como fazer isso corretamente:

```
val navController = findNavController(R.id.nav_host_fragment)
navController.setGraph(R.navigation.nav_graph)
```

Depois disso, você pode chamar:

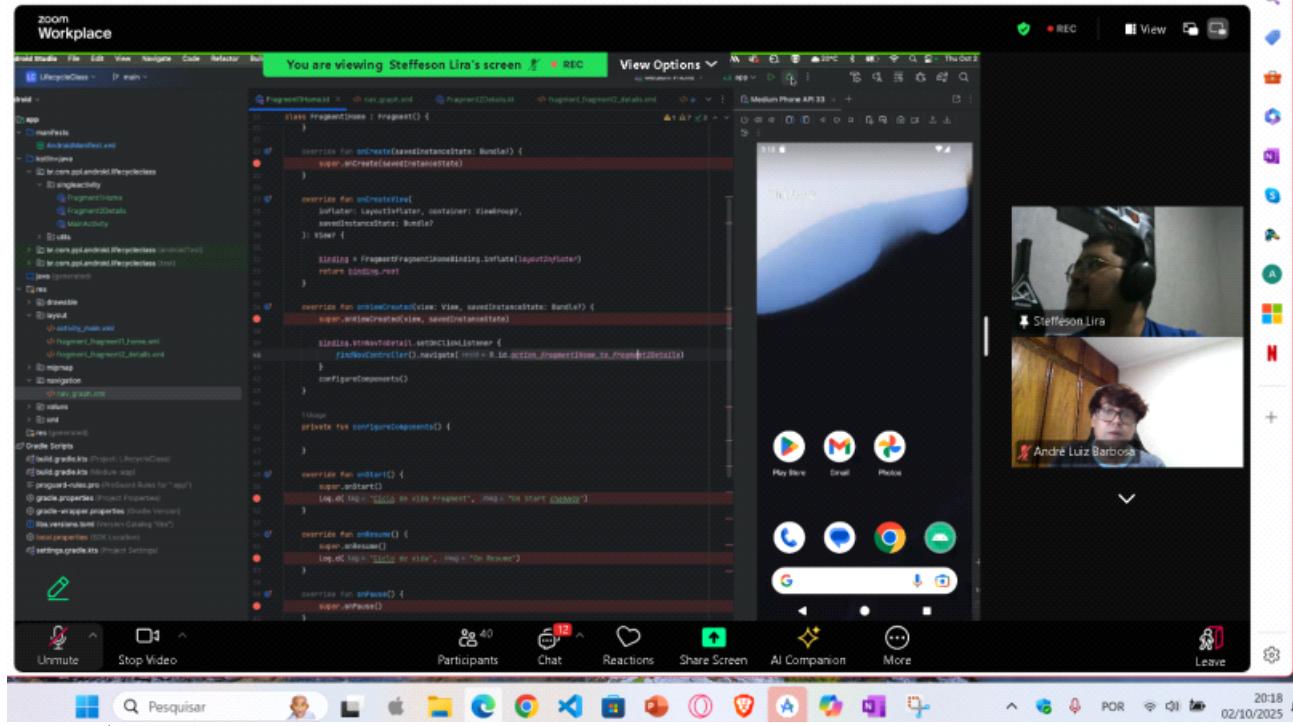
```
val graph = navController.graph
```

## Dica extra

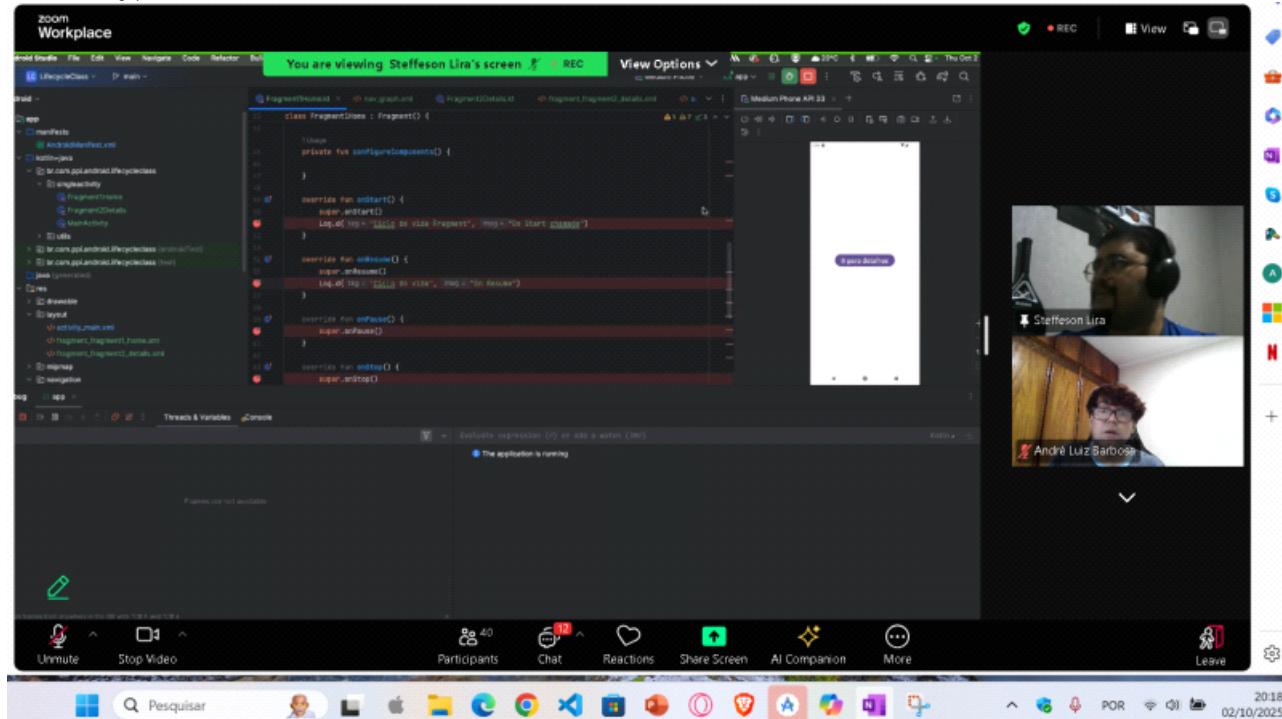
Se você estiver usando NavHostFragment no XML com app:navGraph="@navigation/nav\_graph", o gráfico já é atribuído automaticamente. Nesse caso, você **não precisa** chamar setGraph() manualmente — mas precisa garantir que o NavController esteja sendo acessado **depois** que a Activity ou Fragment foi completamente inicializado.

Se quiser, posso revisar o trecho exato onde você está usando o NavController e te mostrar como ajustar. Só me mandar o código que está na Viewpager2Class na linha 18 (como o Logcat indicou). Vamos resolver isso juntos!

De <<https://copilot.microsoft.com/chats/e19yspcn99lg5j5ca2szl>>



Vai mostrar com log apenas na 2



Tirou da pilha

You are viewing Steffeson Lira's screen REC View Options M 2020 Thu Oct 2

FragmentHome.java

```
class FragmentHome : Fragment() {
    ...
    override fun onPause() {
        super.onPause()
    }
    ...
    override fun onStart() {
        super.onStart()
    }
    ...
    override fun onDestroyView() {
        super.onDestroyView()
    }
    ...
    override fun onDestroy() {
        super.onDestroy()
    }
    ...
    override fun onDetach() {
        super.onDetach()
    }
}
```

Medium Phone API 33

Steffeson Lira

André Luiz Barbosa

Vai tirar os em azul:

You are viewing Steffeson Lira's screen REC View Options M 2020 Thu Oct 2

FragmentHome.java

```
class FragmentHome : Fragment() {
    ...
    override fun onPause() {
        super.onPause()
    }
    ...
    override fun onStart() {
        super.onStart()
    }
    ...
    override fun onDestroyView() {
        super.onDestroyView()
    }
    ...
    override fun onDestroy() {
        super.onDestroy()
    }
    ...
    override fun onDetach() {
        super.onDetach()
    }
}
```

Medium Phone API 33

Steffeson Lira

André Luiz Barbosa

Vai tirar os em azul:

You are viewing Steffeson Lira's screen REC View Options M 2020 Thu Oct 2

FragmentHome.java

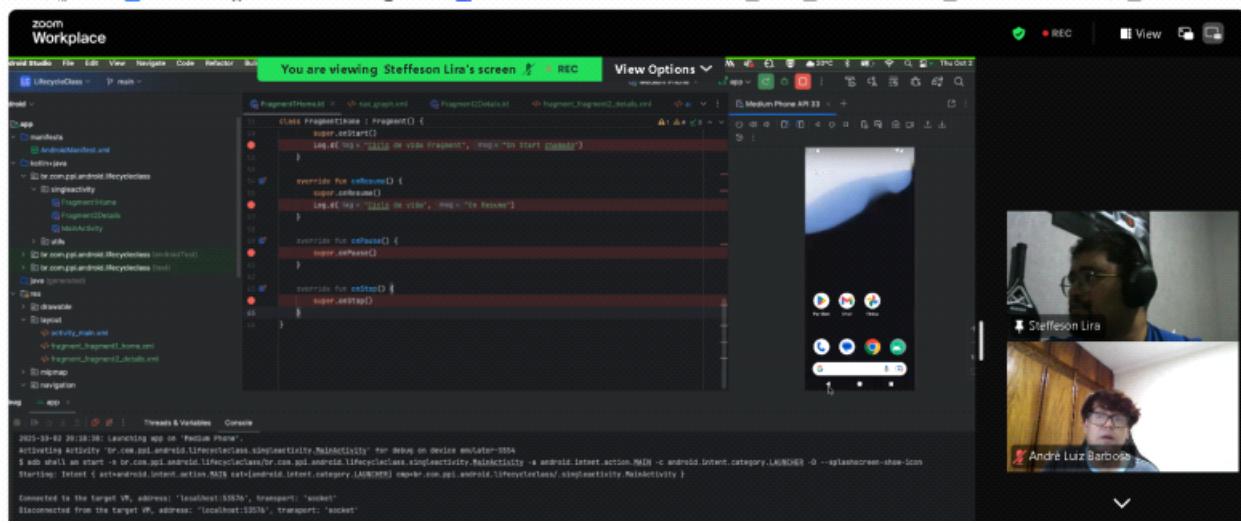
```
class FragmentHome : Fragment() {
    ...
    override fun onPause() {
        super.onPause()
    }
    ...
    override fun onStart() {
        super.onStart()
    }
    ...
    override fun onDestroyView() {
        super.onDestroyView()
    }
    ...
    override fun onDestroy() {
        super.onDestroy()
    }
    ...
    override fun onDetach() {
        super.onDetach()
    }
}
```

Medium Phone API 33

Steffeson Lira

André Luiz Barbosa

Para não fechar

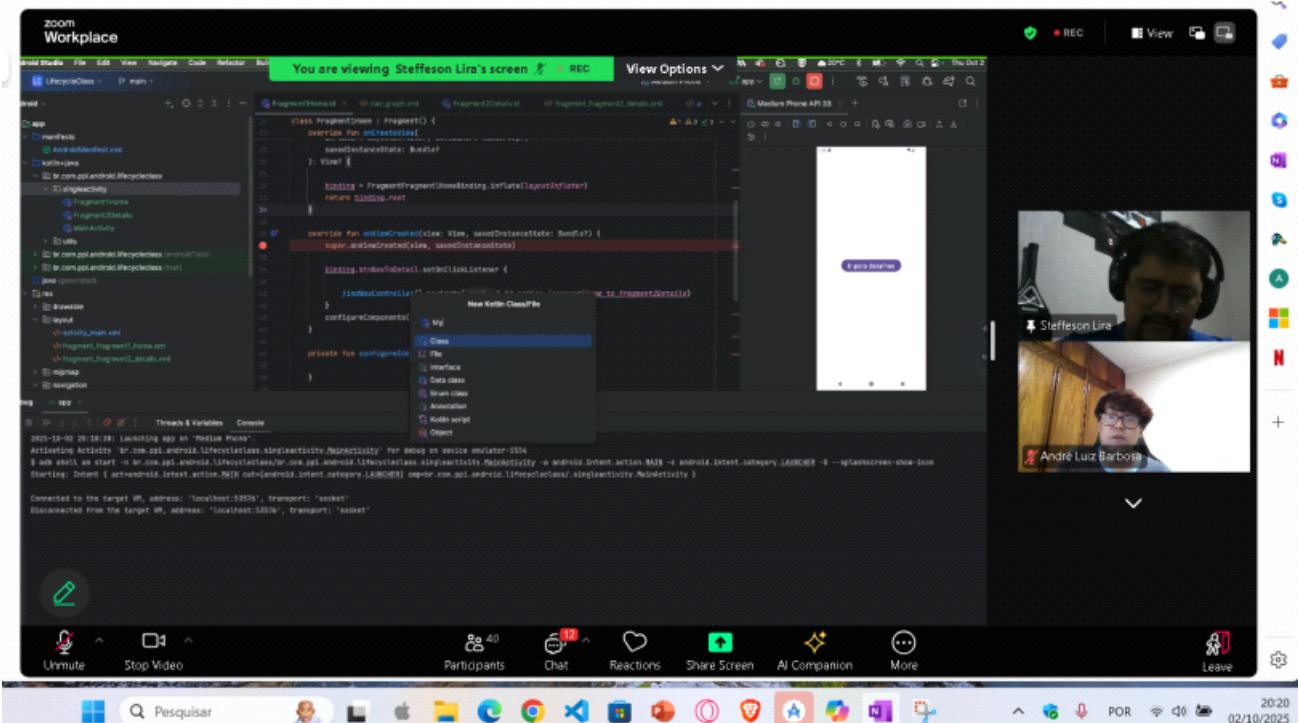


Não é isto não

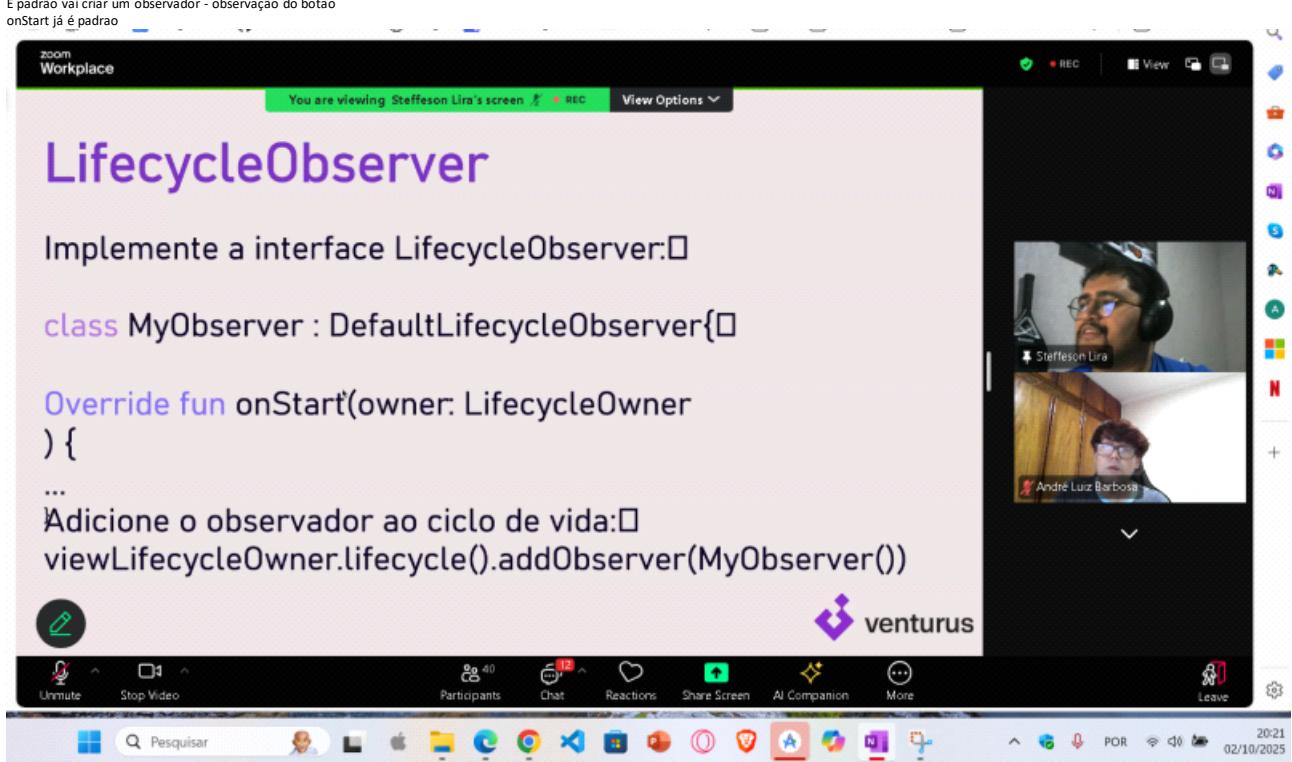
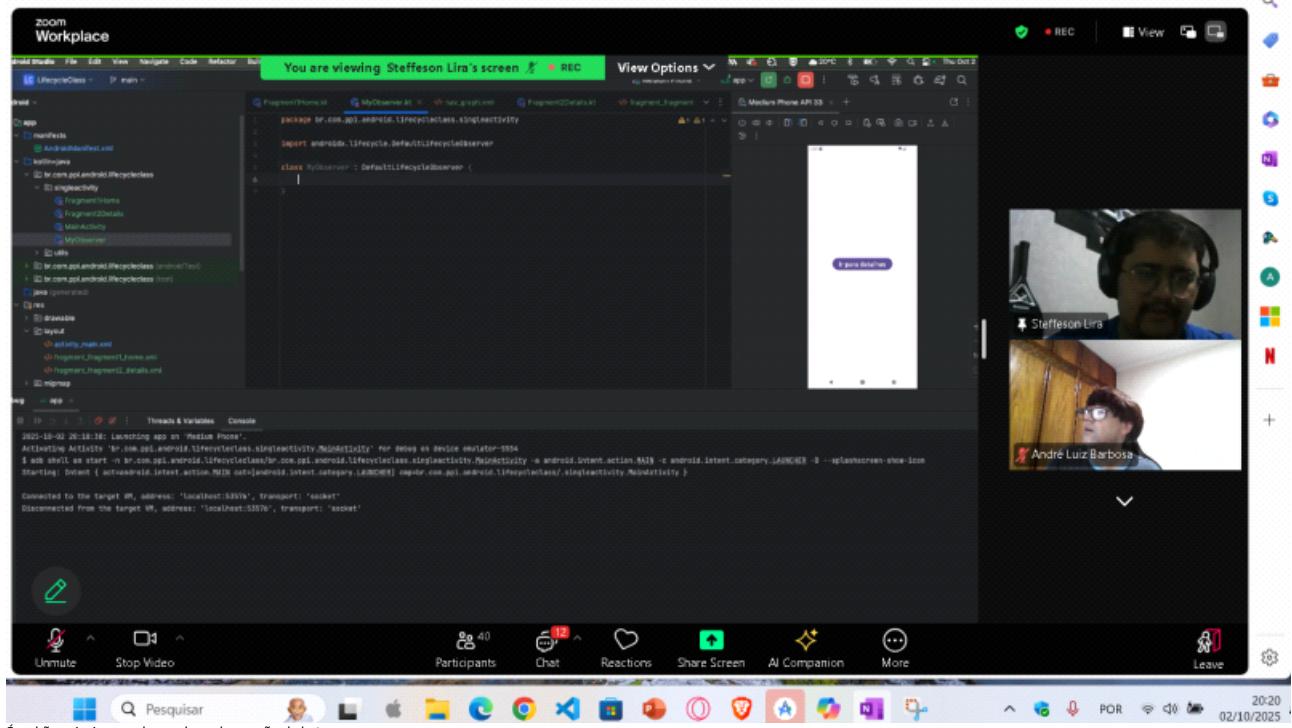
Tira da pilha com o navigate

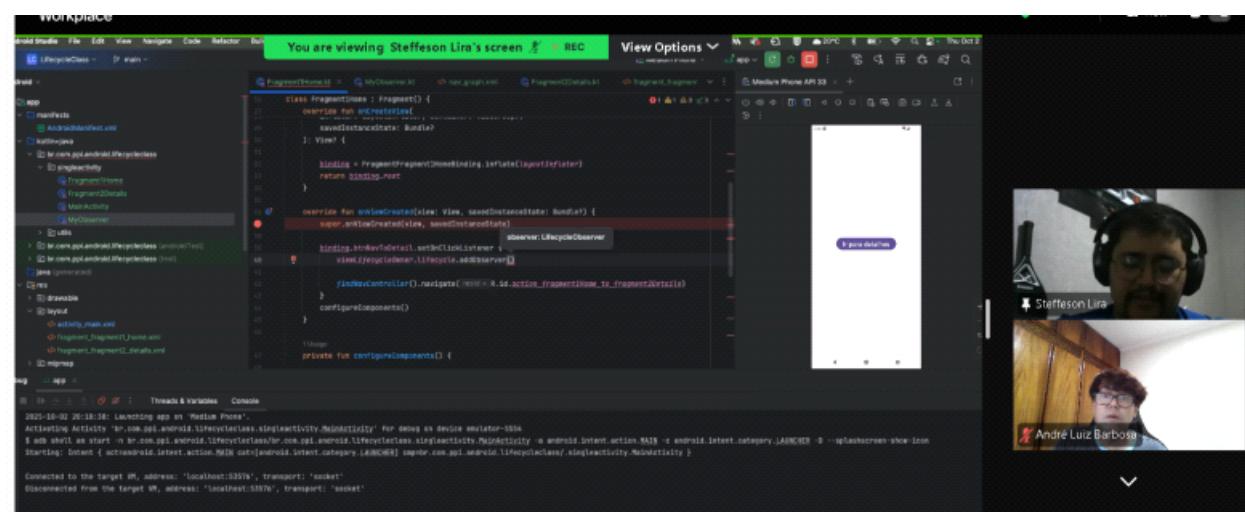
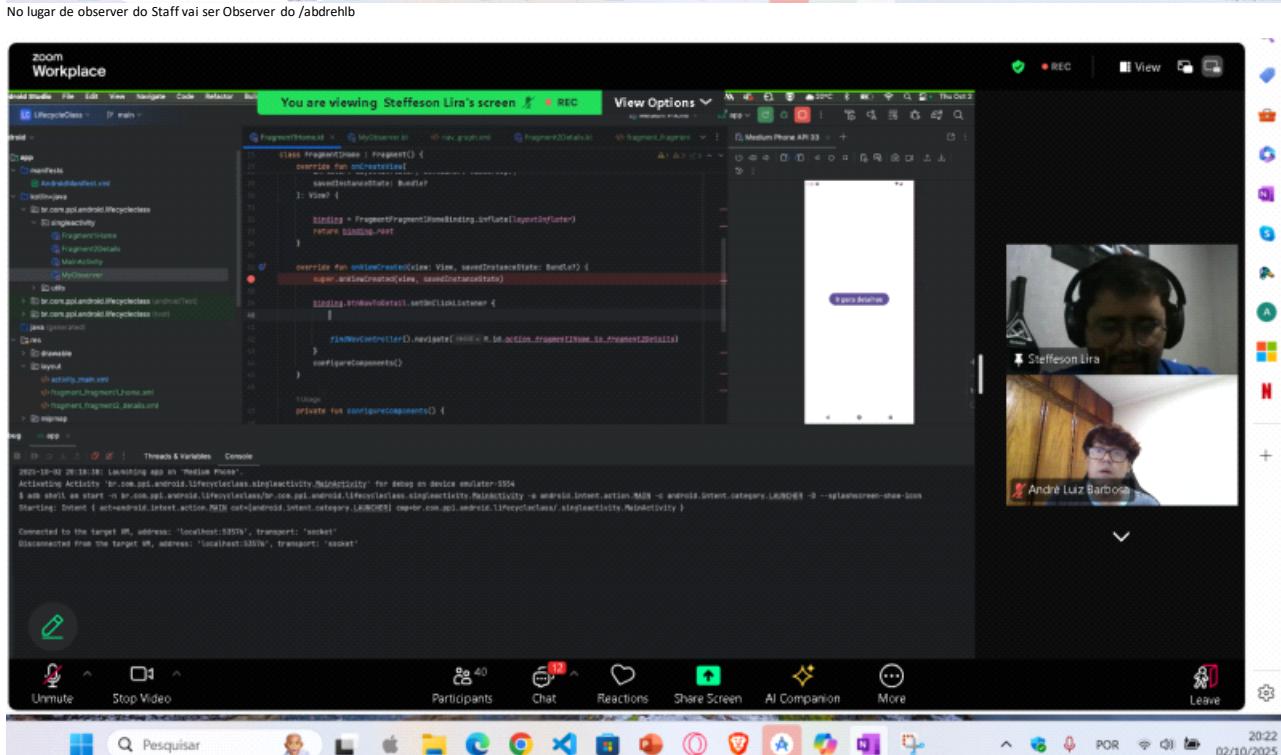
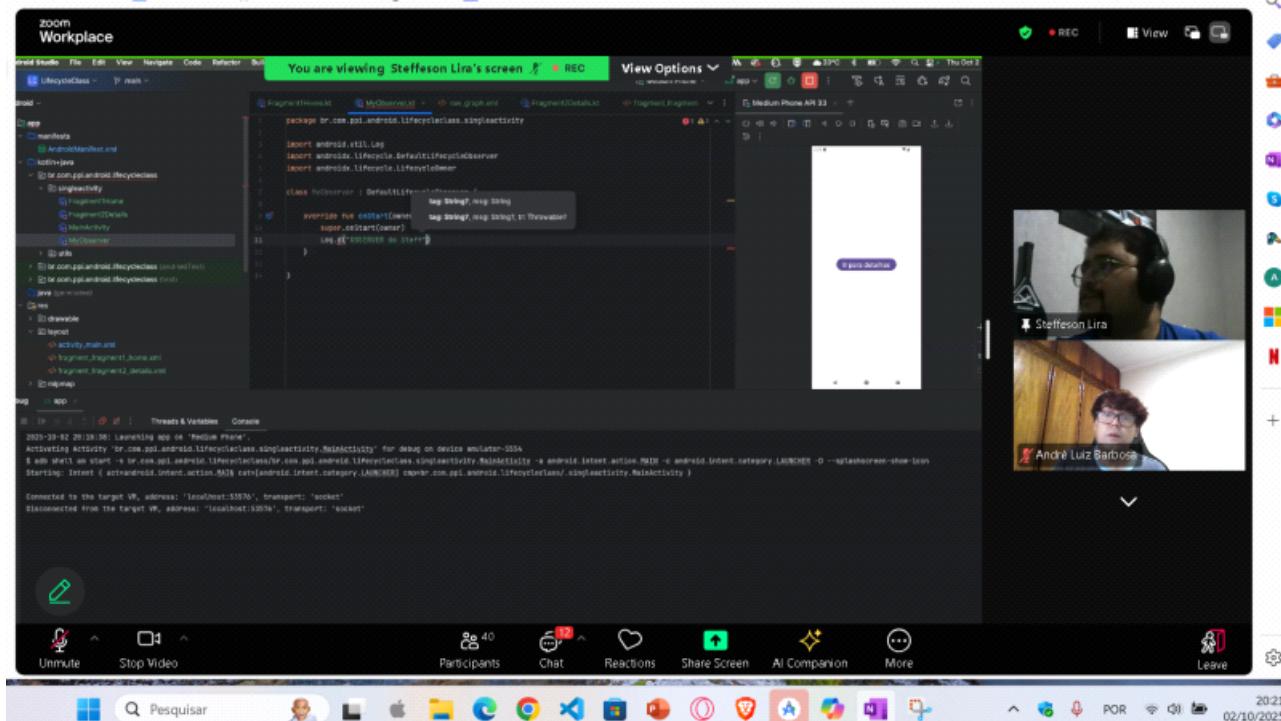
Vai criar a parte do ib

Nova classe



My observer  
Ver no slide observer





Vai criar myobserver abre e fecha

You are viewing Steffeson Lira's screen REC

Java code (LifecycleClass.java) showing the implementation of the FragmentObserver interface:

```
class FragmentObserver : Fragment() {
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        binding = FragmentHomeBinding.inflate(layoutInflater)
        return binding.root
    }

    override fun onActivityCreated(view: View, savedInstanceState: Bundle?) {
        super.onActivityCreated(savedInstanceState)
        binding.lifecycleDetail.setLifecycleListener {
            viewLifecycleOwner.lifecycle.addObserver(this)
        }
        findNavController().navigate(R.id.action_fragmentHome_to_fragmentDetails)
    }

    private fun configureComponents() {
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "LifecycleObserver onStart")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "LifecycleObserver onResume")
    }

    override fun onPause() {
        super.onPause()
        Log.d(TAG, "LifecycleObserver onPause")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "LifecycleObserver onStop")
    }

    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "LifecycleObserver onDestroy")
    }
}
```

Logcat output:

```
2022-10-02 20:18:38: Launching app on 'Medium Phone'.
Activating Activity 'br.com.spd.android.lifecycleguide.singleactivity.MainActivity' for debug on device emulator-5554.
$ adb shell start -n br.com.spd.android.lifecycleguide.br.com.spd.android.lifecycleguide.singleactivity.MainActivity -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -0 --splashscreen showIcon
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=br.com.spd.android.lifecycleguide/.singleactivity.MainActivity }
```

You are viewing Steffeson Lira's screen REC

Java code (LifecycleClass.java) showing the implementation of the FragmentObserver interface:

```
class LifecycleObserver : DefaultLifecycleObserver {
    override fun onStateChanged(owner: LifecycleOwner) {
        super.onStateChanged(owner)
        Log.d(TAG, "onStateChanged $owner, tag = ${owner.tag}, state = ${owner.currentState}")
    }
}
```

Logcat output:

```
2022-10-02 20:18:38: Launching app on 'Medium Phone'.
Activating Activity 'br.com.spd.android.lifecycleguide.singleactivity.MainActivity' for debug on device emulator-5554.
$ adb shell start -n br.com.spd.android.lifecycleguide.br.com.spd.android.lifecycleguide.singleactivity.MainActivity -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -0 --splashscreen showIcon
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=br.com.spd.android.lifecycleguide/.singleactivity.MainActivity }
```

Excuta

You are viewing Steffeson Lira's screen REC

Java code (LifecycleClass.java) showing the implementation of the FragmentObserver interface:

```
class FragmentObserver : Fragment() {
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        binding = FragmentHomeBinding.inflate(layoutInflater)
        return binding.root
    }

    override fun onActivityCreated(view: View, savedInstanceState: Bundle?) {
        super.onActivityCreated(savedInstanceState)
        binding.lifecycleDetail.setLifecycleListener {
            viewLifecycleOwner.lifecycle.addObserver(this)
        }
        findNavController().navigate(R.id.action_fragmentHome_to_fragmentDetails)
    }

    private fun configureComponents() {
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "LifecycleObserver onStart")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "LifecycleObserver onResume")
    }

    override fun onPause() {
        super.onPause()
        Log.d(TAG, "LifecycleObserver onPause")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "LifecycleObserver onStop")
    }

    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "LifecycleObserver onDestroy")
    }
}
```

Logcat output:

```
2022-10-02 20:22:32: Launching app on 'Medium Phone'.
Activating Activity 'br.com.spd.android.lifecycleguide.singleactivity.MainActivity' for debug on device emulator-5554.
$ adb shell start -n br.com.spd.android.lifecycleguide.br.com.spd.android.lifecycleguide.singleactivity.MainActivity -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -0 --splashscreen showIcon
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=br.com.spd.android.lifecycleguide/.singleactivity.MainActivity }
```

Deu clic

You are viewing Steffeson Lira's screen REC

Java code (LifecycleClass.java) showing the implementation of the FragmentObserver interface:

```
class FragmentObserver : Fragment() {
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        binding = FragmentHomeBinding.inflate(layoutInflater)
        binding.lifecycleDetail.setLifecycleListener {
            viewLifecycleOwner.lifecycle.addObserver(this)
        }
        findNavController().navigate(R.id.action_fragmentHome_to_fragmentDetails)
    }

    private fun configureComponents() {
    }

    override fun onStart() {
        super.onStart()
        Log.d(TAG, "LifecycleObserver onStart")
    }

    override fun onResume() {
        super.onResume()
        Log.d(TAG, "LifecycleObserver onResume")
    }

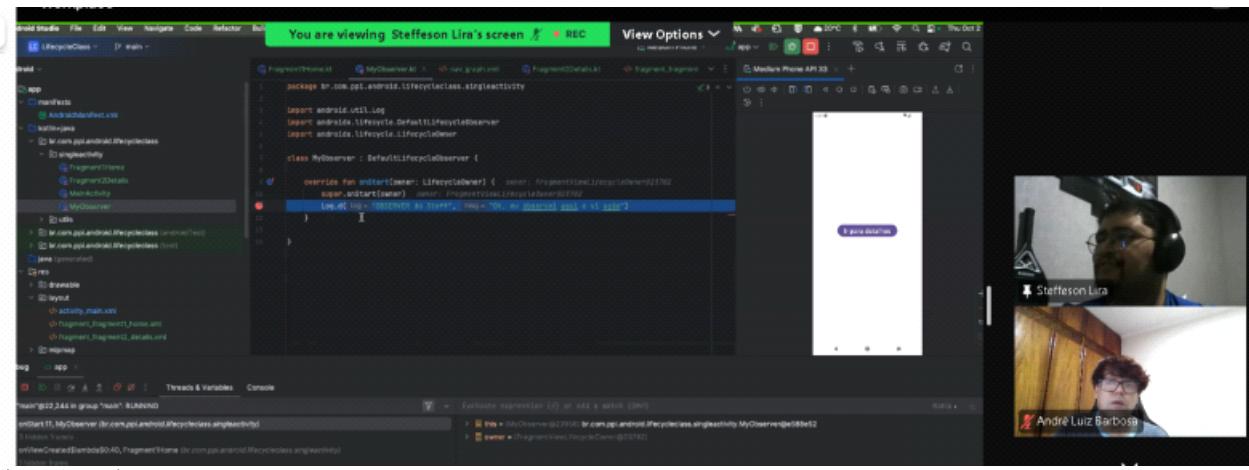
    override fun onPause() {
        super.onPause()
        Log.d(TAG, "LifecycleObserver onPause")
    }

    override fun onStop() {
        super.onStop()
        Log.d(TAG, "LifecycleObserver onStop")
    }

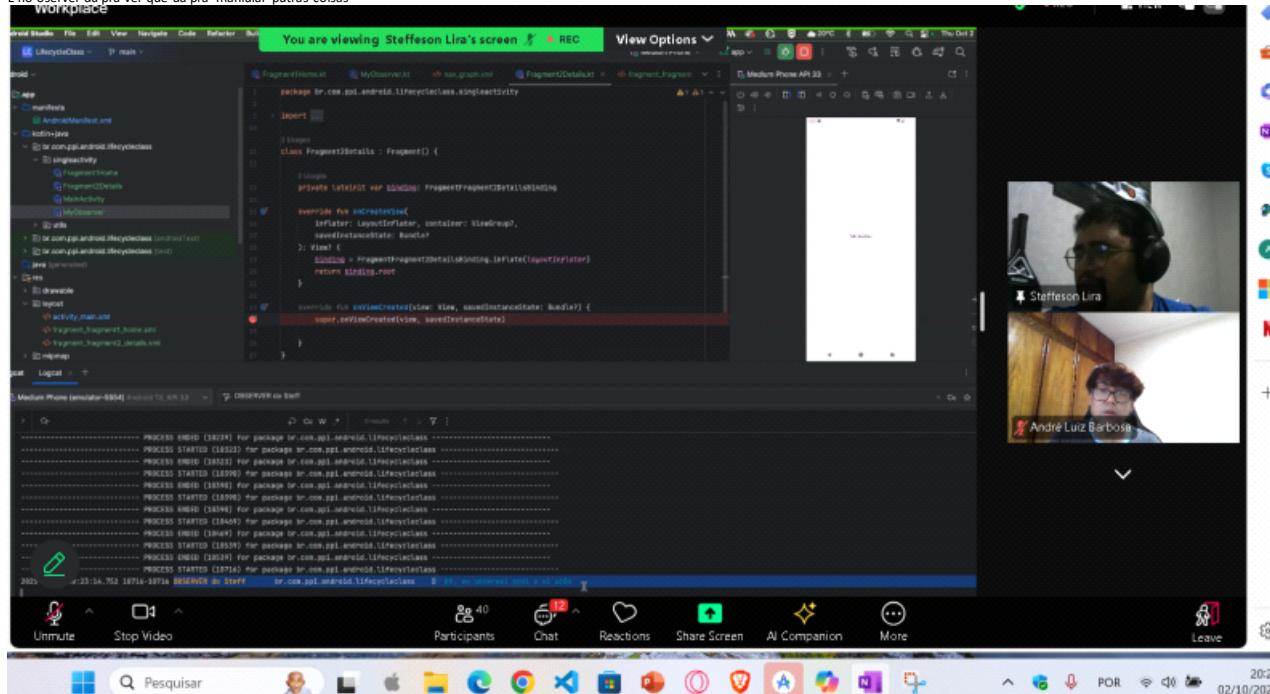
    override fun onDestroy() {
        super.onDestroy()
        Log.d(TAG, "LifecycleObserver onDestroy")
    }
}
```

Logcat output:

```
2022-10-02 20:22:32: Launching app on 'Medium Phone'.
Activating Activity 'br.com.spd.android.lifecycleguide.singleactivity.MainActivity' for debug on device emulator-5554.
$ adb shell start -n br.com.spd.android.lifecycleguide.br.com.spd.android.lifecycleguide.singleactivity.MainActivity -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -0 --splashscreen showIcon
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=br.com.spd.android.lifecycleguide/.singleactivity.MainActivity }
```



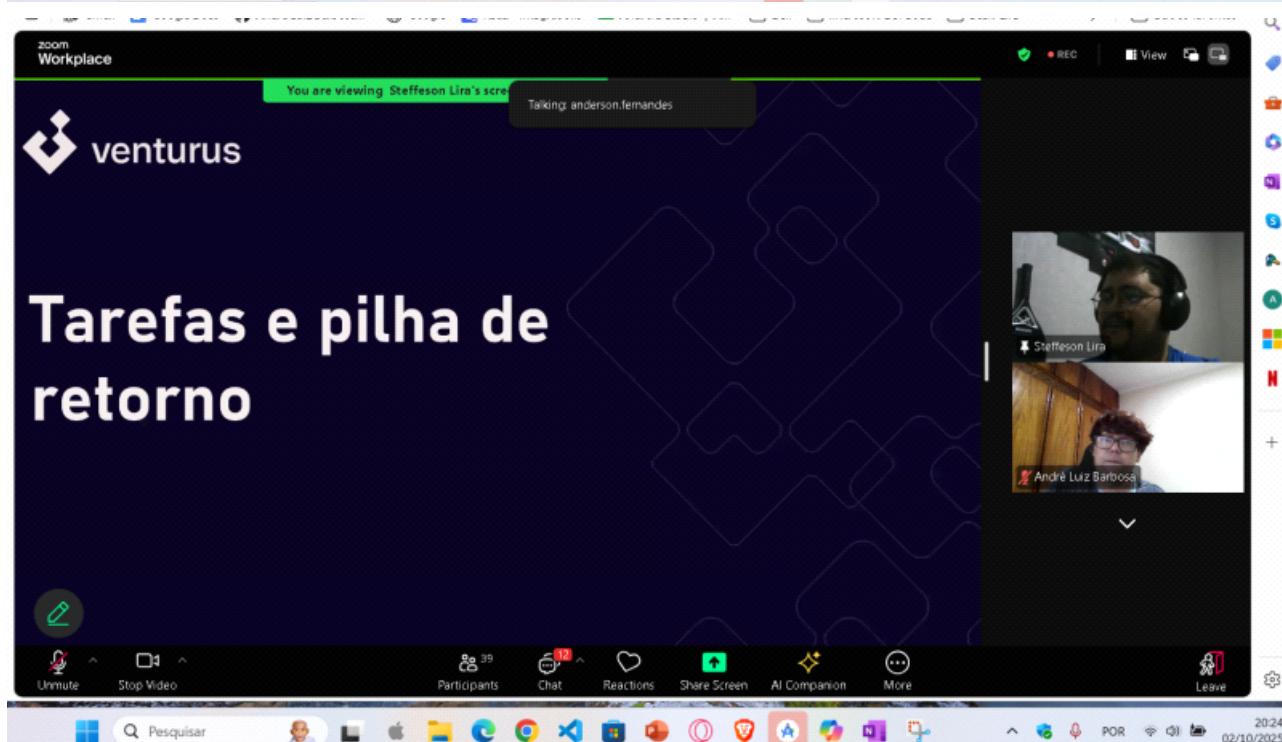
Clicou e açãoou o observer  
E no observer dá pra ver que dá pra manupular putras coisas



Unmute Stop Video Participants Chat Reactions Share Screen AI Companion More Leave

PROCESS\_KILLED (18319) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18322) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18323) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18389) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18390) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18391) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18394) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18449) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18449) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18859) For package br.com.gpt.android.lifecycletest  
PROCESS\_ENDED (18859) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18974) For package br.com.gpt.android.lifecycletest  
PROCESS\_ENDED (18974) For package br.com.gpt.android.lifecycletest

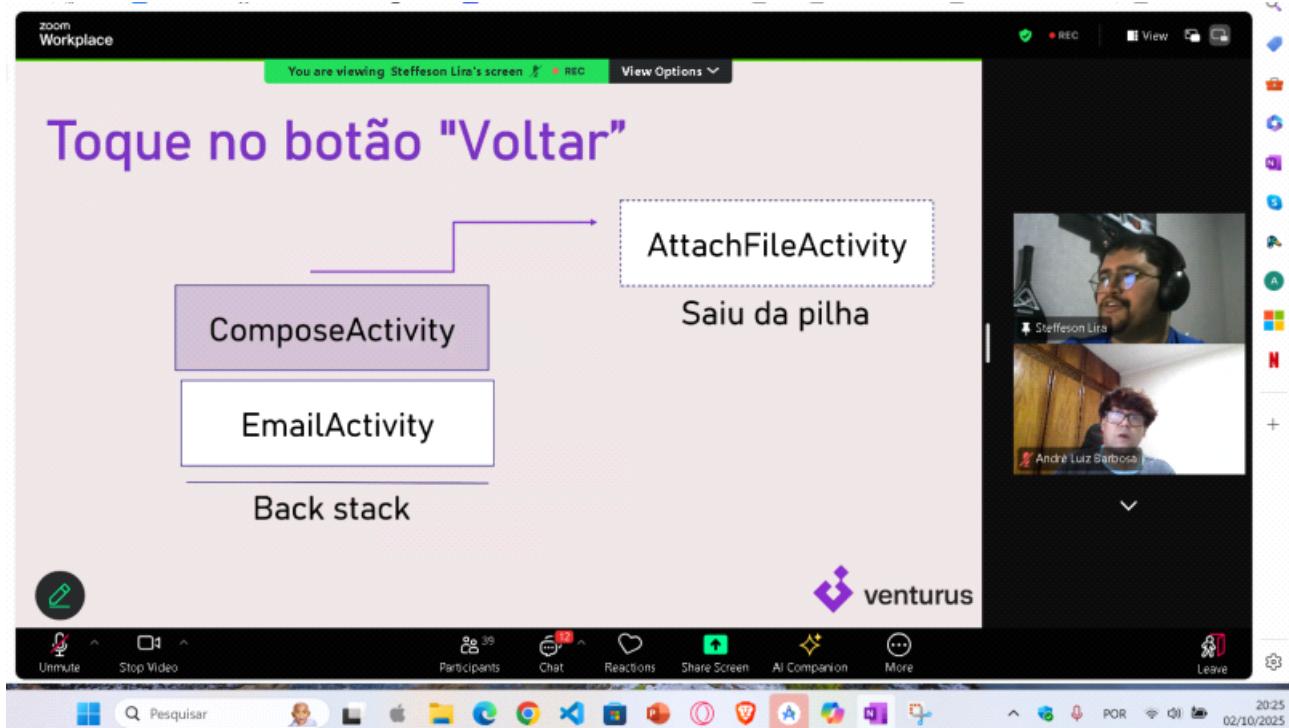
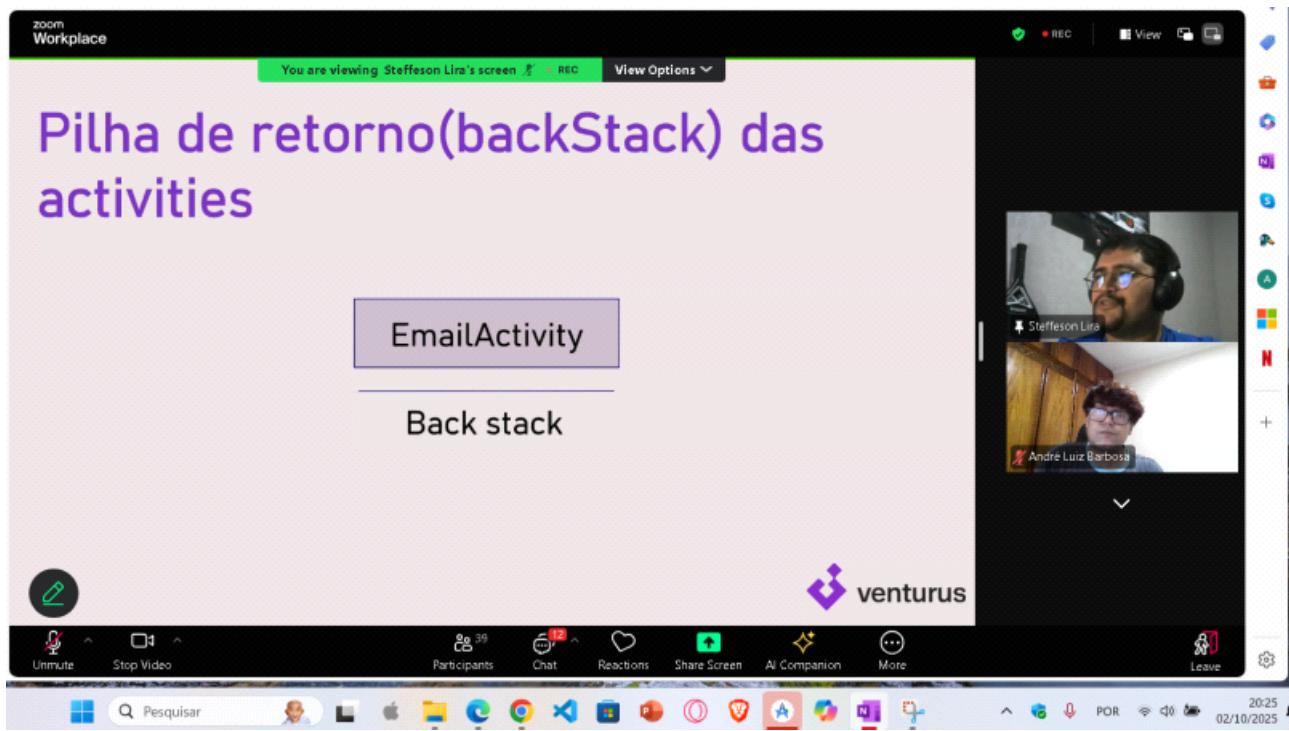
2023 02/10/2025

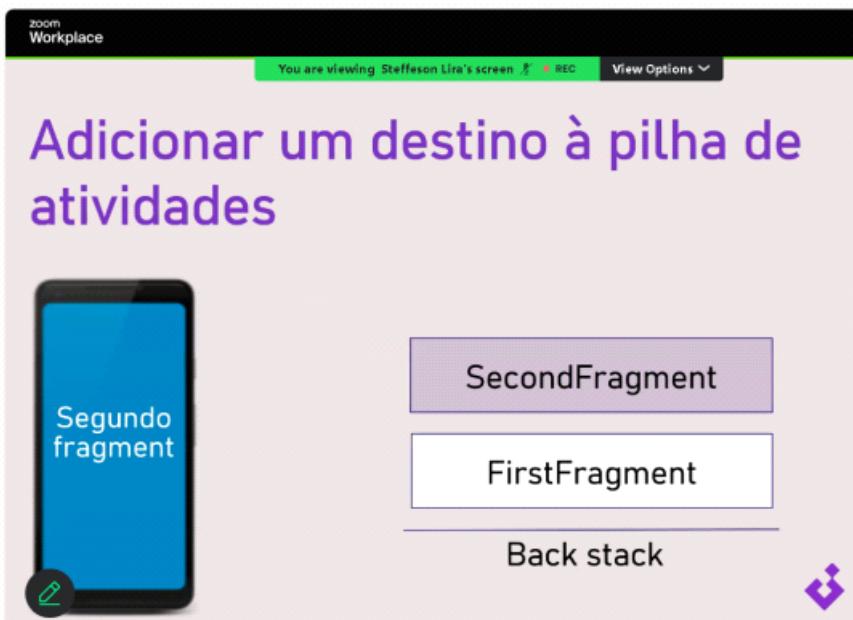


Unmute Stop Video Participants Chat Reactions Share Screen AI Companion More Leave

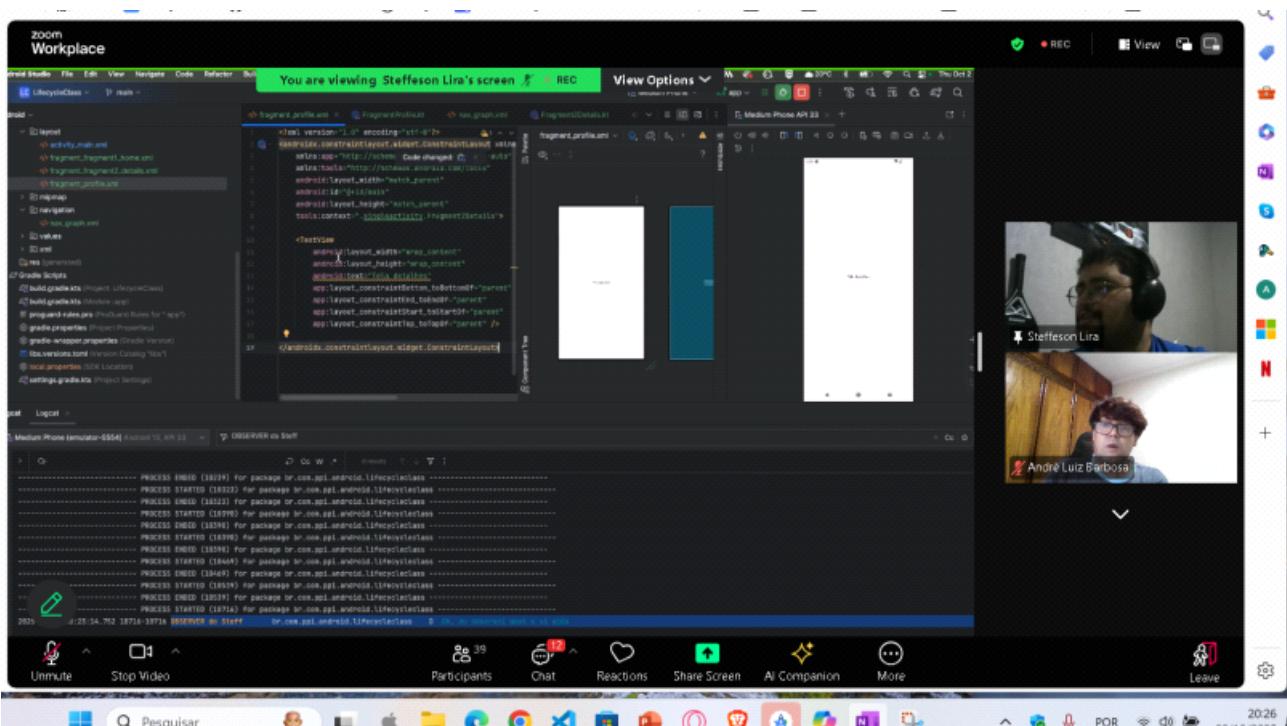
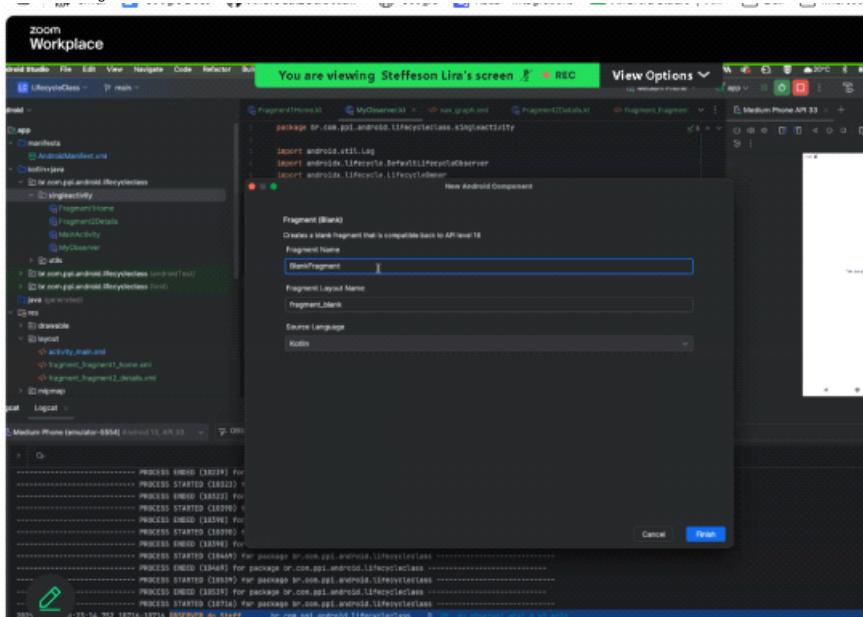
PROCESS\_KILLED (18319) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18322) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18323) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18389) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18390) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18391) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18394) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18449) For package br.com.gpt.android.lifecycletest  
PROCESS\_KILLED (18449) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18859) For package br.com.gpt.android.lifecycletest  
PROCESS\_ENDED (18859) For package br.com.gpt.android.lifecycletest  
PROCESS\_STARTED (18974) For package br.com.gpt.android.lifecycletest  
PROCESS\_ENDED (18974) For package br.com.gpt.android.lifecycletest

2024 02/10/2025

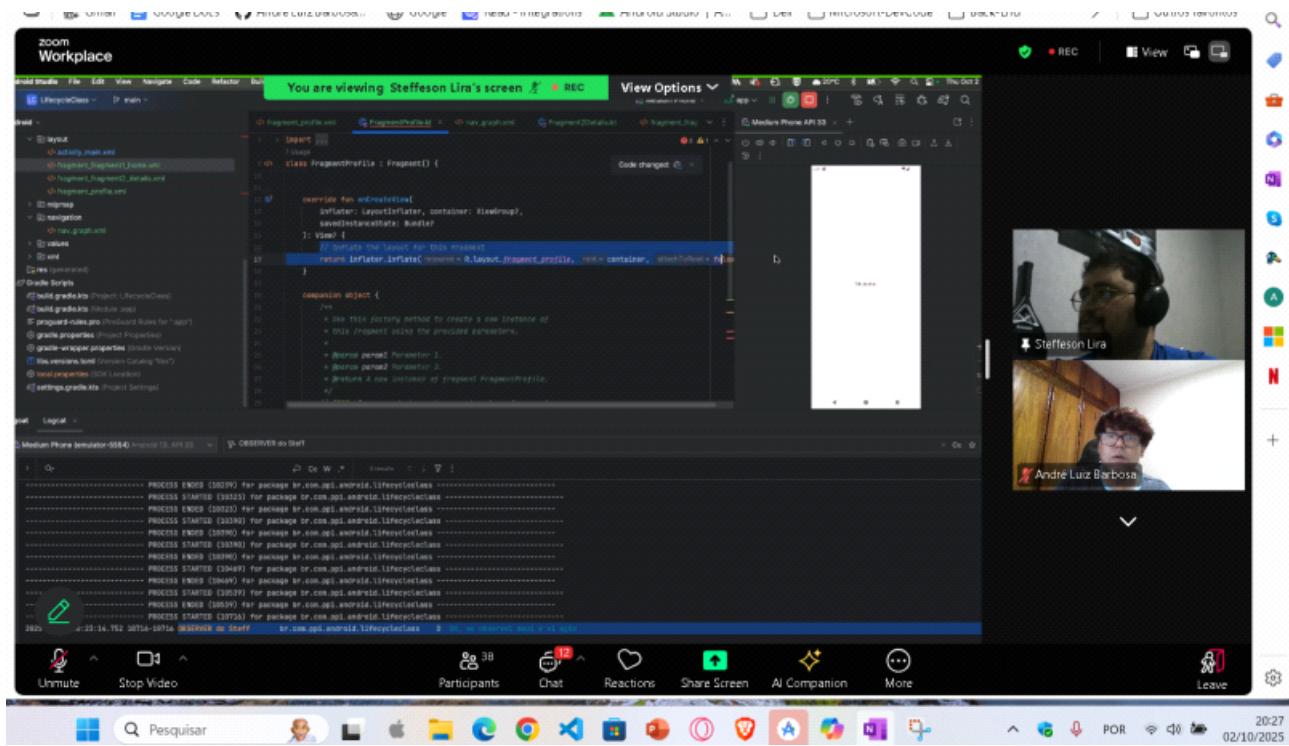
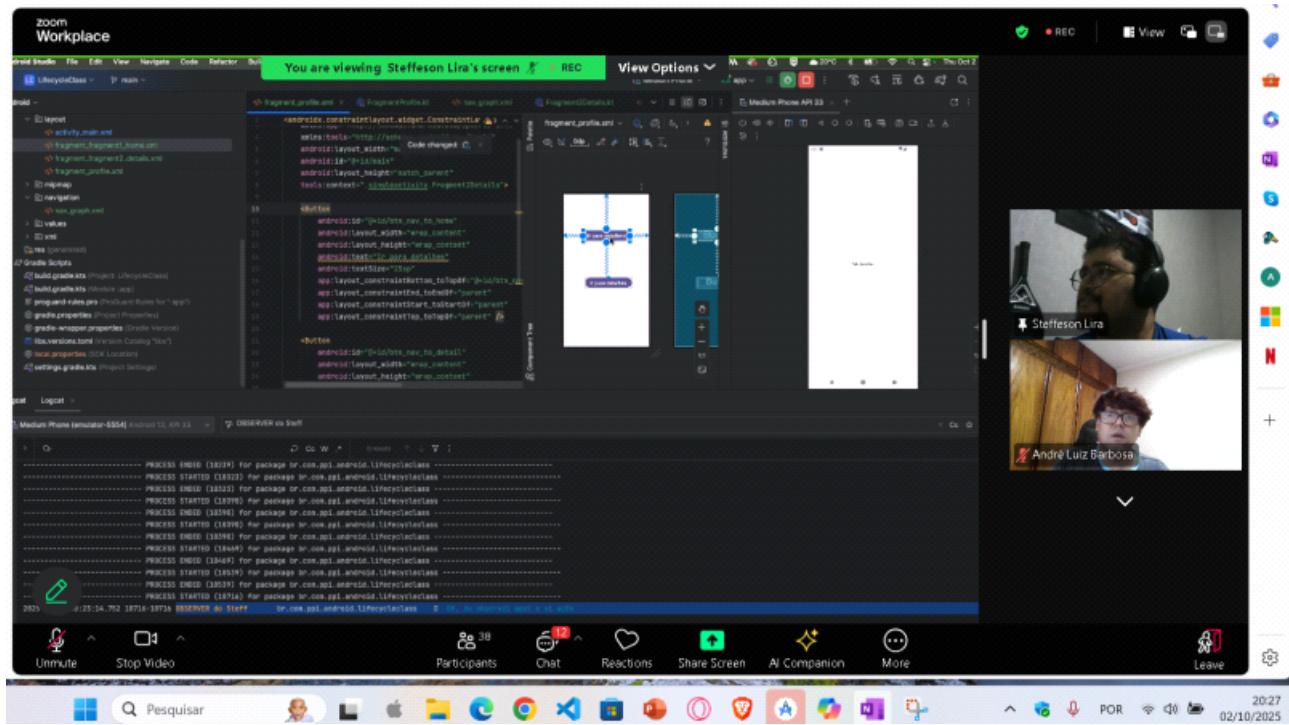




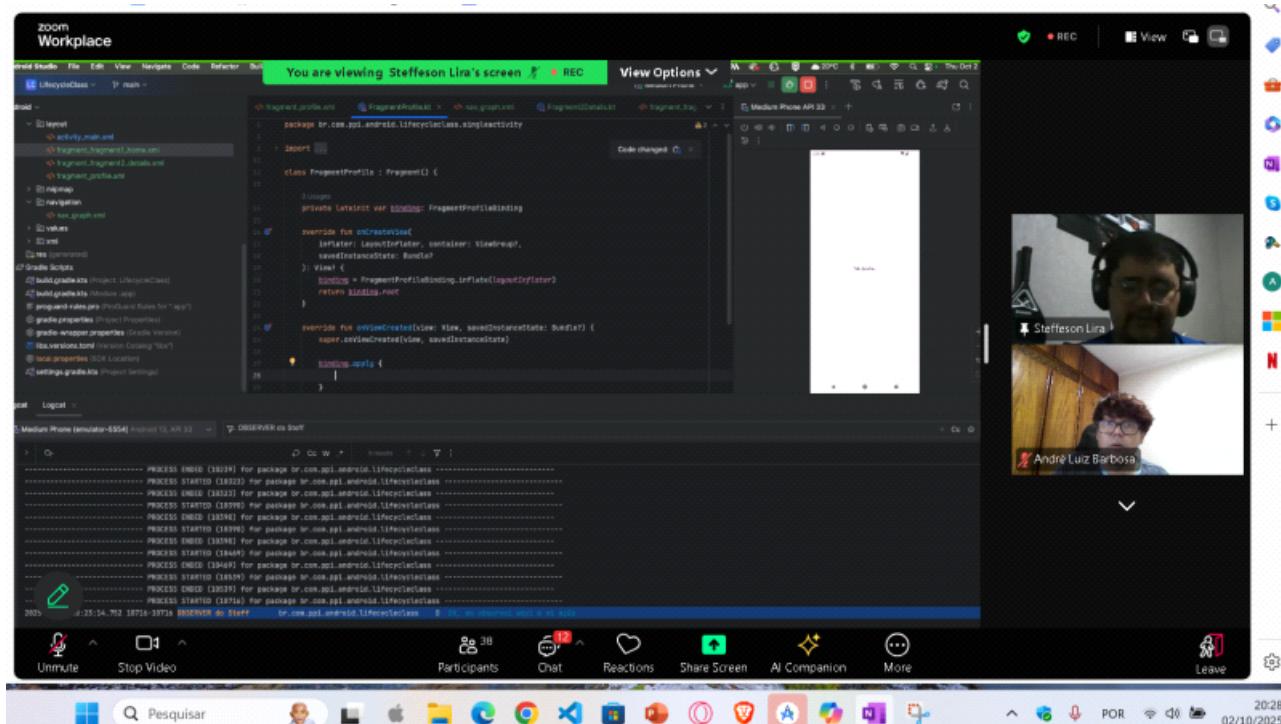
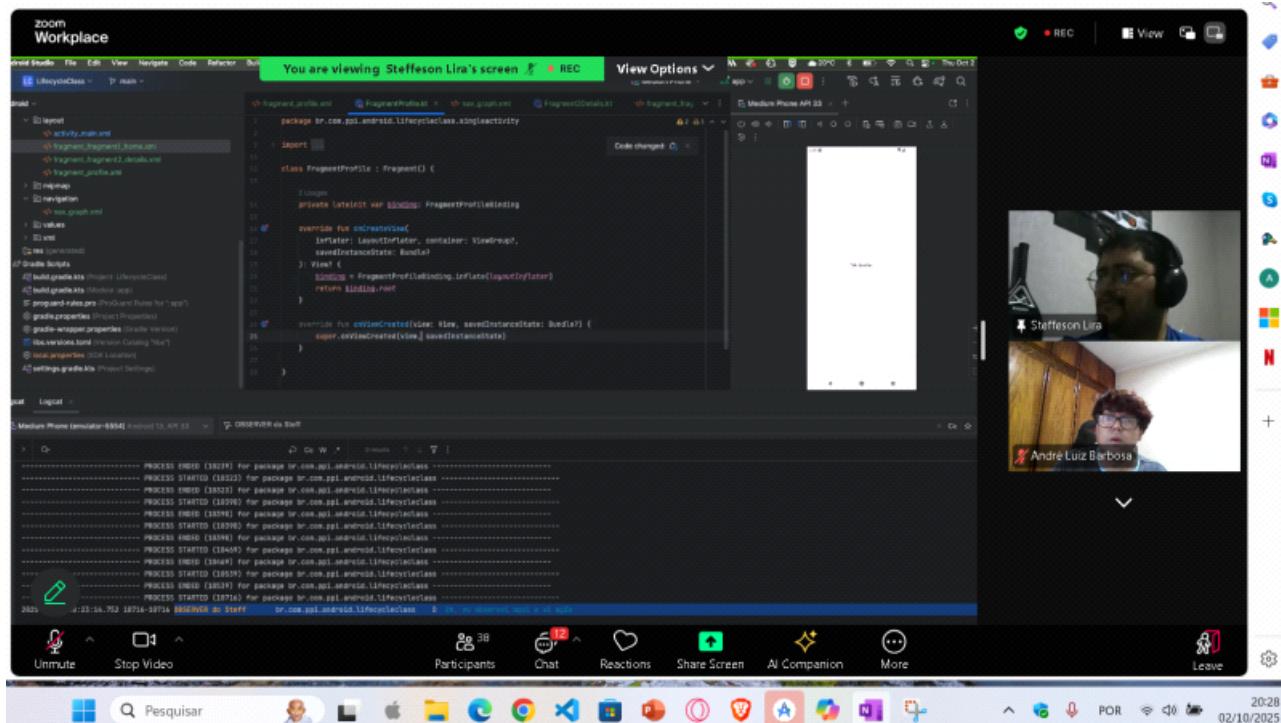
Vai criar o 3º fragment



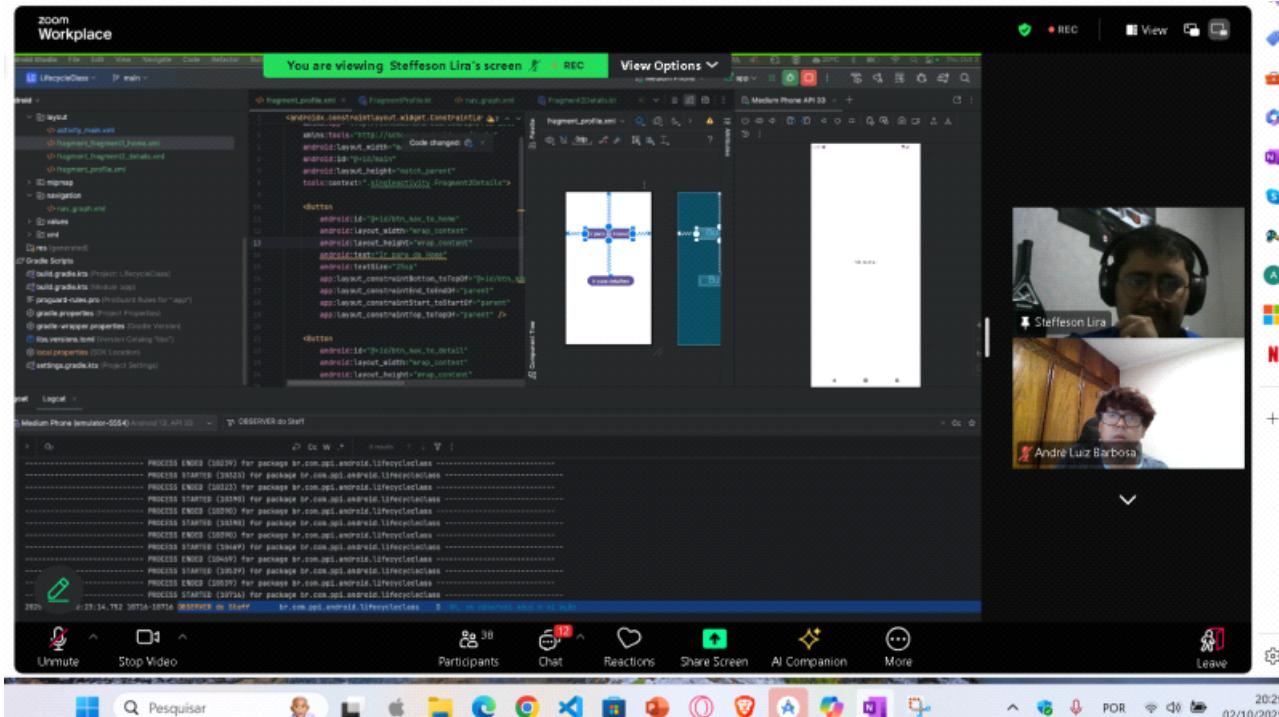
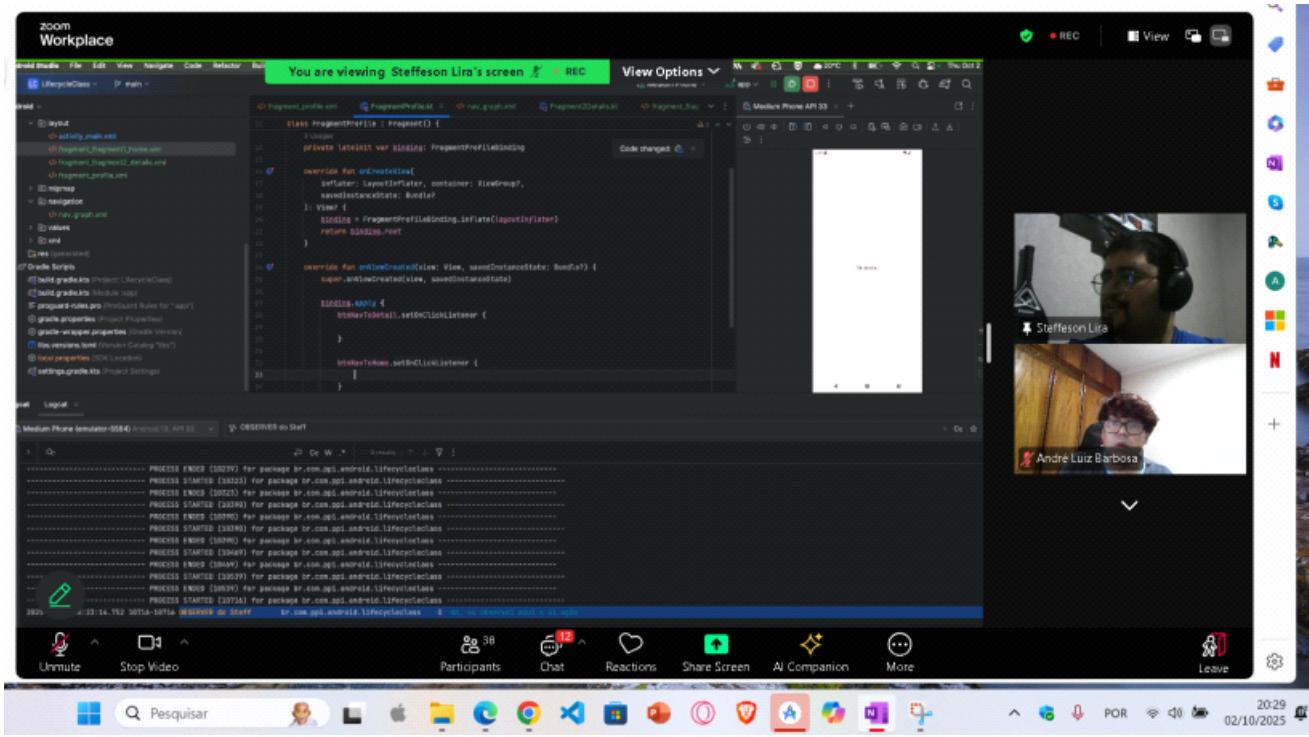
Vai trocar o text view para querer dois ovbtos para gomhome e detail



Bidindo pega do outro exemplo



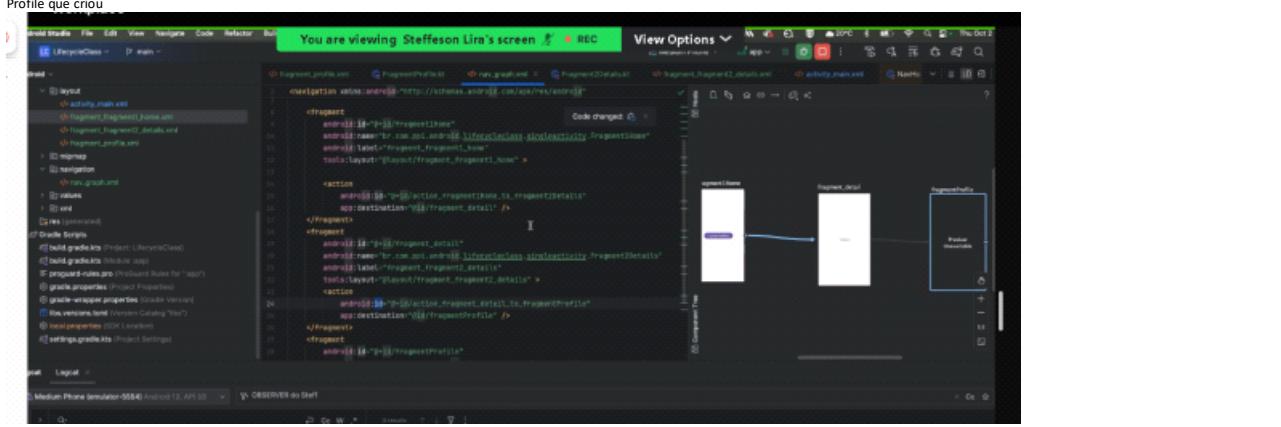
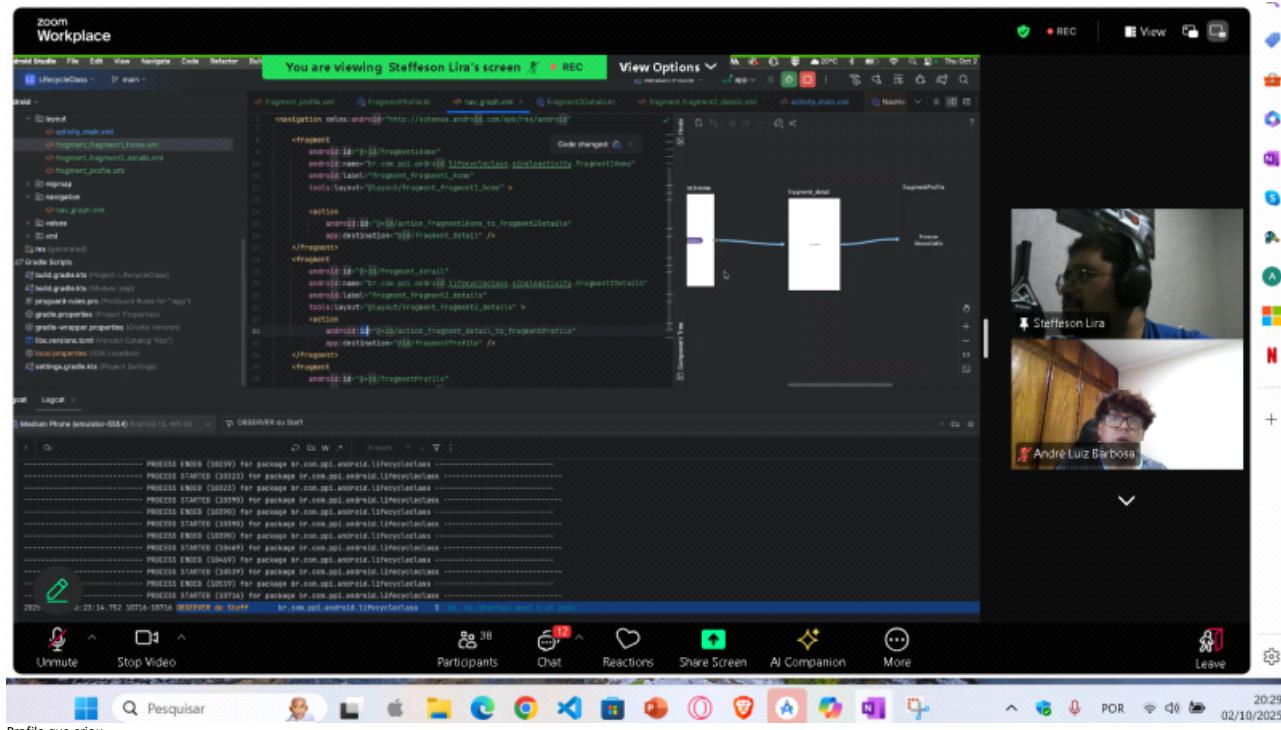
Faz biding ponto aply para criar uma vez só



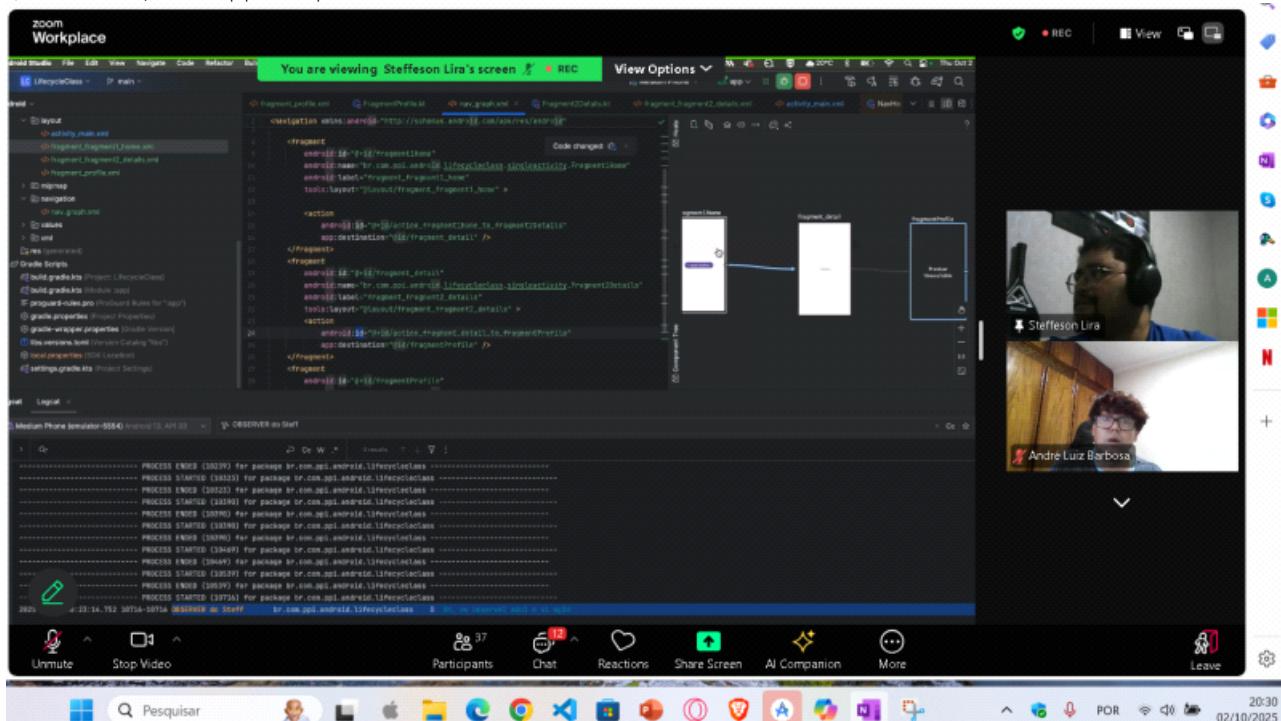
Agora navegar de uma tela para outra, tem que add a tela no profile.

Ligaço direta para ele.

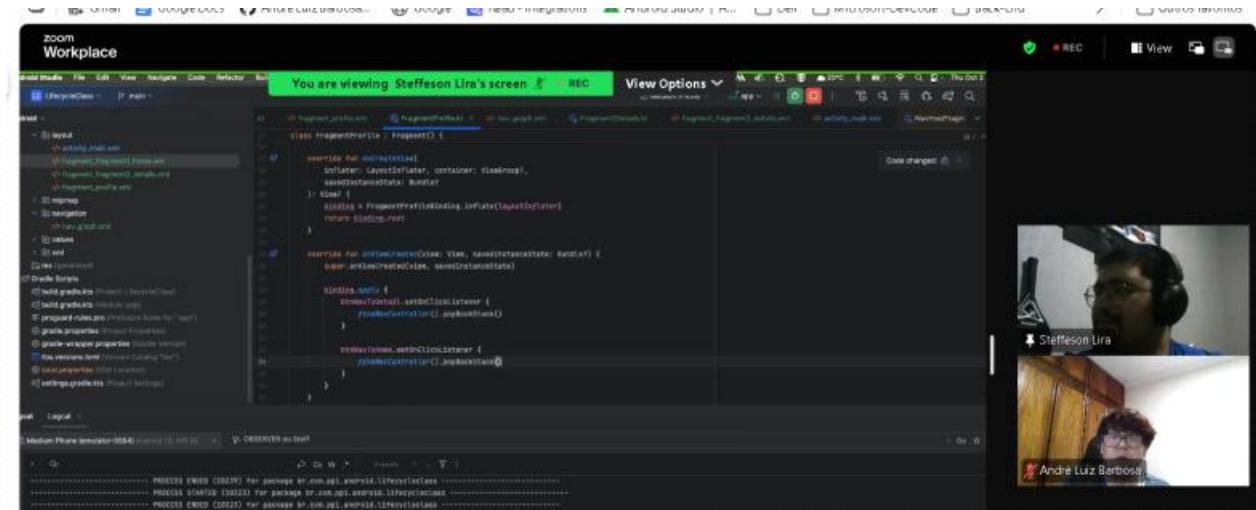
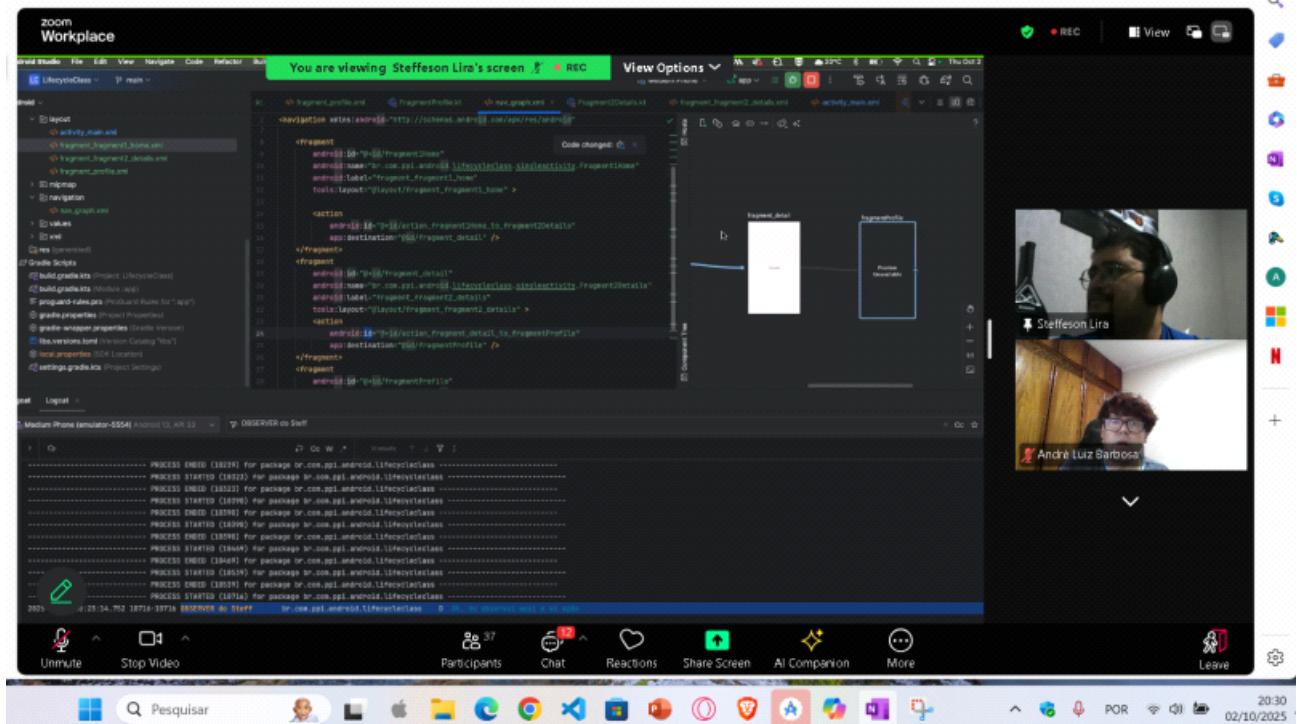
Criou outro action



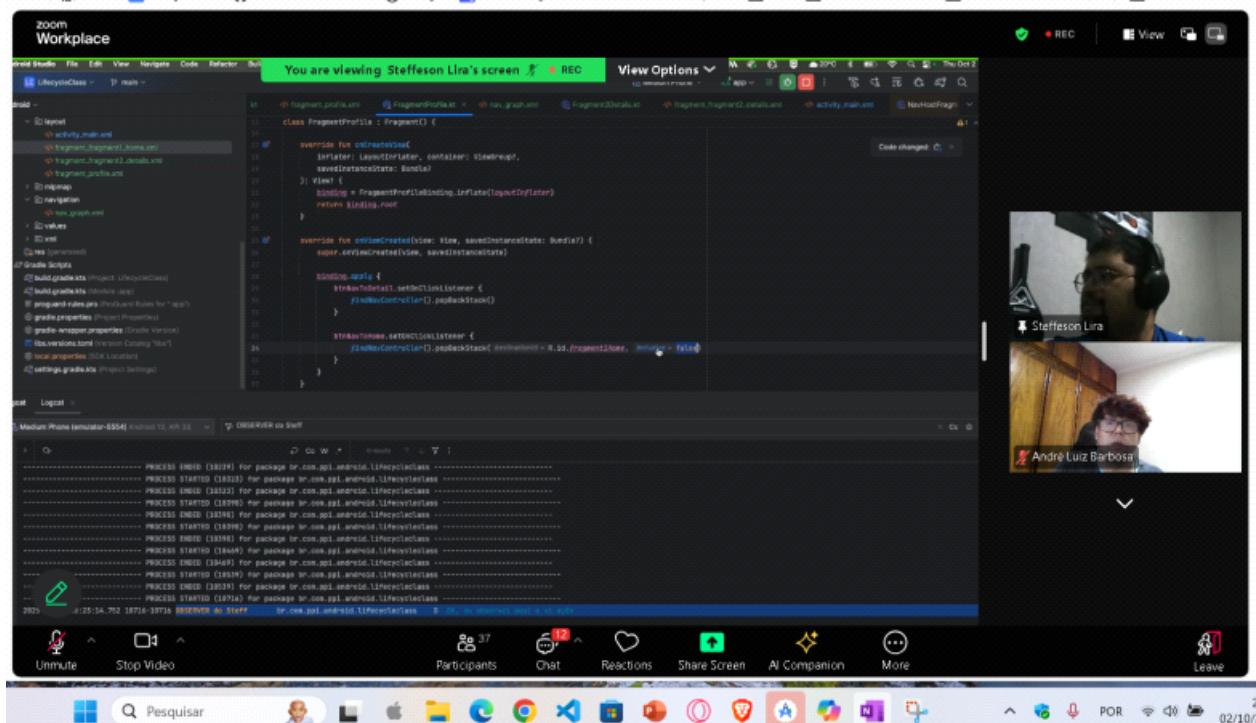
Quando voltar na detail, nav.controle.popBackStack para voitar



Tela 1 home, 2 detail, 3 profile



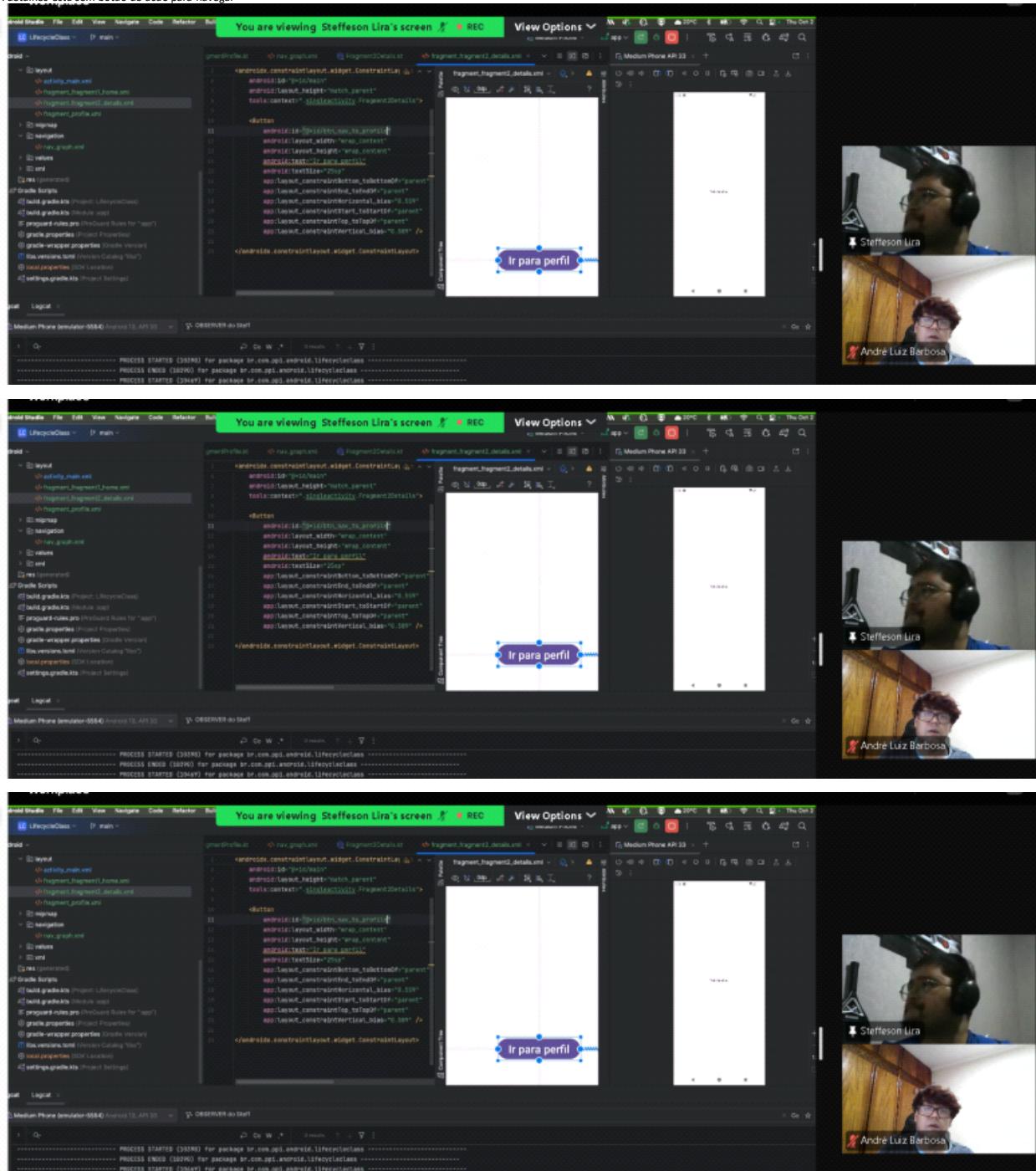
Vai colocar identificador para onde vai navegar na pilha  
Remove tudo que ta na pilha e volta para a tela no caso home que é a que se quer a fragment 1

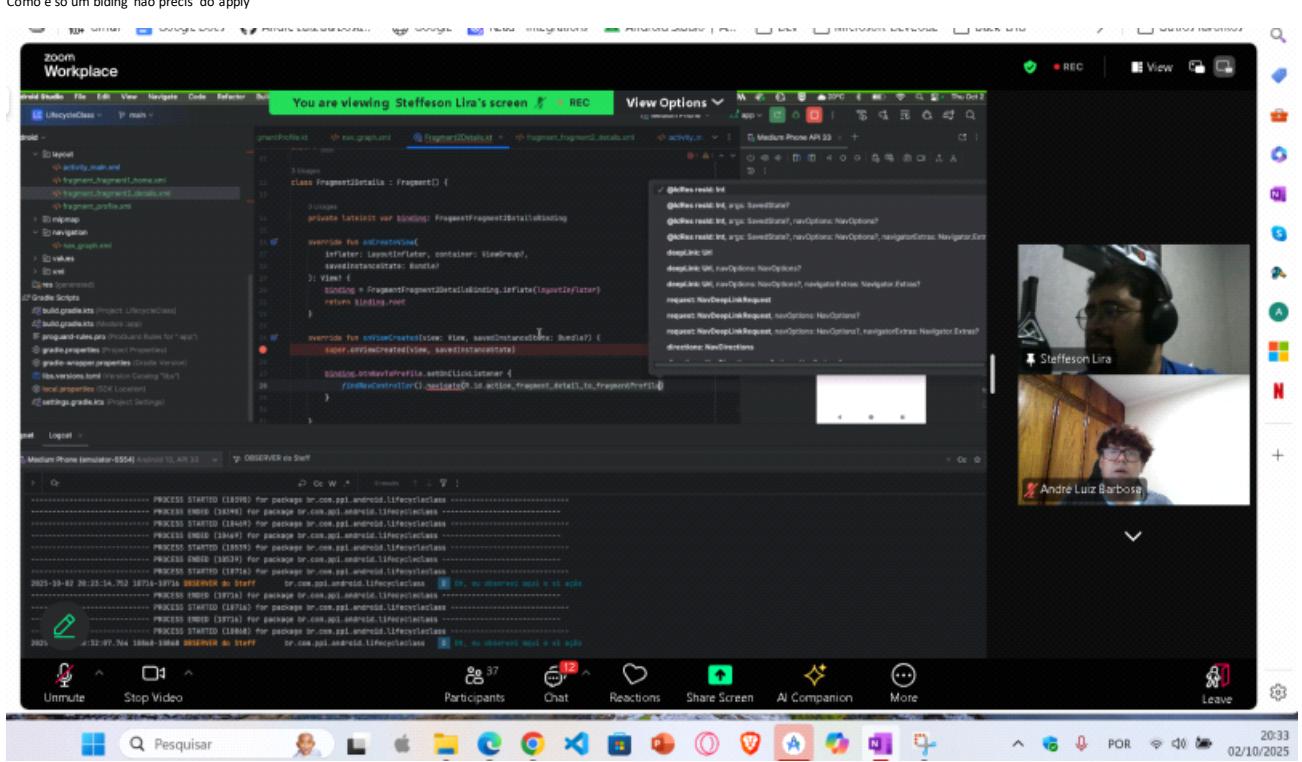
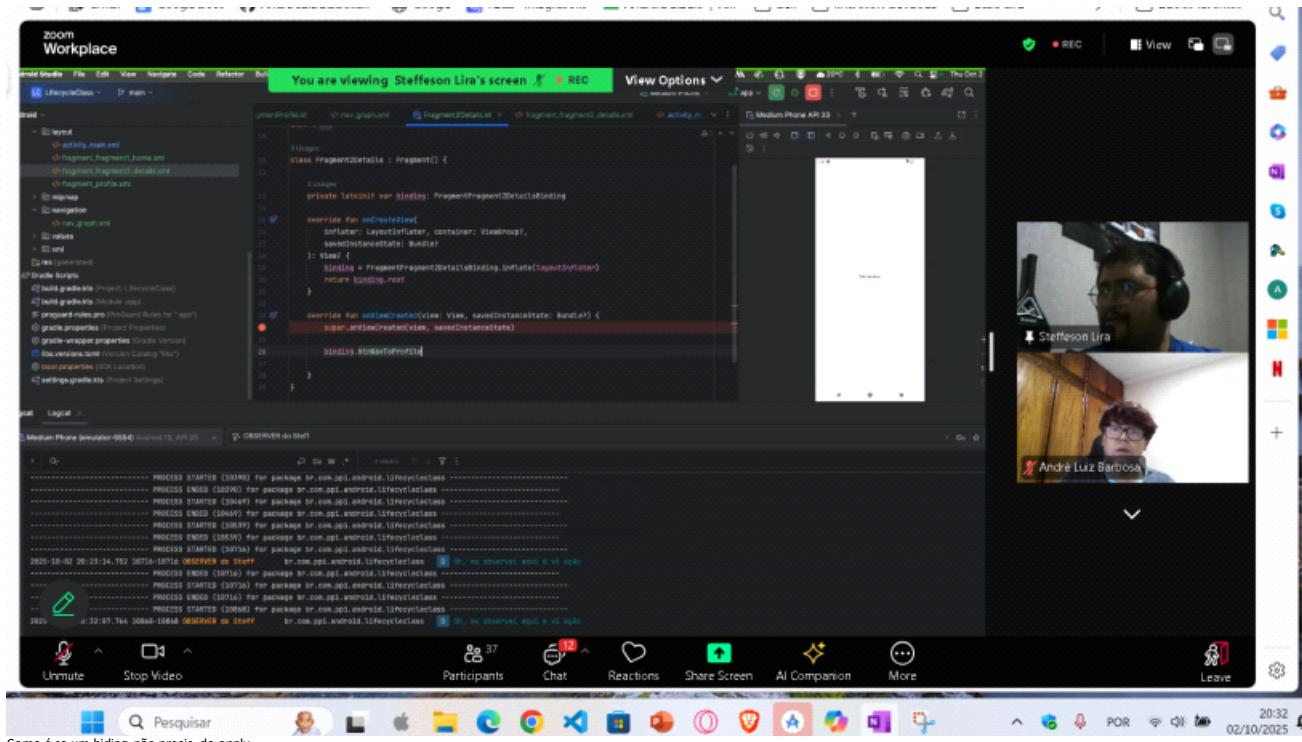


Para isto usou o include

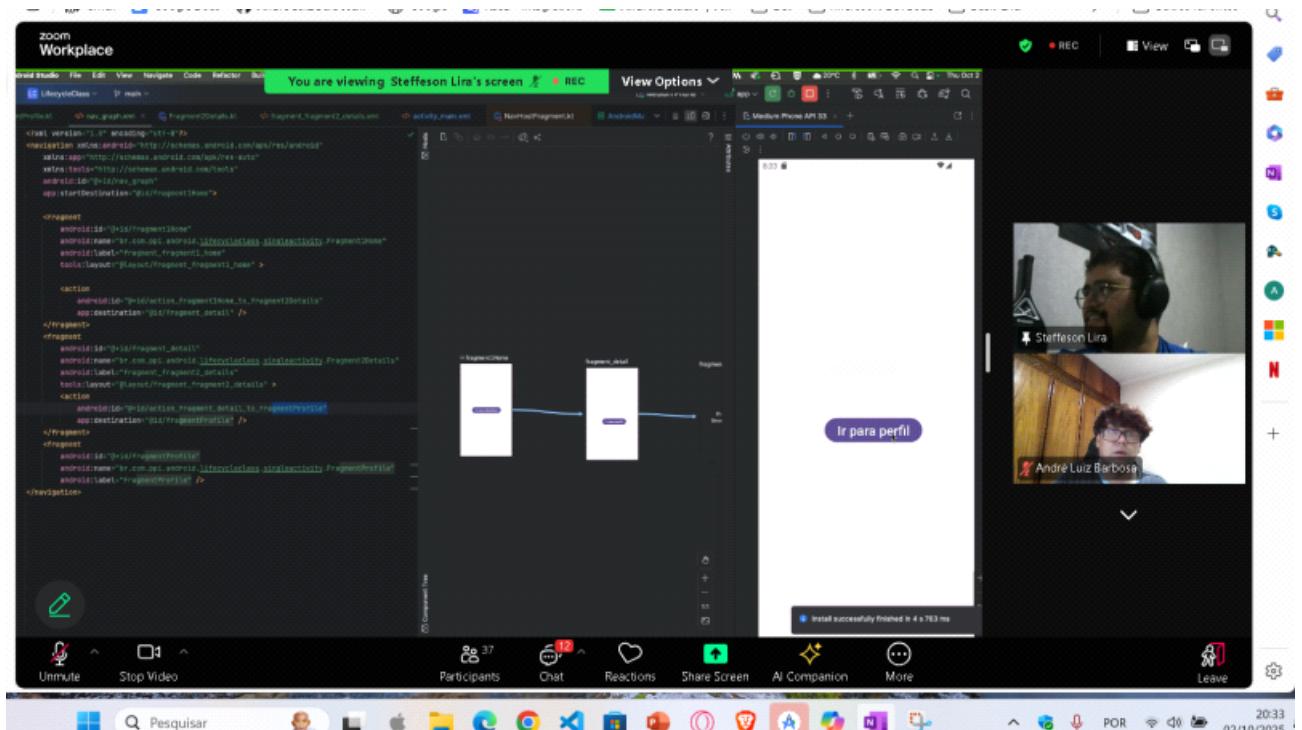
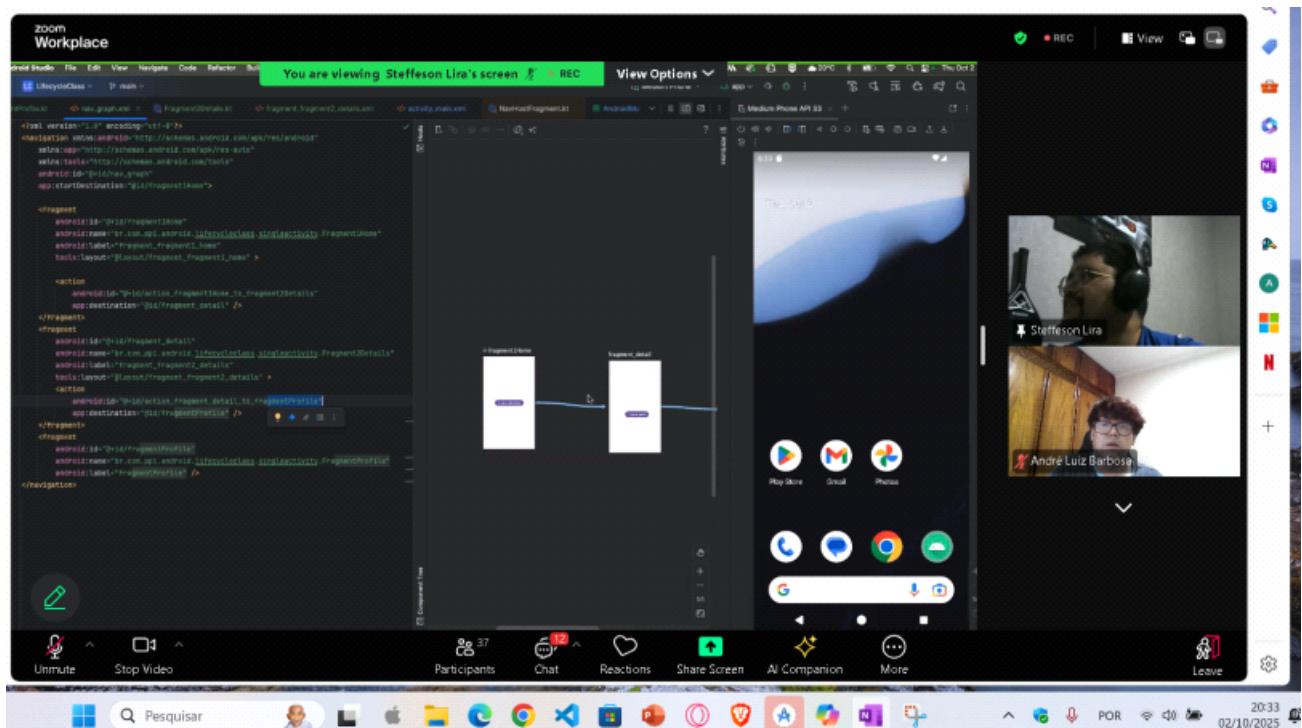
Tem outras maneiras legais para fazer.

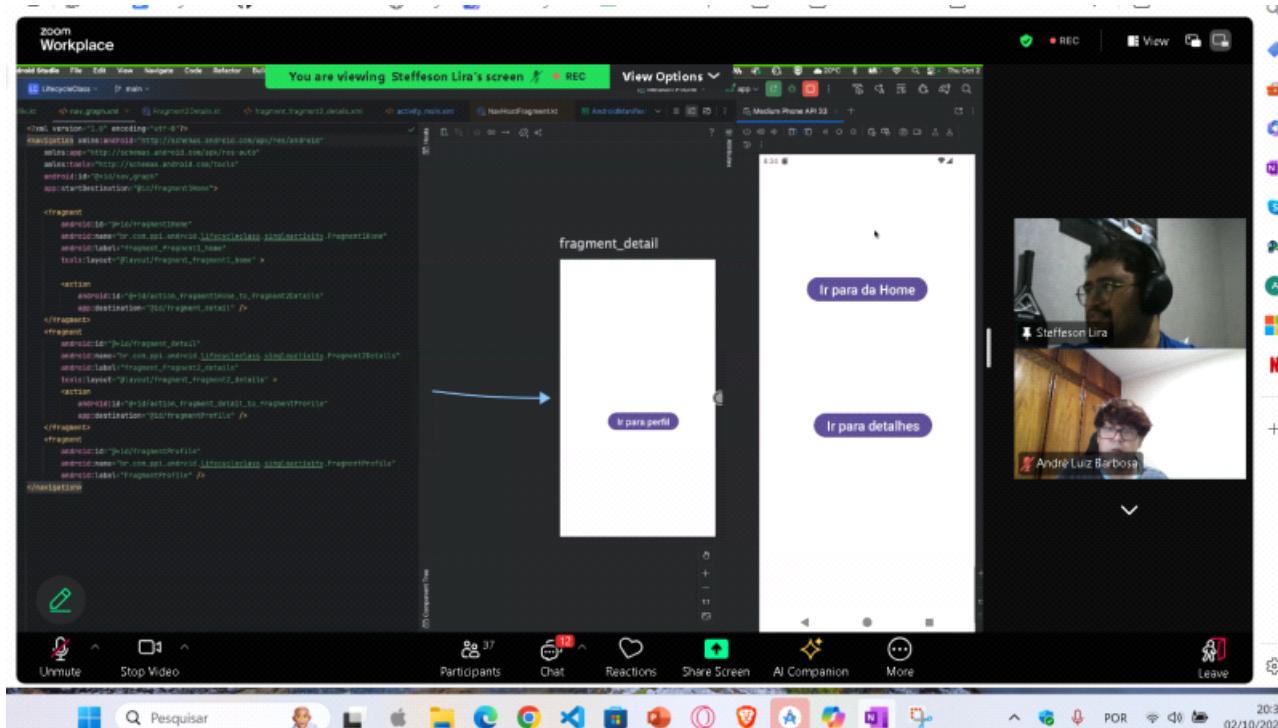
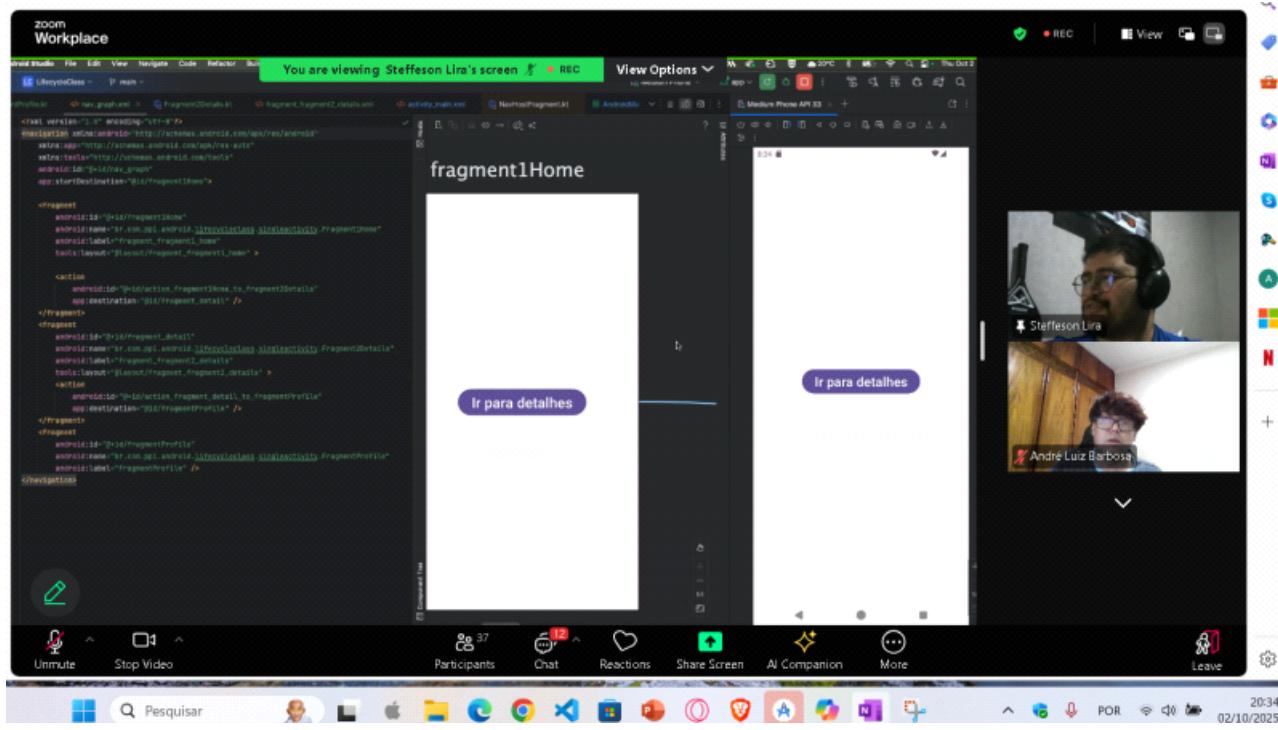
A detalhes está sem botão de ação para navegar

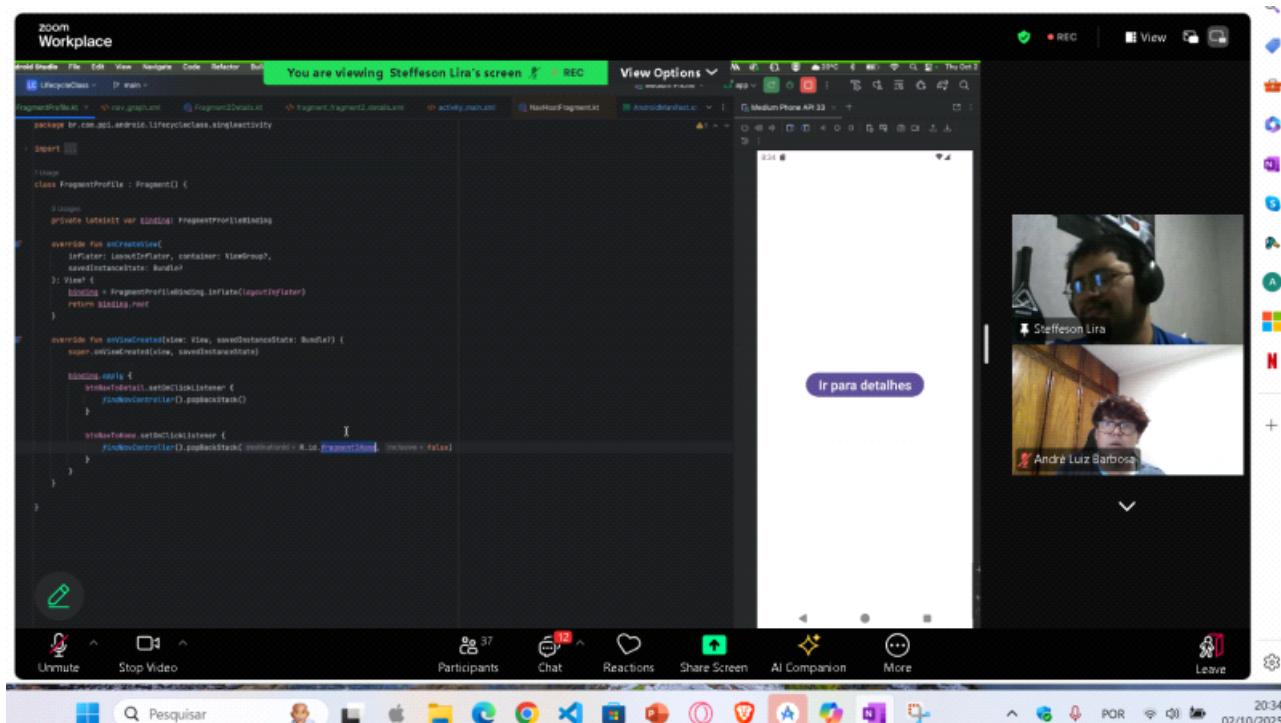
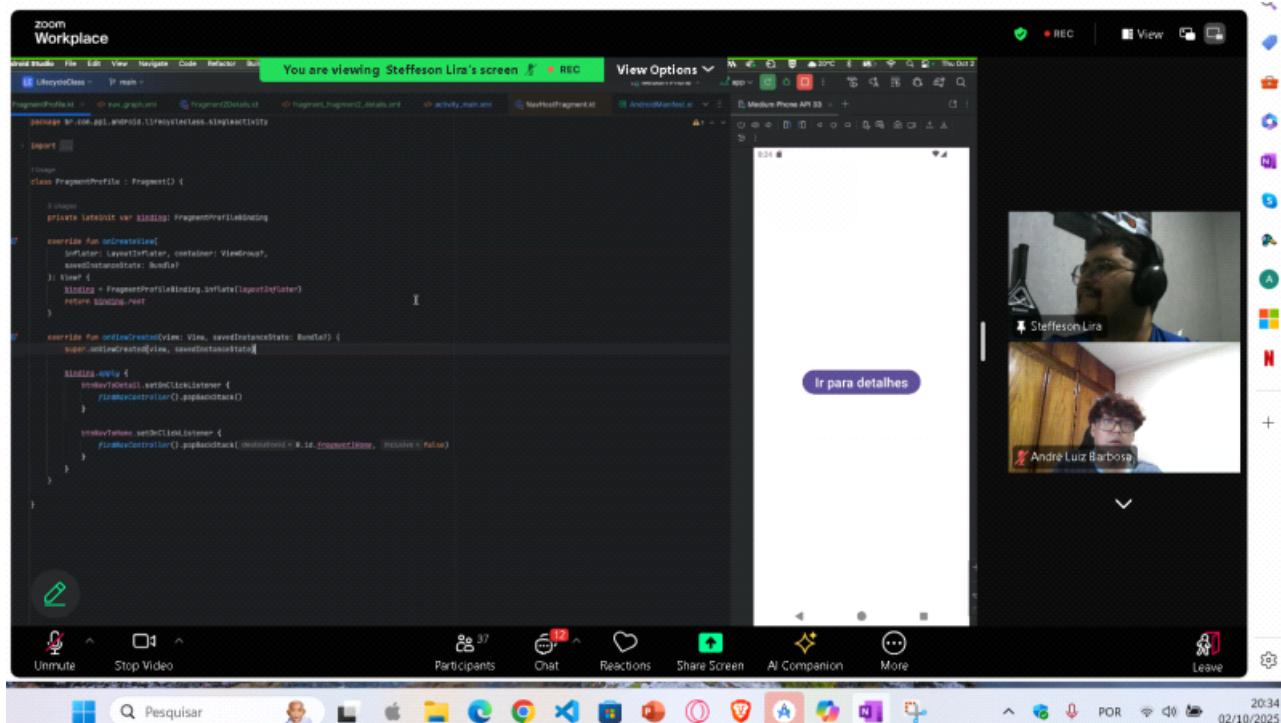




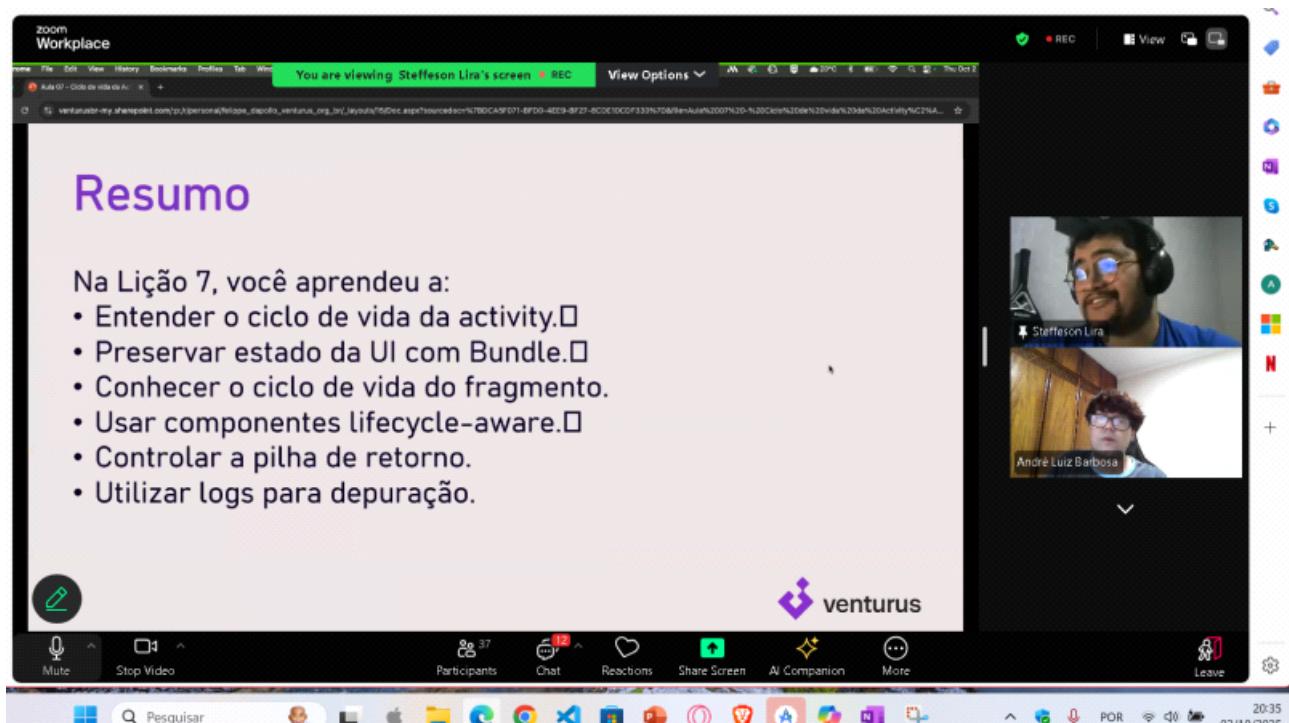
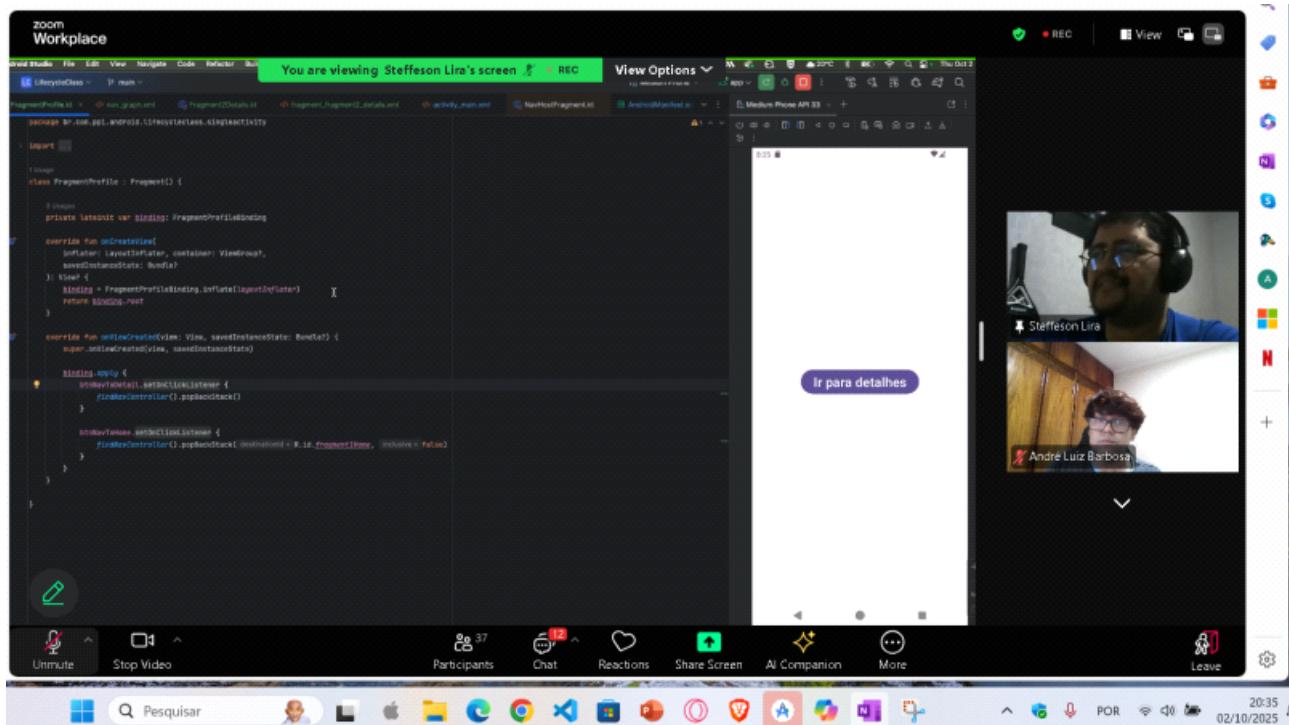
Nome que seja facil de entender de u=ir para uma dpara outra tela

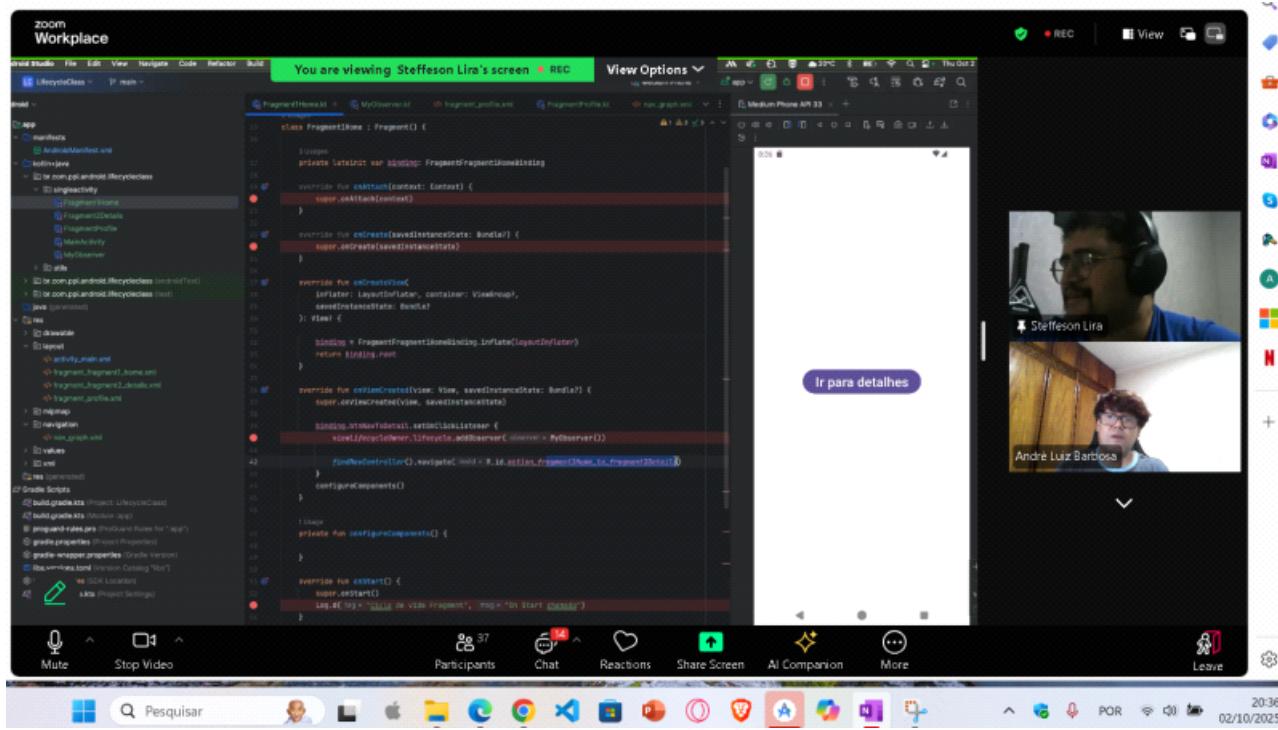




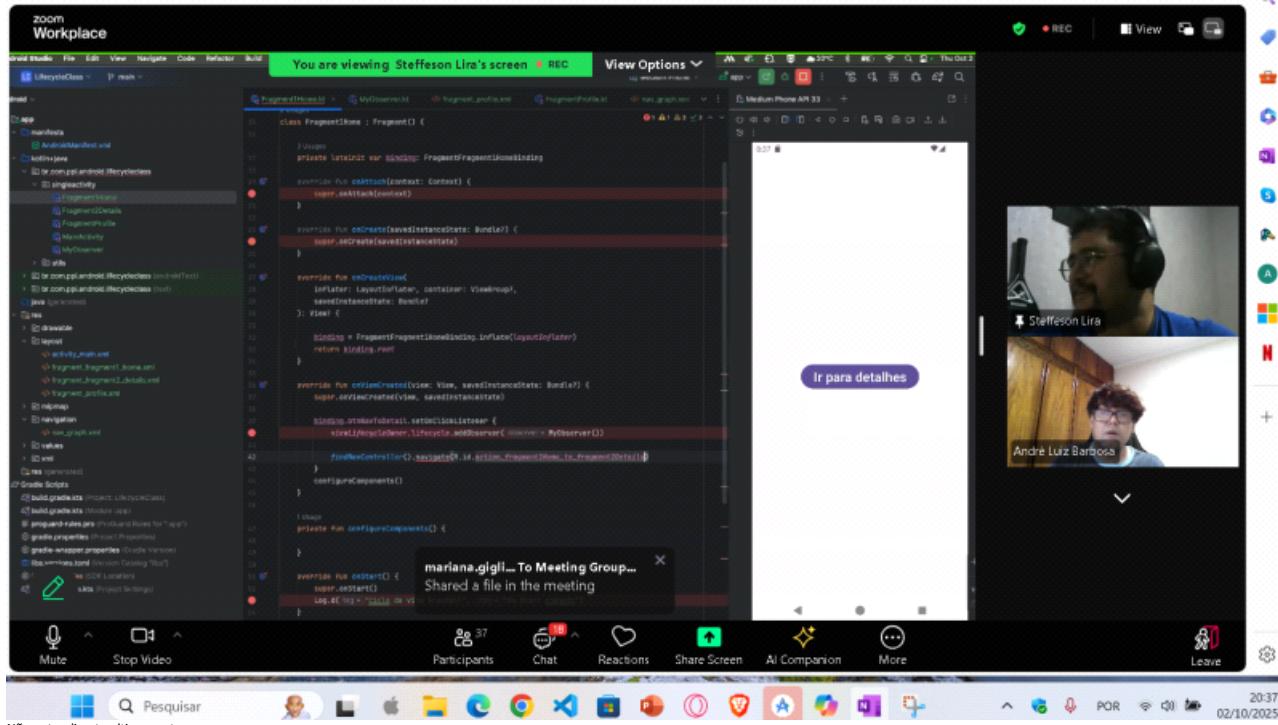


Popbackstack passando por o id??





O argumento que colocou pode passar um string algo assim  
Quando for para outra tela sera assim



Não entendi esta ultima parte

**Resumo**

Na Lição 7, você aprendeu a:

- Entender o ciclo de vida da activity.□
- Preservar estado da UI com Bundle.□
- Conhecer o ciclo de vida do fragmento.
- Usar componentes lifecycle-aware.□
- Controlar a pilha de retorno.
- Utilizar logs para depuração.

**Veja mais**

- Understand the Activity Lifecycle
- Activity class
- Fragment class
- Fragment lifecycle
- Activity lifecycle

The screenshot shows a Zoom meeting interface with a grid of participant thumbnails and their names. The participants are arranged in three rows:

- Row 1:** Steffeson Lira, André Luiz Barbosa, anderson.fernandes, RH Zoom, Eraldo Cunha, Sivonaldo Diogo.
- Row 2:** Iago Ramos Mirandola de Lima, Felipe Dapollo, Reinaldo Cesar Santos, mariana.gigliotti, RENATO, gerdivaldo.santos.
- Row 3:** Renata Rodrigues, Juan Santos de Oliveira, Danilo Gomes, Marcelo Almeida, Daniele Greice, Eliana Ribeiro da Silva.

Below the grid, there are navigation arrows for the previous and next pages (1/2 and 2/2). At the bottom, there are various control buttons: Mute, Stop Video, Participants (32), Chat, Reactions, Share Screen, AI Companion, More, and Leave.

A small pop-up window titled "Rafael Femina To Meeting Group..." with the message "Boa noite pessoal" is visible in the center of the screen.