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

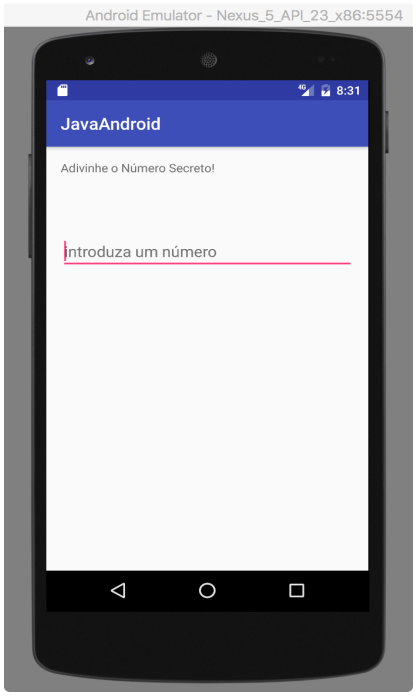
Ficha de exercícios N/00

Introdução à linguagem Java para Android

Geral

O desenvolvimento de aplicações para dispositivos *Android* recorre à linguagem de programação Java, implementada de forma específica para esta plataforma. "Java" na plataforma *Android* tem a mesma sintaxe, fornece implementação das classes base/fundamentais, mas faz uso de uma versão da máquina virtual específica para dispositivos móveis. Assim, nem sempre é possível usar bibliotecas Java, ou código, desenvolvidas para a plataforma Java da Oracle.

Os exercícios da aula serão desenvolvidos em *Android Studio* 2.2, tendo como versão mínima o *Android* 4.4 (*KitKat*).



No fim deste exercício deverá ter uma aplicação *Android* com um campo de texto (*TextView*) e um campo para inserir números (*EditView*, configurada para tipo **number**) com o aspeto da imagem à esquerda.

Este é um pequeno jogo onde o utilizador é convidado a adivinhar um número secreto.

Dependendo do tipo de simulador escolhido, o aspeto poderá ser diferente do apresentado na imagem.

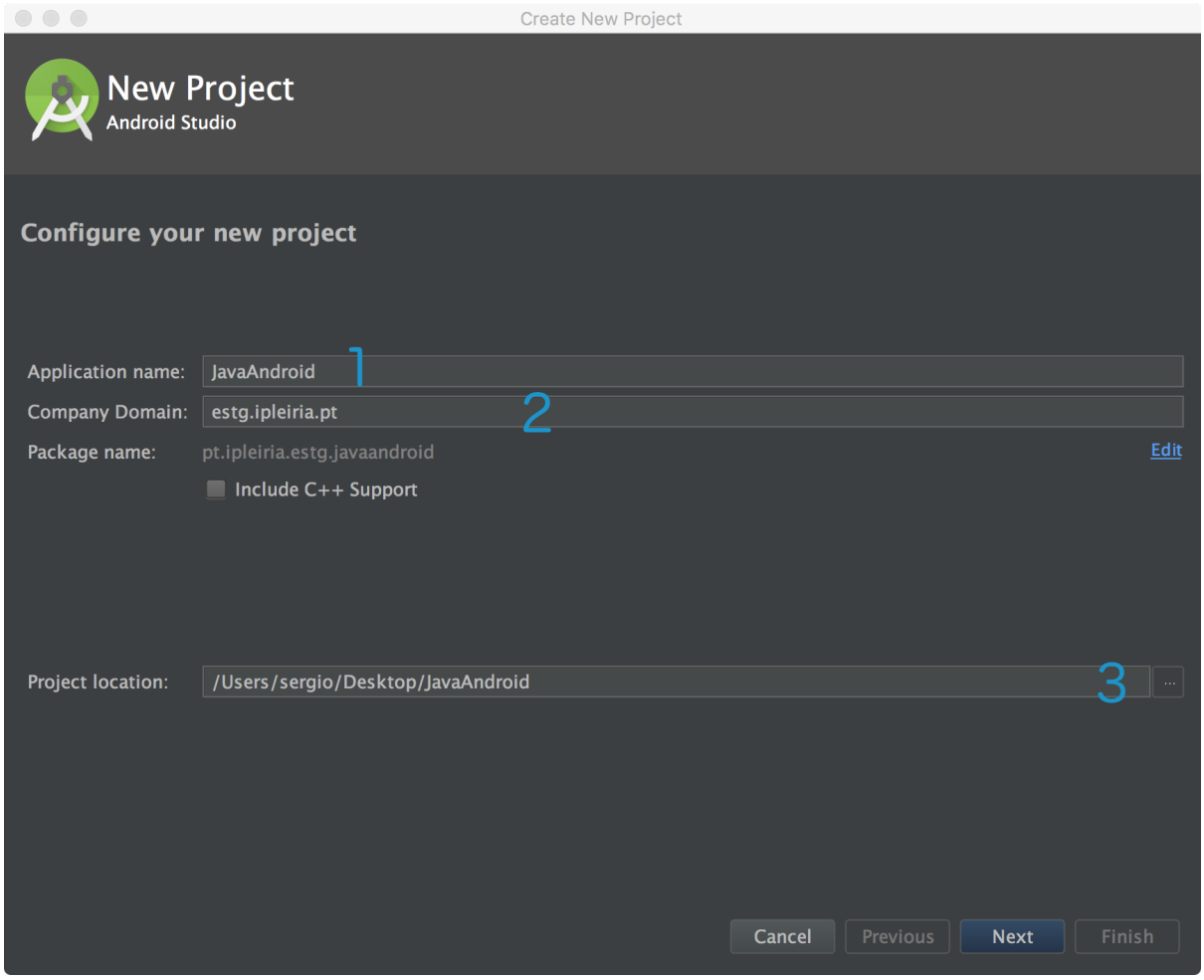
Exercício – Jogo de Adivinha

Para este exercício pretende-se a criação de uma aplicação para *Android* que ofereça ao utilizador um pequeno jogo em que terá a possibilidade de adivinhar um número secreto, determinado pela aplicação. Assim, a aplicação deve, ao iniciar, determinar aleatoriamente, um número inteiro, entre 0 e 10, e pedir ao utilizador que tente adivinhar o número secreto.

A aplicação deve avisar o utilizador se a sua sugestão é inferior, superior ou igual ao número secreto, sendo que o jogo termina quando o utilizador descobrir o número ou tiver tentado 5 vezes.

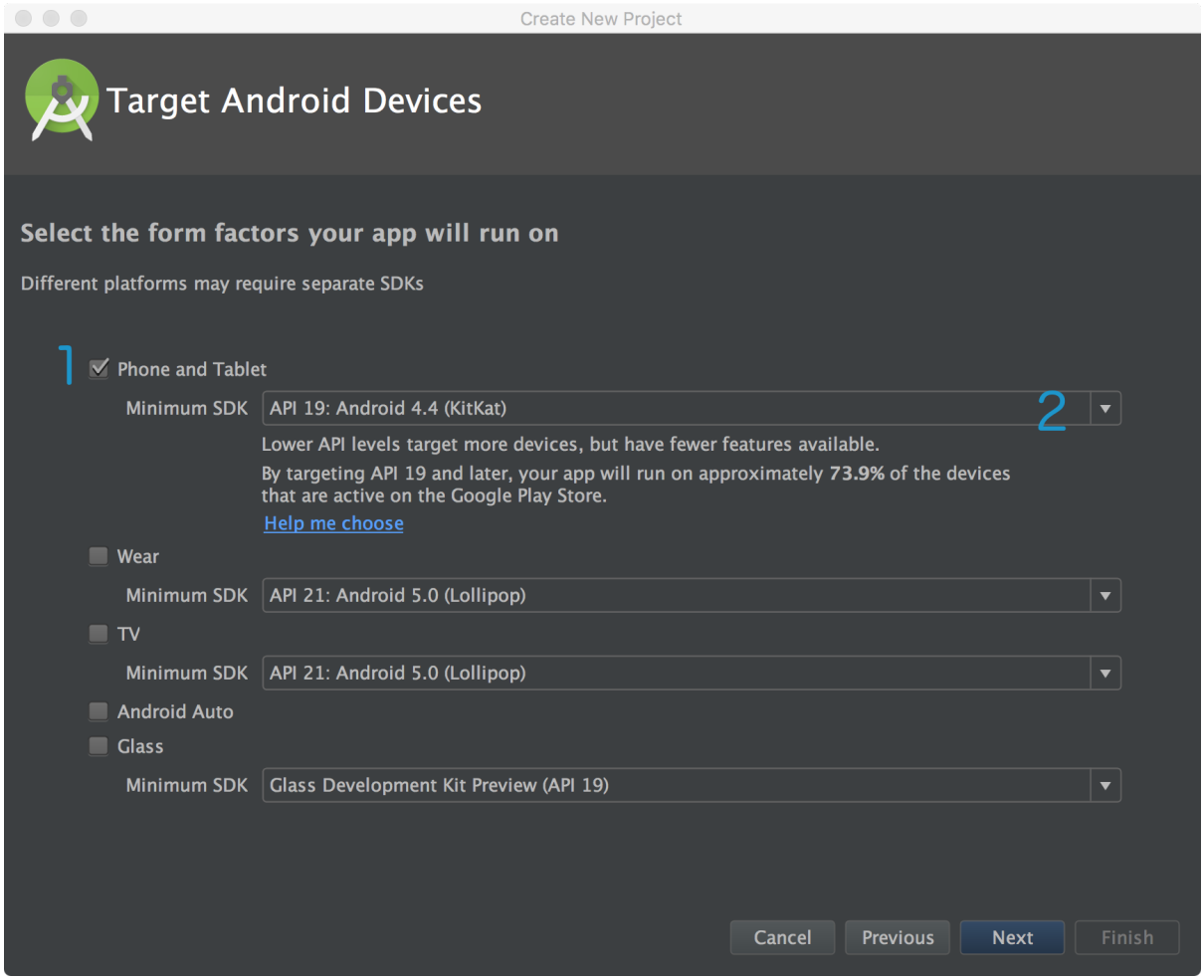
Crie um projeto *Android* novo com o nome **JavaAndroid**. Siga os passos, e configurações, indicados nas imagens das páginas seguintes.

Figura 1 - Configuração do projeto.



1. **Aplication name**: Nome da aplicação, usado como nome do projeto e nome visível da aplicação quando instalada no dispositivo.
2. **Company Domain**: Domínio/endereço WEB para a página da empresa ou da aplicação. É usado para criar o package base da aplicação; semelhante ao *namespace* de C#.
3. **Project location**: Localização, em disco, do projeto. Não deve conter espaços.

Figura 2 - Tipo de dispositivos e funcionalidades disponíveis.



1. Cada *checkbox* permite ativa a compilação para tipos de dispositivos diferentes, com utilização do respetivo SDK.
2. Valor mínimo do SDK usado, limita o tipo de funcionalidades disponíveis durante o desenvolvimento de aplicações. Usando a versão mínima do SDK com valor 19 (*Android KitKat*), a aplicação só poderá ser usada em dispositivos com esta versão, ou superiores, instalada.

Depois de configurados os detalhes base do projeto, aplicação e SDK, o assistente permite criar um esqueleto de código para facilitar o início de aplicações típicas. Neste caso iremos usar a o modelo **Empty Activity** que criar uma atividade vazia, contendo apenas um componente de texto (*TextView*).

Figura 3 - Lista de modelos/esqueletos para criação de aplicações.

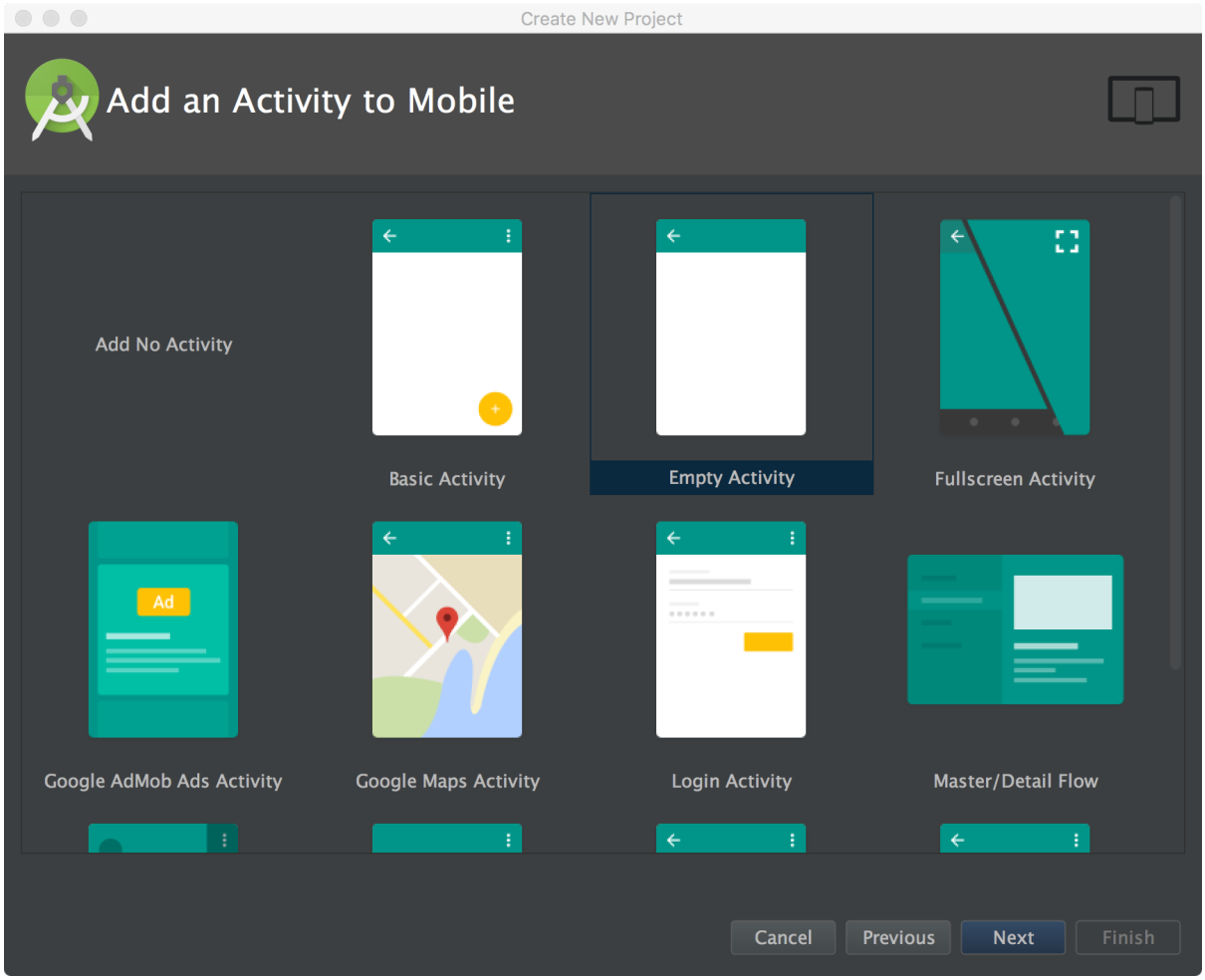
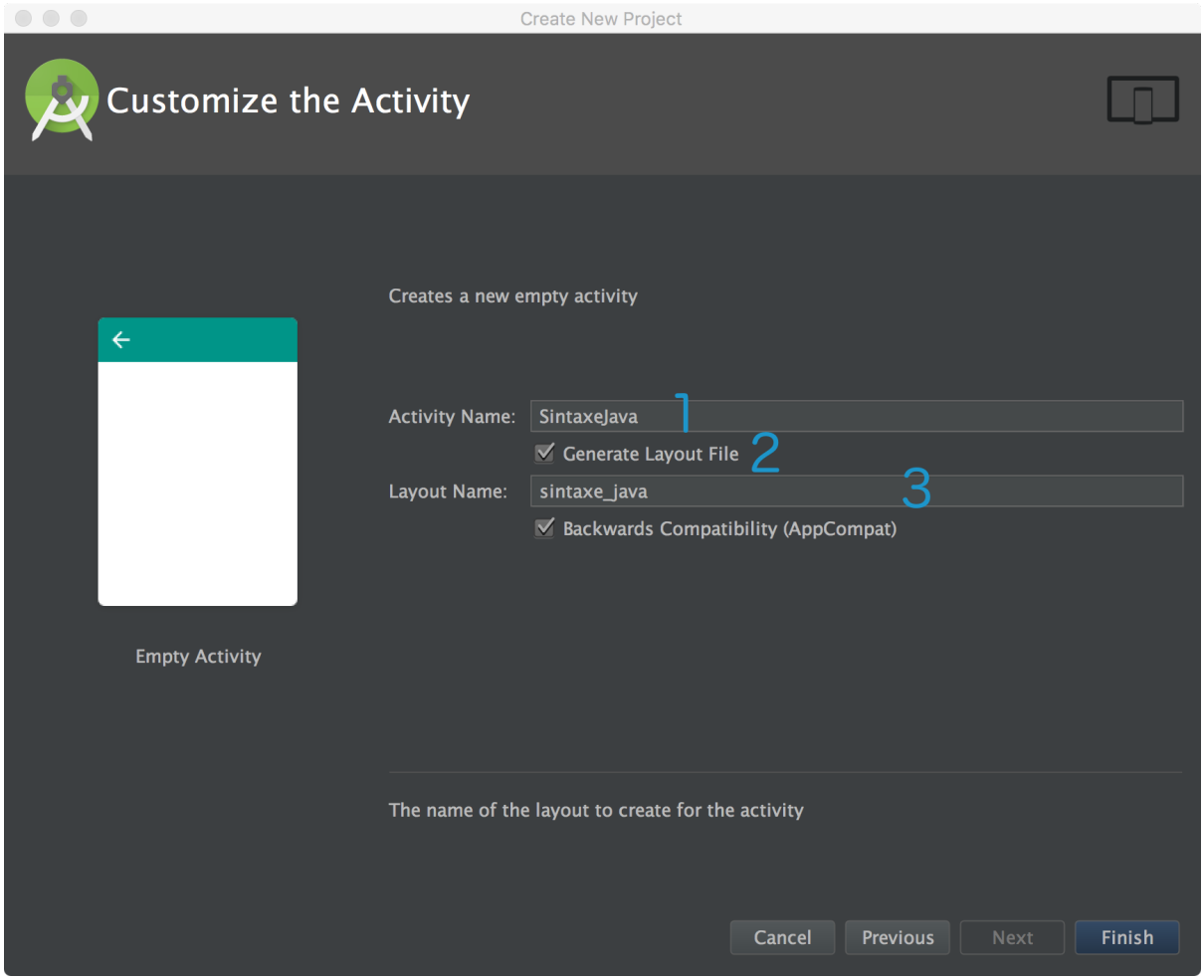


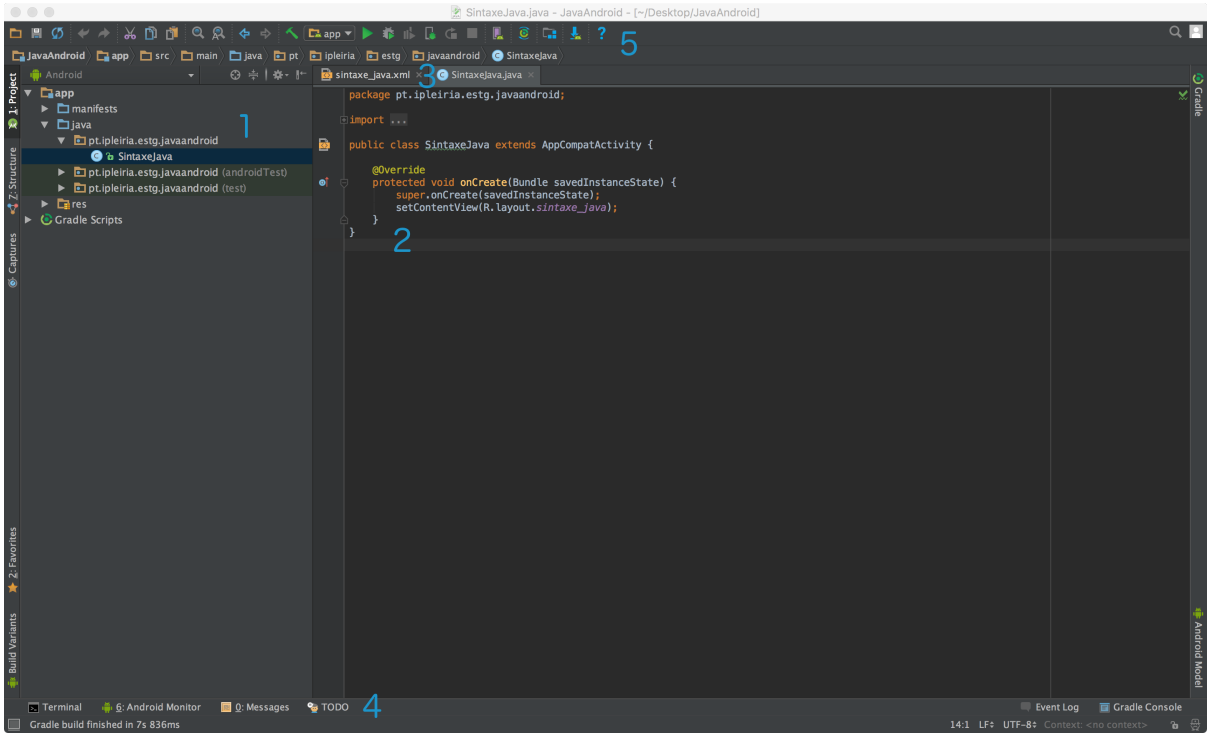
Figura 4 - Configuração da atividade a criar.



1. **Activity Name**: nome da atividade, usado como nome da classe.
2. **Generate Layout File**: podemos criar a atividade sem um *layout* correspondente se não quisermos o *layout* exemplo, neste caso mantemos a opção selecionada.
3. **Layout Name**: nome do ficheiro de layout.

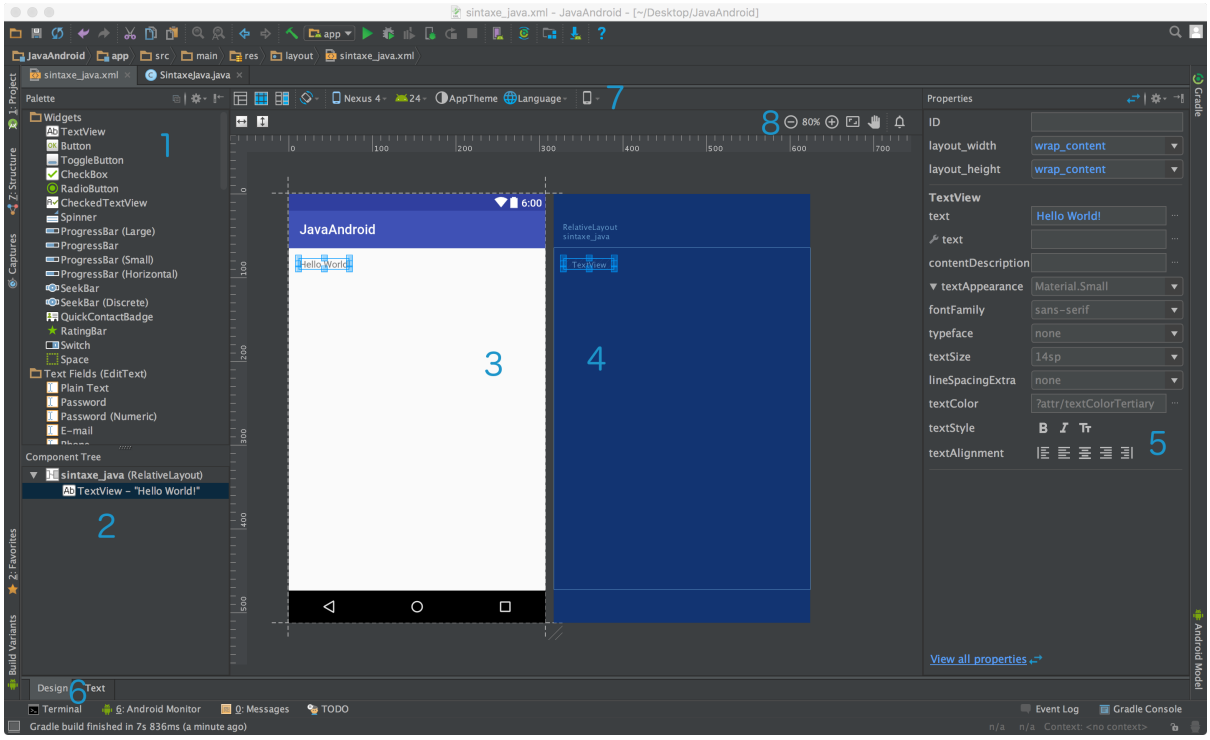
Terminadas todas as configurações, o IDE irá gerar os ficheiros necessário, iniciar o projeto e executar as tarefas que levam à composição de uma aplicação *Android*. O processo de geração pode demorar algum tempo, mas ao terminar deverá, automaticamente, apresentar dois ficheiros abertos e uma estrutura de pastas (e ficheiros) do projeto na área lateral esquerda.

Figura 5 - Área de código/interface principal do IDE.



1. Área com listas de recursos (ficheiros) existentes no projeto. Permite o acesso aos ficheiros de código, configuração e layout do projeto.
2. Área principal, neste exemplo, com o código da atividade criada.
3. Separadores para escolha de ficheiros, podemos ver dois ficheiros abertos (sintaxe_java.xml e SintaxeJava.java)
4. Janelas de output/mensagens, onde são apresentados erros de compilação, mensagens geradas pelo dispositivo ou pelo simulador e resultados dos comandos executados pelo IDE.
5. Barras de ferramentas com opções para executar, testar, compilar, etc., e barra de navegação que permite navegar na estrutura de pastas do projeto.

Figura 6 - Área de desenho de IU/interface principal do IDE.



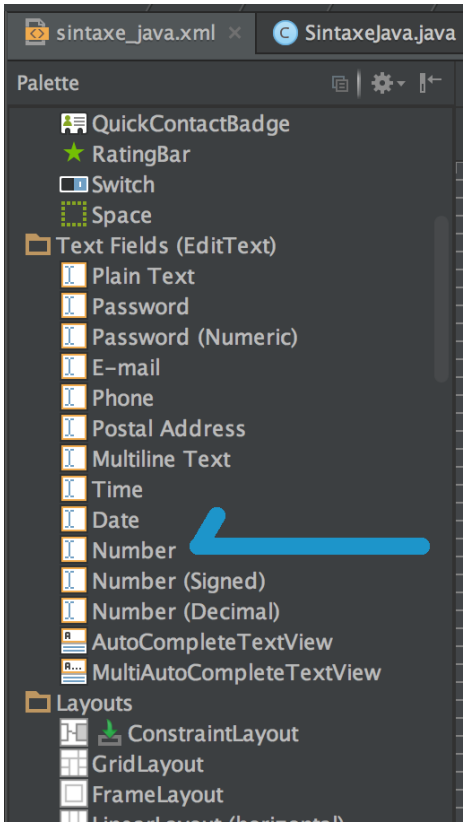
1. Paleta de componentes, com a lista de todos os componentes (*widgets*) disponíveis. Podem ser arrastados para a área de edição da interface gráfica (3). Alguns componentes não representam *widgets* diferentes, mas sim configurações diferentes para a mesma *widget*, como é o caso dos componentes na secção ***Text Fields***, que são todos do tipo *EditText* mas que já estão pré-configurados (ex.: ***Plain Text*** é uma *EditText* que permite introdução de qualquer tipo de texto, ***Password*** é uma *EditText* para introdução de passwords; nos dois casos a classe/componente é a mesma).
2. Árvore de componentes já adicionados à aplicação.
3. Área de edição/desenho da interface gráfica.
4. Área de estado do layout, mostra o estado atual do layout e da relação entre os vários componentes.
5. Janela de propriedades do componente. Permite configurar, de forma visual, as propriedades do componente que está selecionado.
6. Botões que permitem alternar entre vista de desenho (apresentada na figura) ou vista de código, que mostra o XML que constitui a interface gráfica. O desenvolvimento da interface gráfica implicará a troca constante entre as duas visualizações.
7. Opções para teste da interface gráfica e configurações de simulador e vista de desenho.

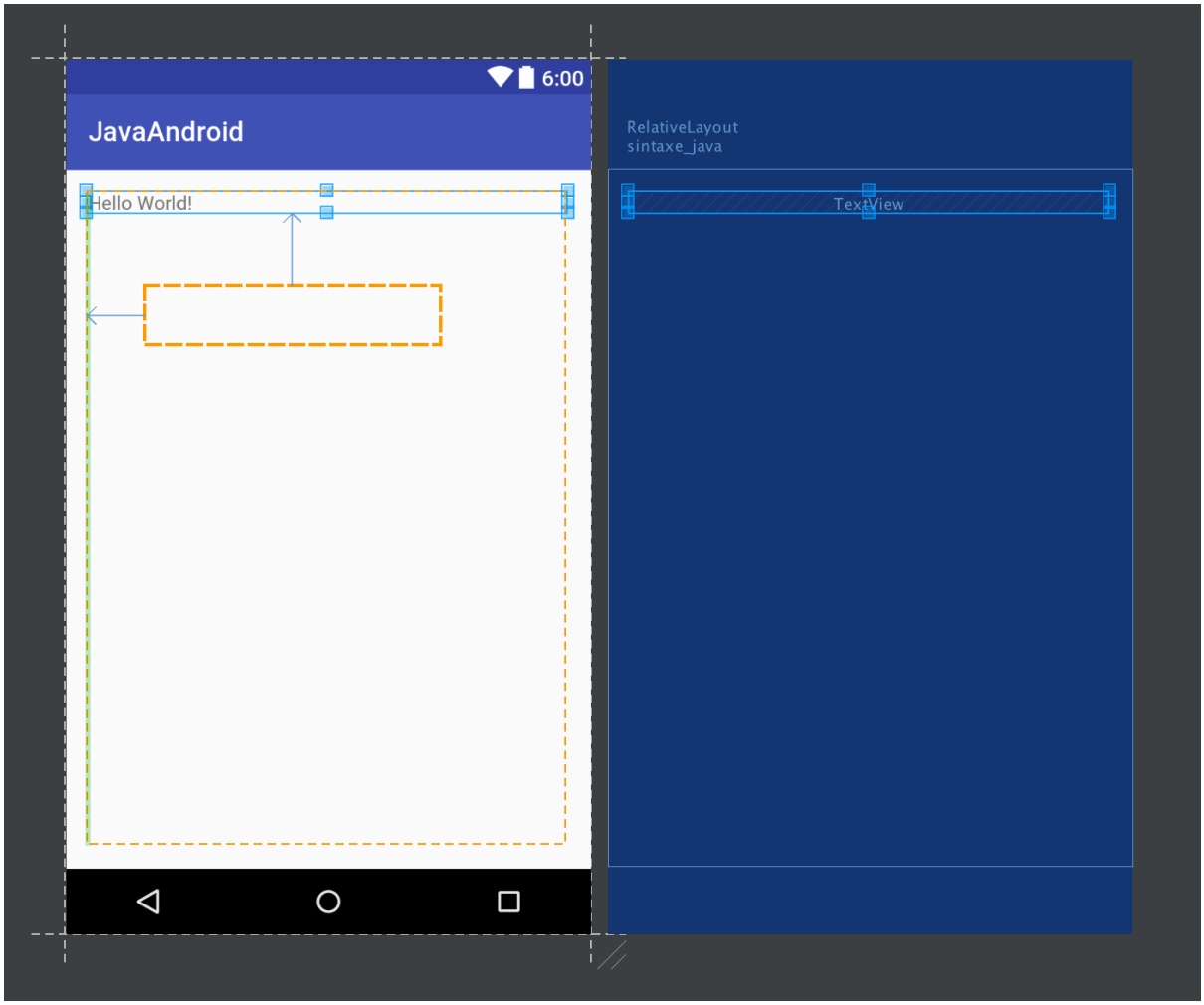
Com a utilização de um modelo para criação da aplicação, e como a nossa aplicação inicial é simples, a interface gráfica está quase completa.

Já temos um campo de apresentação de texto, precisamos apenas adicionar um campo de introdução que permite aos nossos utilizadores indicarem o número secreto a adivinhar.

Para isso basta arrastar um componente ***Number***, disponível na secção ***Text Fields*** da paleta de componentes.

Assim, vamos arrastar o componente e coloca-lo abaixo do texto já presente no ecrã de desenho.





Ao adicionarmos um novo componente, é-nos apresentado o local onde o componente irá ficar e, dependendo da área onde deixamos o rato, são apresentadas setas horizontais e verticais que indicam a que outro componente será registada a relação de posicionamento. A margem verde, visível no lado esquerdo, indica o ponto de referência para a origem do componente.

Num layout relativo, como o que estamos a usar (por termos usado um modelo de atividade vazio) todos os componentes são colocados relativamente a outro componente. O primeiro é posicionado relativo ao layout, que neste caso equivale à área do ecrã com as devidas margens. Mais tarde iremos abordar o posicionamento de componentes e os diferentes tipos de layout existentes.

Falta apenas alterar a propriedade largura dos dois componentes (*TextView* e *EditText*) para o valor "***match_parent***", fazendo com que ocupem a largura do ecrã. E colocar a mensagem "***Adivinhe o Número Secreto!***" no primeiro componente.

Com a interface gráfica desenha, é necessário implementar o código do jogo. Passe para o ficheiro de código da atividade SintaxeJava.java e adicione o código que se encontra na página seguinte.

Código para executar

```
//Colocar no topo da classe, antes do método onCreate()
private String mensagemBase = "Adivinhe o Número Secreto!";
private int numeroSecreto;
private int tentativas = 5;
private TextView txtResultado;
private EditText edtNumeroInserido;

// Colocar dentro do método onCreate(), depois das duas linhas que já se encontram nesse método.
txtResultado = (TextView) findViewById(R.id.resultado);
edtNumeroInserido = (EditText) findViewById(R.id.numero);

edtNumeroInserido.setOnEditorActionListener(new TextView.OnEditorActionListener() {

    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        if (actionId == EditorInfo.IME_ACTION_DONE ||
            (event != null && event.getAction() == KeyEvent.ACTION_DOWN)) {

            if (tentativas == 0) {
                txtResultado.setText("Acabaram as tentativas!");
                return true;
            }

            if (edtNumeroInserido.getText().length() > 0) {
                tentativas--;

                int valor = Integer.parseInt(edtNumeroInserido.getText().toString());
                if (valor > numeroSecreto) {
                    txtResultado.setText(mensagemBase + " - O número é inferior");
                } else if (valor < numeroSecreto) {
                    txtResultado.setText(mensagemBase + " - O número é superior");
                } else {
                    txtResultado.setText("Parabéns! Acertou no número.");
                }
            }
            return true;
        }
        return false;
    }
});
```