Departamento de Engenharia Informática
TeSP em Programação de Sistemas de Informação

Acesso Móvel a Sistemas de Informação
2016/2017

Docentes Sérgio Lopes, sergio.lopes@ipleiria.pt
David Safadinho, david.safadinho@ipleiria.pt

Ficha de Exercícios N/01

Introdução ao Android Studio como ferramenta de desenvolvimento

Geral

Os temas para projeto devem ser submetidos pelos alunos, respeitando as regras definidas no presente documento, em conjunto com as regras que sejam indicadas nas restantes unidades curriculares nas quais se integra a aplicação móvel a desenvolver em AMSI. Todos os temas estão sujeitos a aprovação prévia.

Os projetos devem ser criados com as seguintes características:

- Nome do projeto: CalculadoraRPNActivity
- SDK: API 19
- Modelo: Empty Ativity
- Nome da atividade: CalculadorRPN
- Nome do layout: calculadora

Ao longo dos exercícios o desenho e implementação inicial irá sofrer alterações, assim, se pretender manter o código que é desenvolvido em cada exercício deverá fechar o *Android Studio* e copiar a pasta do projeto à medida que os exercícios forem sendo concluídos.

Calculadora RPN (Reverse Polish Notation, ou POSTFIX, https://en.wikipedia.org/wiki/Reverse_Polish_notation) pretende implementar uma calculadora simples, para operações sobre inteiros, usando a notação RPN. Nesta notação não existem parêntesis para definir ordem das operações, sendo que os operadores (+, -, *. /) aparecem depois dos operandos (números). Como exemplo, se pretendermos somar os dois números 23 e 98:

- Notação habitual: 23+98
- Notação RPN 23 98 +

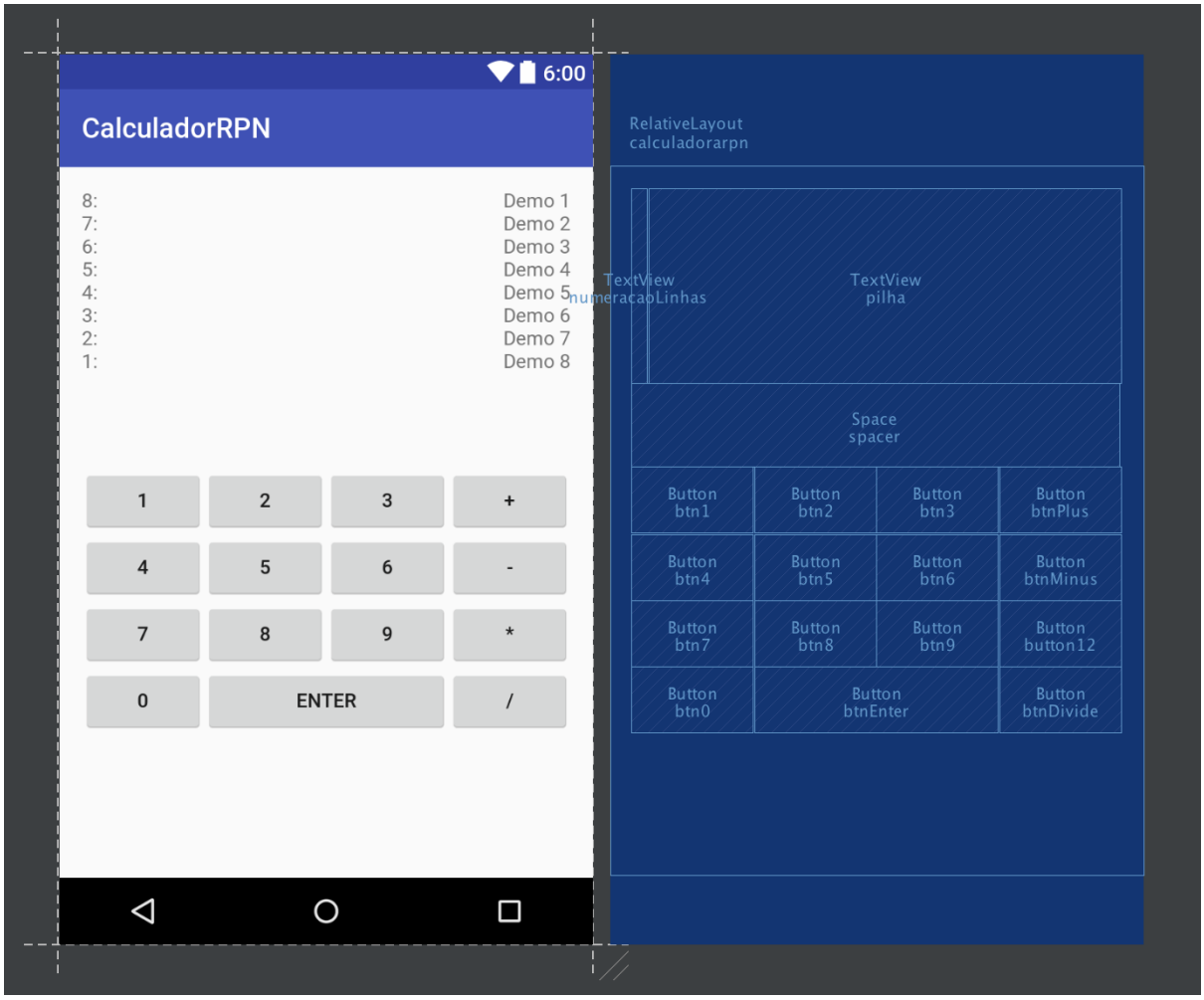
Para o exercício, considera-se que a calculadora opera sempre sobre os dois últimos números disponíveis e assim, se o utilizador clicar num dos operadores serão sempre usados os dois números presentes no visor. O resultado da operação é colocado no visor, logo na primeira posição. Se apenas existir um número no visor não deverá ser possível executar qualquer operação, sendo que nada acontece se o utilizador pressionar os botões de +, -, / ou *.

O visor deve apresentar 8 linhas, onde cada linha corresponde a um valor guardado pela calculadora. Podem ser guardados mais que 8 valores, mas só são apresentados os primeiros 8.

Exercício 1 – Desenho da interface gráfica

Depois de criar o projeto, comece por desenhar a interface gráfica seguinte o modelo da imagem abaixo.

Figura 1 - Sugestão de interface da calculadora.



Como estamos a usar um layout relativo (Relative Layout), a posição dos componentes relativa a outros componentes presentes no layout, e o Android Studio tenta definir as relações automaticamente com base na ordem de adição dos componentes e com base no local onde o componente foi introduzido (atenção às setas horizontais/verticais de posicionamento). Por esse motivo, recomenda-se que sejam adicionados os componentes começando de cima para baixo, da esquerda para a direita.

Se o layout não estiver correto, será necessário editar o XML e definir, manualmente, todas as relações. Neste caso devem ser usadas as propriedades:

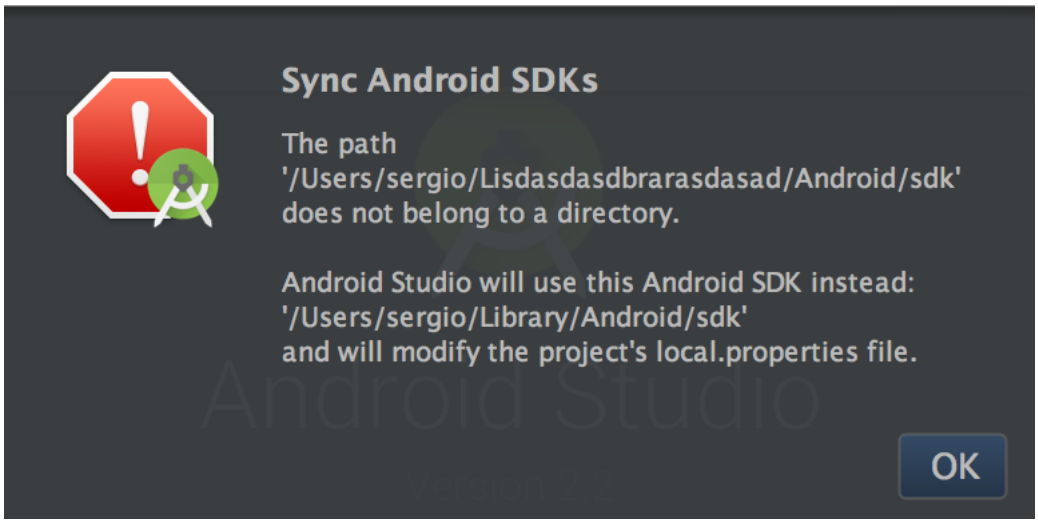
- android:layout_below – Posiciona este componente debaixo de outro componente.
- android:layout_toEndOf – Alinha o início do componente pelo fim de outro componente (prende à esquerda).
- android:layout_alignEnd – Alinha o fim do componente pelo fim de outro componente (prende à direita).

Podem ser necessárias outras propriedades, pelo que devem consultar a tabela de propriedades disponíveis em https://developer.android.com/reference/android/widget/RelativeLayout.LayoutParams.html

QUESTÃO: Como resolver o desafio de alinhar o botão “ENTER”, de modo a ocupar o espaço de dois botões?

NOTA: Ao partilharmos os projetos, muitas vezes, não temos atenção às configurações locais (diferentes para cada máquina) que o mesmo contém. Se apenas copiarmos a pasta do projeto com o código e todos os recursos, ao abrirmos o projeto noutra máquina é possível se que seja apresentada uma mensagem de erro como a seguinte:

Figura 2 - Erro apresentado se o conteúdo do ficheiro local.properties estiver errado.



Este erro é inofensivo e o IDE corrige-o automaticamente. Para o evitar podemos apagar o ficheiro ***local.properties*** presente na raiz do projeto. Este ficheiro não deve ser colocado no sistema de controlo de versões, deve ser adicionado à lista de exclusões.

Exercício 2 – Implementação simplificada

A primeira implementação irá recorrer a um vetor do tipo *Integer* com limite para 12 posições. Esta será a nossa pilha onde se colocarão os valores, sendo o seu "topo" a posição zero. Recorde que numa pilha, os valores são inseridos e removidos apenas no "topo" e que quando um elemento é inserido os restantes avançam uma posição, quando um elemento é retirado, os restantes recuam uma posição. Neste caso consideramos a posição zero como sendo o topo da pilha, mas poderíamos também considerar a posição 11 (fim do vetor).

Adicione a cada um dos botões o(s) evento(s) necessário(s) para implementar as funcionalidades base da calculadora:

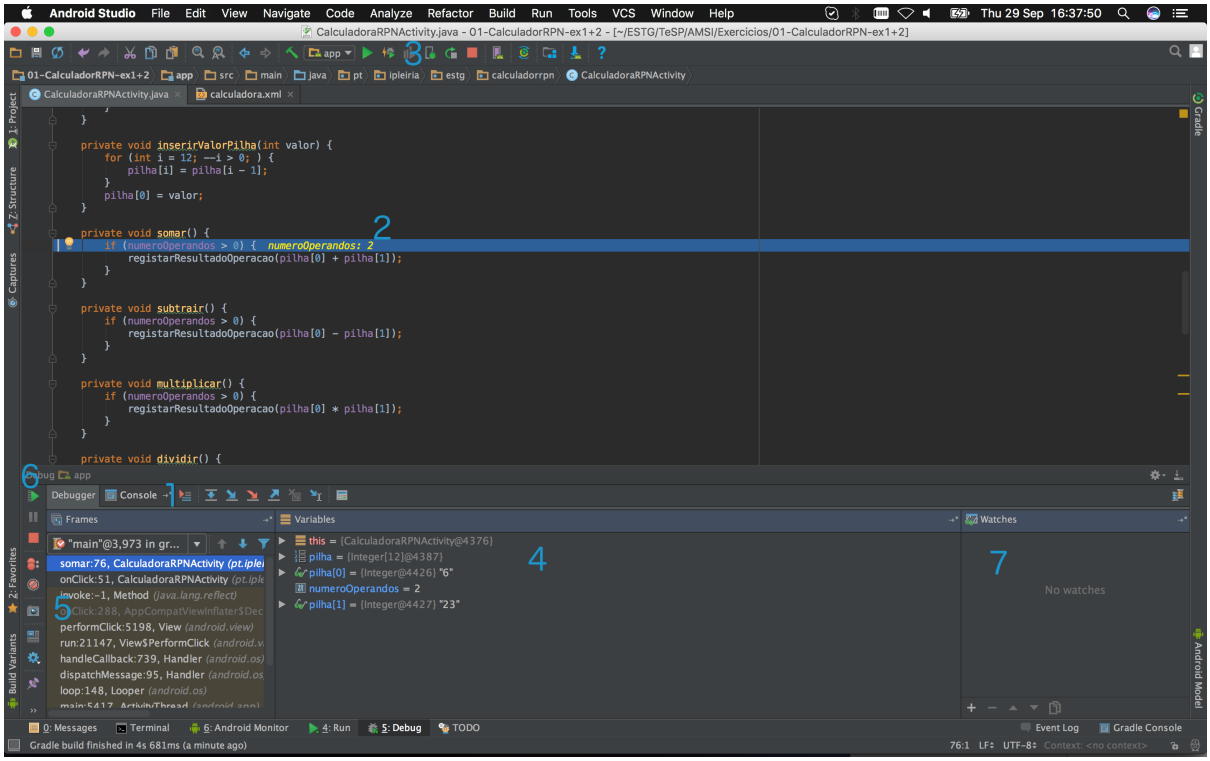
- Tecla "Enter" regista o valor na pilha e atualiza o visor da calculadora
- Teclas de operações executam a operação indicada e atualizam o visor:
    - Somar dois números
    - Subtrair dois números
    - Dividir dois números
    - Multiplicar dois números

DICA: Adicione um método com a assinatura public void onClick(View v) na classe da atividade e faça com que esta implemente a interface View.OnClickListener.

Exercício 3 – Fluxo de execução da aplicação e ferramentas de depuração (debug)

A implementação fornecida contém alguns erros. Usando as ferramentas de depuração.

Figura 3 - Elementos de depuração de aplicações.



1. Botões de controlo da sessão de depuração, permitem saltar linhas, entrar em métodos, etc.
2. Informação de contexto fornecida pelo IDE para a linha em que a execução está parada.
3. Botão para iniciar a sessão de depuração.
4. Apresenta a lista de variáveis, e valores, conhecidos para o contexto atual (classe, método).
5. Lista de métodos invocados até ao momento, permite visualizar os métodos e a ordem pela qual foram chamados.
6. Coluna de botões com opções sobre a sessão de depuração.
7. Área onde são colocadas variáveis de interesse que pretendemos observar. À medida que os valores das variáveis, ou expressões, vão sofrendo alterações, a informação aparece nesta janela.

Adicione os métodos de ciclo de vida da aplicação (mencionados na aula teórica), inclua neles instruções para imprimir textos para a consola/log e execute a aplicação verificando a ordem das chamadas. Use o *debugger* para poder parar em cada método e ver as informações de contexto sobre o estado da aplicação.

Exercício 4 – Melhorar implementação

Usar um vetor de tamanho fixo, limitado a 12 elementos, reduz a utilidade da nossa calculadora. Assim, para tornar a implementação mais útil altere o código para fazer uso da classe *Stack* fornecida pelo SDK e que nos permite implementar uma pilha genérica com tamanho variável (https://developer.android.com/reference/java/util/Stack.html).