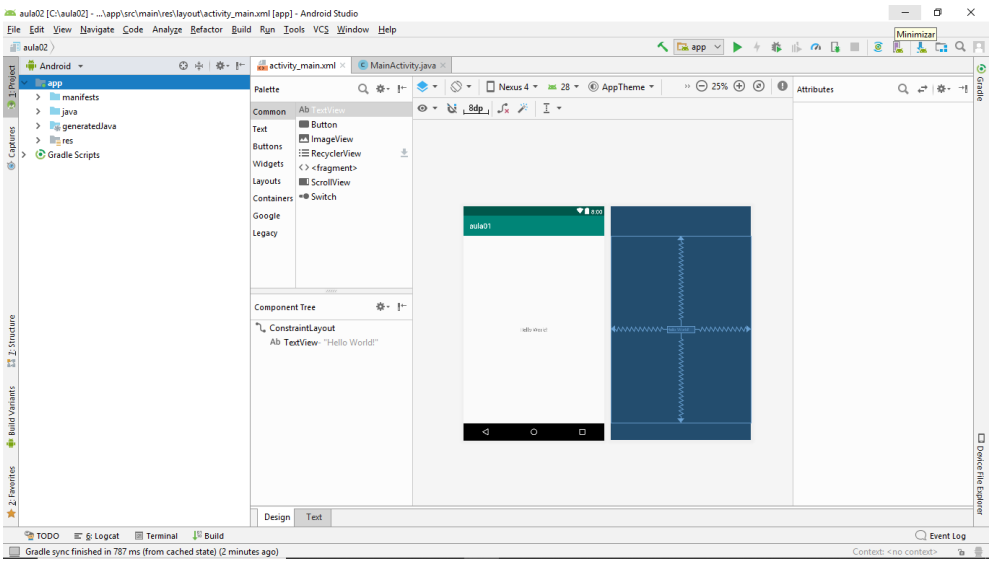
1. Aula 2

1.1. A interface do usuário

A janela principal do Android Studio é composta de diversas áreas.

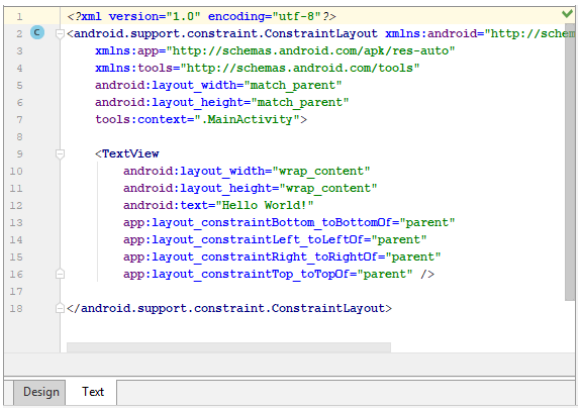


Vamos conhecer as principais áreas do programa.

A **barra de ferramentas** permite executar diversas ações, incluindo executar aplicativos e inicializar ferramentas do Android.

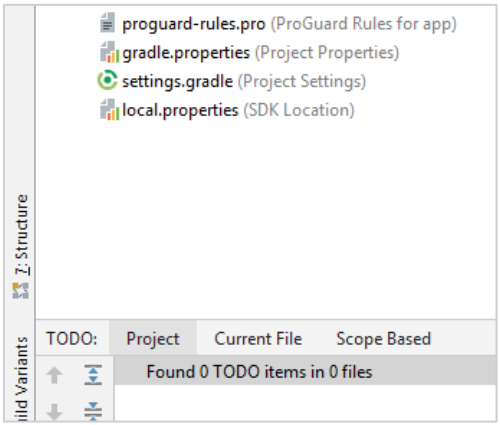
A **barra de navegação** ajuda na navegação pelo projeto e na abertura de arquivos para edição. Ela oferece uma visualização mais compacta da estrutura visível na janela Project.

A **janela do editor** é o local em que você cria e modifica código. Dependendo do tipo de arquivo atual, o editor pode mudar. Por exemplo, ao visualizar um arquivo de layout, o editor abre o Editor de layout.

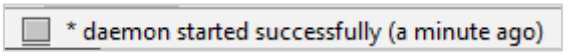


A **barra de janela de ferramentas** fica fora da janela do IDE, contendo os botões que permitem expandir ou recolher a janela de cada ferramenta.

A **janela das ferramentas** dá acesso a tarefas específicas, como gerenciamento de projetos, busca, controle de versão e muitos outros. Você pode expandi-las e recolhê-las.



A **barra de status** mostra o status do projeto e do próprio IDE, além de advertências e mensagens.



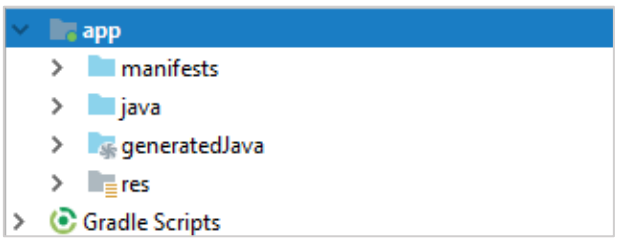
1.2. Estrutura do projeto

Todos os arquivos da compilação podem ser vistos no nível superior em Gradle Scripts e cada módulo de aplicativo contém as pastas a seguir:

Manifestos: contém o arquivo AndroidManifest.xml.

Java: contém os arquivos de código-fonte do Java, incluindo o código de teste do JUnit.

Recursos: contém todos os recursos que não são código, como layouts XML, strings de IU e imagens em bitmap.



MANIFESTS

Todo aplicativo tem que ter um arquivo AndroidManifest.xml (precisamente com esse nome) no diretório raiz. O arquivo de manifesto apresenta informações essenciais sobre o aplicativo ao sistema Android, necessárias para o sistema antes que ele possa executar o código do aplicativo.

Entre outras coisas, o arquivo do manifesto serve para:

Nomear o pacote Java para o aplicativo. O nome do pacote serve como identificador exclusivo para o aplicativo.

Descrever os componentes do aplicativo que abrangem atividades, serviços, receptores de transmissão e provedores de conteúdo que compõem o aplicativo. Ele também nomeia a classe que implementa cada um dos componentes e publica suas capacidades, como as mensagens Intent que podem lidar. Essas declarações informam ao sistema Android os componentes e as condições em que eles podem ser inicializados.

Determinar os processos que hospedam os componentes de aplicativo.

Declarar as permissões que o aplicativo deve ter para acessar partes protegidas da API e interagir com outros aplicativos. Ele também declara as permissões que outros devem ter para interagir com os componentes do aplicativo.

Listar as classes Instrumentation que fornecem geração de perfil e outras informações durante a execução do aplicativo. Essas declarações estão presentes no manifesto somente enquanto o aplicativo está em desenvolvimento e são removidas antes da publicação do aplicativo.

Declarar o nível mínimo da Android API que o aplicativo exige.

Listar as bibliotecas às quais o aplicativo deve se vincular.

JAVA:

Pasta com dois Pacotes:

1º. MainActivity (onde iremos criar nossos arquivos java);

2º. Application Test (onde faremos testes da nossa Aplicação).

Drawable

Recursos gráficos que podem ser bitmaps ou XML.

arquivos Bitmap (.png, .jpg, .gif) ou arquivos XML que são compilados nos seguintes resources:

arquivos Bitmap

9-Patch (bitmaps redimensionáveis)

listas de estado

Shapes (formas)

Animações (frame animations)

Outros tipos.

LAYOUT

Você pode definir vários layouts para sua UI e alterná-los de acordo com o estado da aplicação.

Arquivos XML que definem a sua UI (user interface).

MIPMAP

Arquivos “drawable” especificamente para ícones de aplicações de diferentes tipos de densidade.

VALUES

Arquivos XML que contêm valores, como strings, inteiros e cores. Dentro dessa pasta, você deve nomear cada arquivo XML com o nome do seu resource e dessa forma irá acessar com uma subclasse de R. Por exemplo, o arquivo string.xml será acessado por R.string. Abaixo algumas convenções:

arrays.xml para Arrays.

colors.xml para valores de cores.

dimens.xml para dimensões.

strings.xml para todas as strings.

styles.xml para estilos.

Gradle

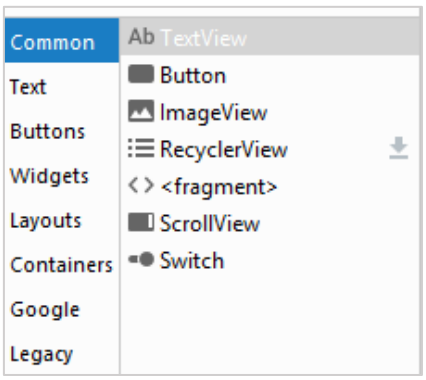
Desde que o Android Studio se tornou a ferramenta oficial do Google para desenvolvimento Android, o Gradle veio como principal aliado para ajudar no gerenciamento do projeto e controle de dependências.

Para entender como funciona o Gradle, antes precisamos entender um pouco sobre como funciona o processo de build de um aplicativo Android.

O build é o processo de compilação, ou seja, construção de recursos que vão integrar o projeto.

Quando executamos o Gradle para construir seus projeto e módulos, o processo do build é executado em segundo plano.

1.3. Componentes do Android Studio



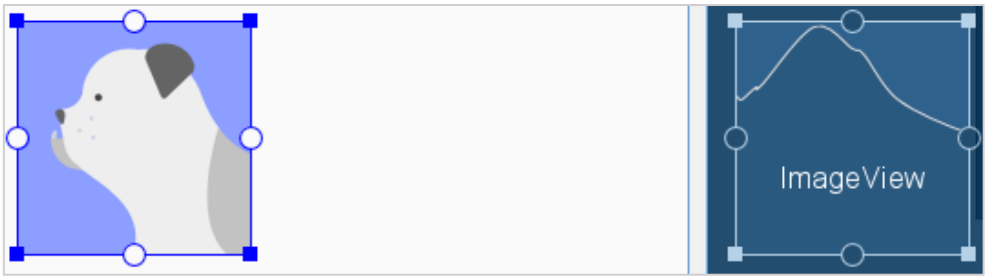
A primeira aba é a **Common** (Comum), onde ficam os componentes indicados como mais usados.

Vamos conhecer alguns elementos:

TextView: uma caixa de saída de texto, utilizada para textos, títulos ou para passar informações.

Button: um botão comum, utilizado em menus de opções.

ImageView: uma imagem de demonstração, como ícones e fotos.

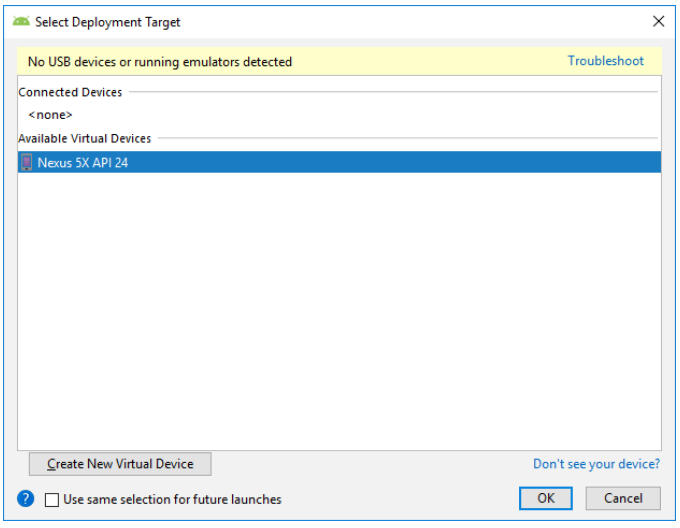


1.4. Emulador

O Android Emulator permite executar aplicativos Android em um ambiente simulado dentro do Windows, simula um dispositivo e o exibe no seu computador de desenvolvimento. Com ele, você pode criar protótipos, desenvolver e testar aplicativos do Android sem usar um dispositivo de hardware. O emulador é compatível com celulares e tablets Android, com o Android Wear e com dispositivos Android TV. Ele vem com tipos de dispositivo predefinidos para que você comece rapidamente e, além disso, também é possível criar suas próprias definições de dispositivo e aparências para o emulador.

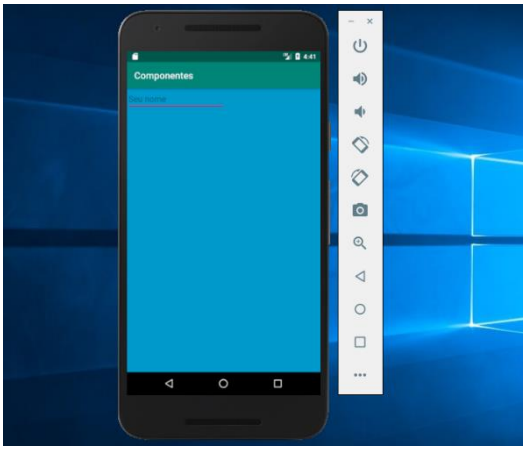
Barra de ferramentas

O botão **Run** (app), destacado com um contorno vermelho, serve para executar o projeto.



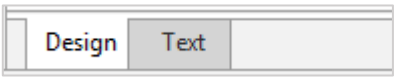
O Android Emulator é uma ferramenta rápida, eficaz e repleta de recursos. Ele pode transferir informações com mais rapidez do que um dispositivo de hardware conectado, o que agiliza o processo de desenvolvimento. O recurso de vários núcleos permite que o emulador aproveite os processadores de vários núcleos do seu computador de desenvolvimento para melhorar ainda mais o desempenho do emulador.

Você interage com o emulador da mesma forma que faria com um dispositivo de hardware, mas usando o mouse e o teclado, além dos botões e controles do emulador. O emulador é compatível com botões e touchscreens de hardware virtual, incluindo operações com dois dedos, pads direcionais, trackballs, rodas e diversos sensores. É possível redimensionar dinamicamente a janela do emulador conforme a necessidade, aumentar e reduzir o zoom, alterar a orientação e até registrar uma captura de tela.



1.5. Modos de visualização

O Android Studio possui o modo de visualização



Modo Design: O modo design permite acrescentar diversos recursos como: Component Tree, no canto inferior esquerdo mostra a hierarquia de visualizações do layout. Nesse caso, a visualização raiz é um ConstraintLayout que contém apenas um objeto TextView. Utilizar a “Pallete”que permite adicionar vários recursos como: elementos para entrada de texto simples, com senha, para validar telefone, e-mail, entre outros, a categoria “Buttons” que permite adicionar elementos de formulário como Checkbox, radiobutton, entre outros.

Modo Text: O modo texto permite trabalhar com o código da página, se já foram aplicados recursos no modo design, estes serão apresentados no código, onde de forma mais prática poderemos fazer alterações em suas propriedades, como tamanho e cor.