

# Universidade da Beira Interior

## Departamento de Informática

[width=191pt]brasaoubi.JPG

**Grupo Nº 4 - 2024: *[Navegação e ciclo de Vida- Protocolo]***

Elaborado por:

**[Clausemen Nanro, Ivandra Bernardo, João Maravilhoso]**

Orientador:

**Professor**

October 25, 2025

# Protocolo Experimental: Ciclo de Vida e Intents no Android

## 1 Objetivo

Analisar o ciclo de vida de *Activities* Android e o uso de *Intents* explícitos e implícitos em cenários como mudanças de configuração, interrupções e transições entre *Activities*, avaliando a preservação do estado e o gerenciamento de recursos.

## 2 Materiais

- Dispositivo Android (API 21 ou superior).
- Android Studio (versão mais recente).
- Aplicativo com `MainActivity` (botões para *Intents*) e `SecondActivity`, implementando todos os callbacks do ciclo de vida.
- Logcat para monitoramento de logs.

## 3 Metodologia

1. **Configuração:** Configurar um projeto no Android Studio com `MainActivity` exibindo o estado do ciclo de vida em uma `TextView` e botões para *Intents* explícito e implícito. Criar `SecondActivity` para receber dados via *Intent* explícito. Adicionar logs em todos os callbacks.
2. **Teste de Ciclo de Vida:** Executar o aplicativo e verificar a sequência de `onCreate()`, `onStart()`, `onResume()`. Rotacionar o dispositivo para testar `onSaveInstanceState()` e `onRestoreInstanceState()`. Simular uma interrupção (e.g., chamada) para observar `onPause()` e `onStop()`.
3. **Teste de Intent Explícito:** Clicar no botão para iniciar `SecondActivity` com *Intent* explícito, passando uma mensagem. Verificar se `onPause()` é chamado em `MainActivity` e se `SecondActivity` exibe a mensagem.
4. **Teste de Intent Implícito:** Clicar no botão para iniciar *Intent* com `ACTION_DIAL.Confirmar que`
4. **Teste de Retorno:** Retornar de `SecondActivity` ou do discador, verificando se `onResume()` restaura o estado.

5. Teste de Recurso: Implementar um recurso simulado (e.g., variável de conexão) inicializado em `onResume()` e liberado em `onPause()`. Testar durante transições.

## 4 Resultados Esperados

- Callbacks do ciclo de vida são chamados na ordem correta.
- O estado da `TextView` é preservado após mudanças de configuração.
- *Intent* explícito inicia `SecondActivity` com a mensagem correta.
- *Intent* implícito abre o discador adequadamente.
- Recursos são gerenciados sem vazamentos.

## 5 Análise

Analisar logs do Logcat para confirmar a sequência de callbacks e o comportamento dos *Intents*. Avaliar a preservação do estado e o gerenciamento de recursos. Verificar a configuração de *intent-filters* no `AndroidManifest.xml`.

## 6 Conclusão

Este protocolo avalia a integração do ciclo de vida com *Intents*, garantindo robustez, preservação de estado e eficiência no gerenciamento de recursos.