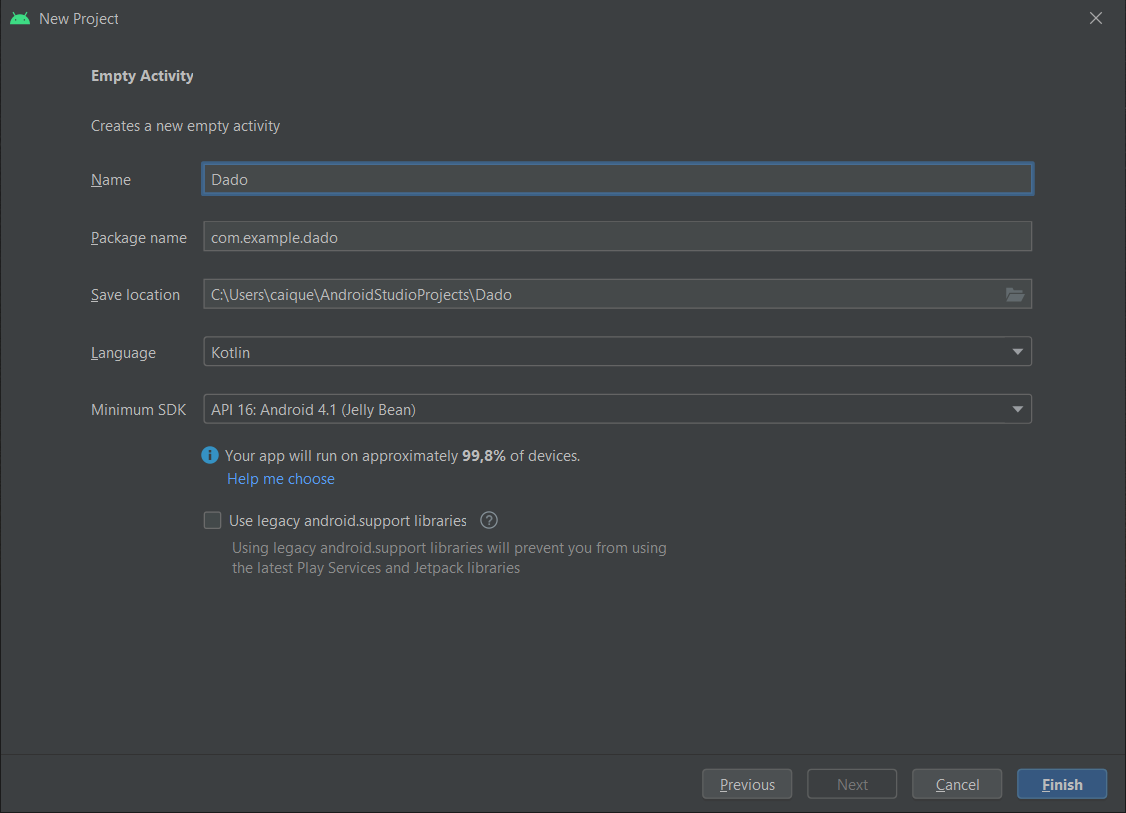
# Etapa 1: Examinar MainActivity

MainActivityé um exemplo de um Activity. Uma  Activity é uma classe principal do Android que desenha uma interface de usuário (IU) do aplicativo Android e recebe eventos de entrada. Quando seu aplicativo é iniciado, ele inicia a atividade especificada no arquivo AndroidManifest.xml.

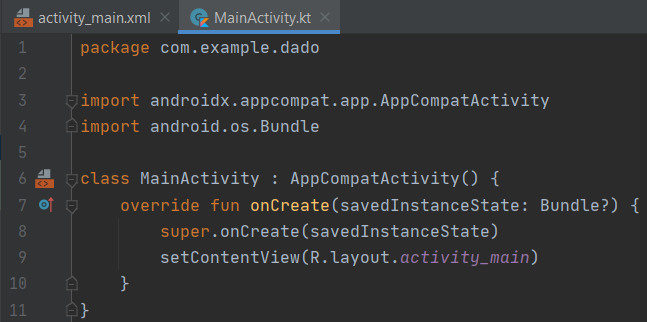
Muitas linguagens de programação definem um método principal que inicia o programa. Os aplicativos Android não têm um método principal. Em vez disso, o AndroidManifest.xmlarquivo indica que MainActivitydeve ser iniciado quando o usuário toca no ícone do iniciador do aplicativo. Para iniciar uma atividade, o sistema operacional Android usa as informações no manifesto para configurar o ambiente para o aplicativo e construir o MainActivity. Em seguida, MainActivityele faz algumas configurações.

Cada atividade possui um arquivo de layout associado. A atividade e o layout são conectados por um processo conhecido como *inflação de layout* . Quando a atividade começa, as visualizações definidas nos arquivos de layout XML são transformadas (ou "infladas" em) objetos de visualização Kotlin na memória. Assim que isso acontecer, a atividade pode desenhar esses objetos na tela e também modificá-los dinamicamente.

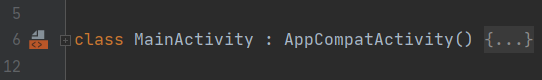
1. No Android Studio, selecione **Arquivo> Novo> Novo projeto** para criar um novo projeto. Use a atividade Vazia e clique em **Avançar** .
2. Chame o projeto **DiceRoller** e verifique todos os outros valores para o nome do projeto local do projeto. Certifique-se de que a opção "Usar artefatos AndroidX" esteja marcada. Clique em **Concluir** .



1. No painel **Projeto> Android** , expanda **java> com.example.android.diceroller** . Clique duas vezes em **MainActivity** . O editor de código mostra o código em MainActivity.



1. Abaixo do nome do pacote e das instruções de importação está a declaração de classe para MainActivity. A MainActivityaula se estende AppCompatActivity.



**AppCompatActivity** é uma subclasse de **Activity** que oferece suporte a todos os recursos modernos do Android, ao mesmo tempo que oferece compatibilidade com versões anteriores do Android. Para disponibilizar seu aplicativo para o maior número possível de dispositivos e usuários, sempre use **AppCompatActivity**.

1. Observe o onCreate()método. As atividades não usam um construtor para inicializar o objeto. Em vez disso, uma série de métodos predefinidos (chamados "métodos de ciclo de vida") são chamados como parte da configuração da atividade. Um desses métodos de ciclo de vida é onCreate(), que você sempre substitui em seu próprio aplicativo. Você aprenderá mais sobre os métodos de ciclo de vida em um codelab posterior.

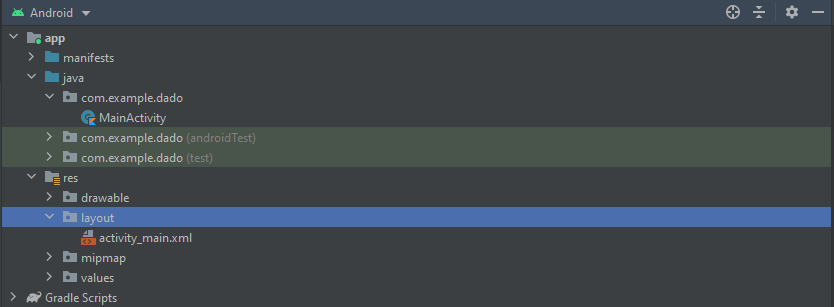
Em onCreate(), você especifica qual layout está associado à atividade e aumenta o layout. O setContentView()método faz ambas as coisas.

override fun onCreate(savedInstanceState: Bundle?) {  
   super.onCreate(savedInstanceState)  
   setContentView(R.layout.activity\_main)  
}  
O setContentView()método faz referência ao layout usando R.layout.activity\_main, que na verdade é uma referência de inteiro. A Rclasse é gerada quando você constrói seu aplicativo. A Rclasse inclui todos os ativos do aplicativo, incluindo o conteúdo do resdiretório.

Nesse caso, R.layout.activity\_mainrefere-se à Rclasse gerada , à layoutpasta e ao activity\_main.xmlarquivo de layout. (Os recursos não incluem extensões de arquivo.) Você se referirá a muitos dos recursos do aplicativo (incluindo imagens, strings e elementos no arquivo de layout) usando referências semelhantes na Rclasse.

# Etapa 2: examine e explore o arquivo de layout do aplicativo

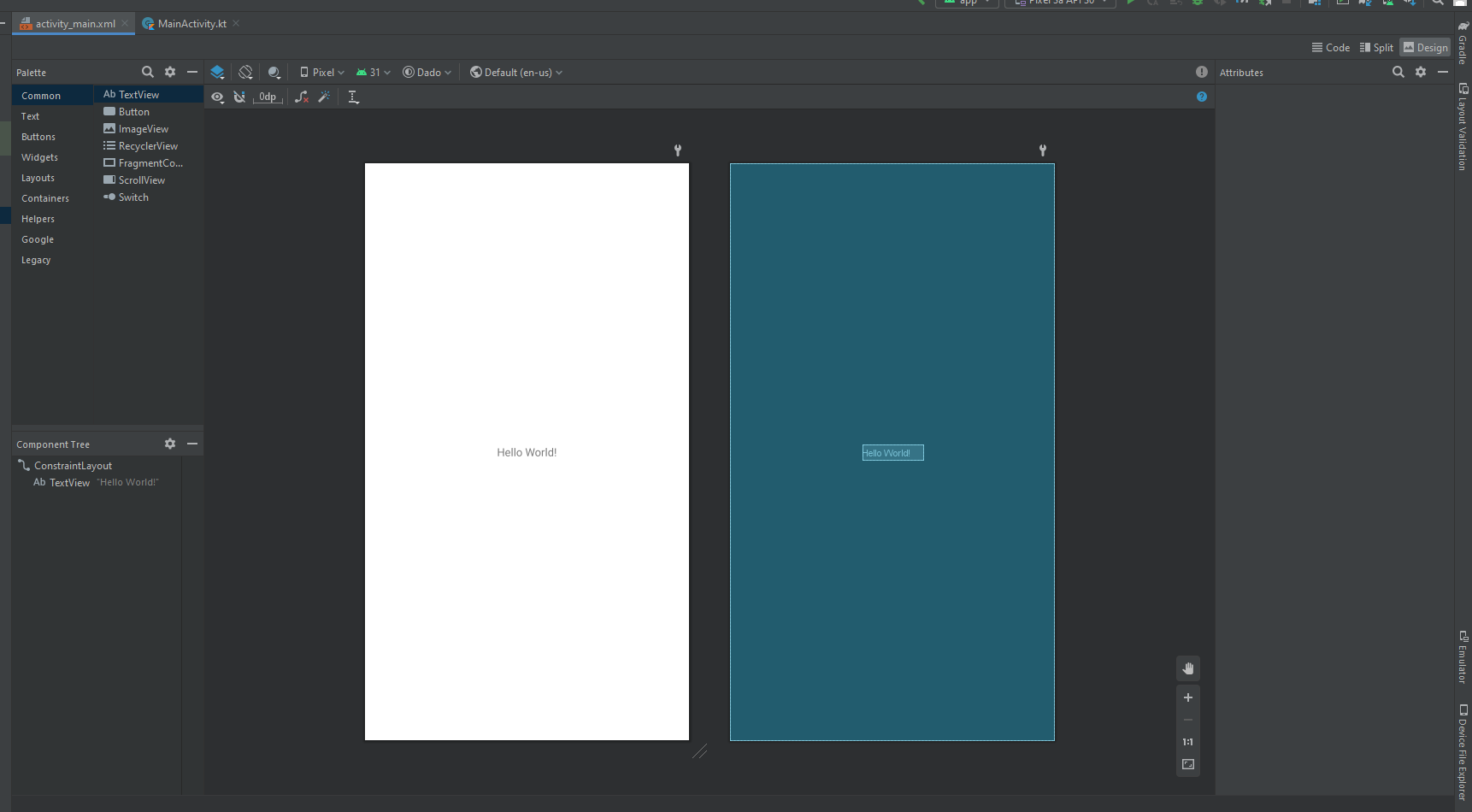
Todas as atividades em seu aplicativo têm um arquivo de layout associado no res/layoutdiretório do aplicativo . Um *arquivo de layout* é um arquivo XML que expressa a aparência real de uma atividade. Um arquivo de layout faz isso definindo visualizações e definindo onde as visualizações aparecem na tela.

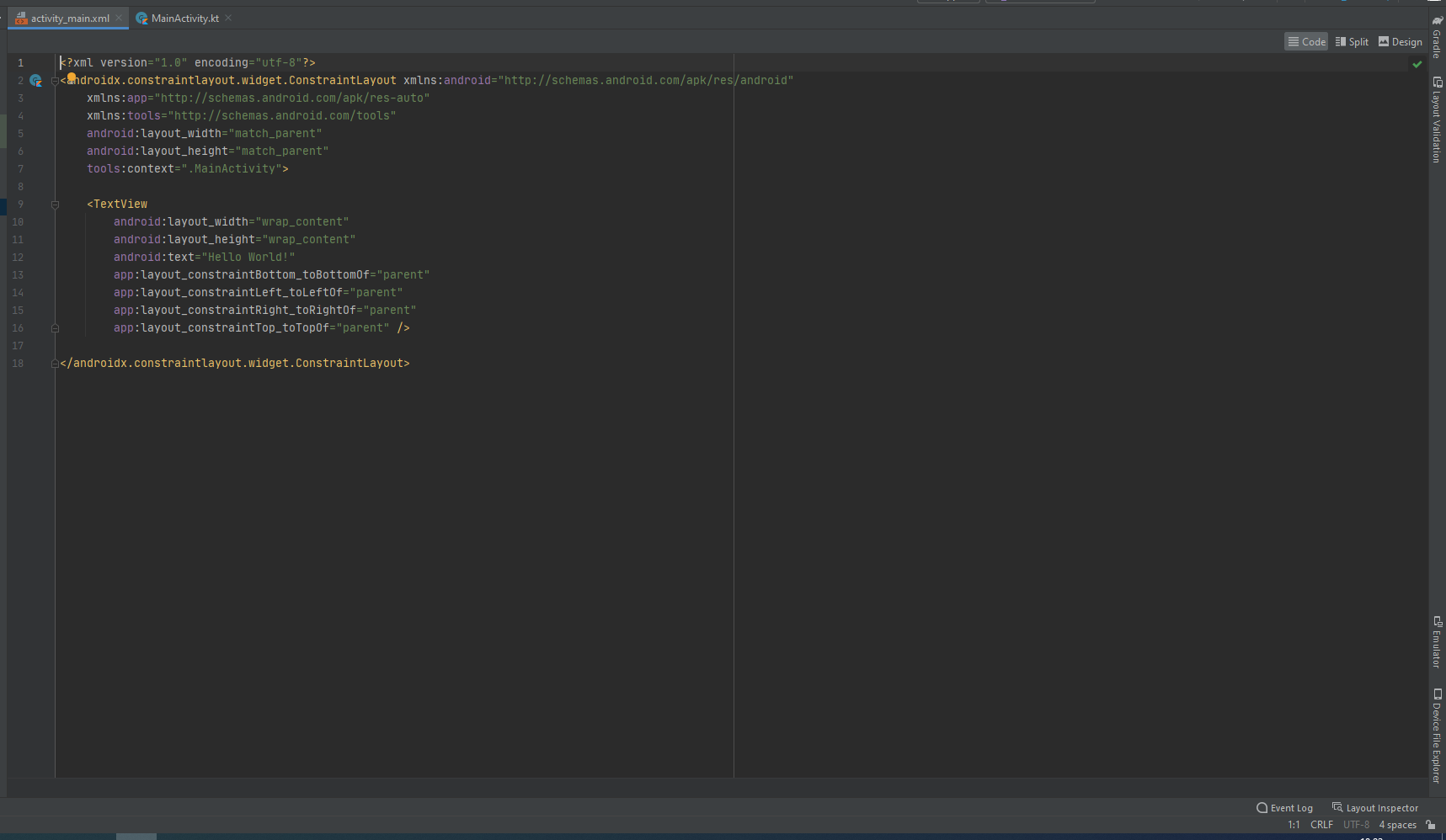


*Visualizações* são coisas como texto, imagens e botões que estendem a Viewclasse. Existem muitos tipos de pontos de vista, incluindo TextView, Button, ImageView, e CheckBox.

Nesta tarefa, você examina e modifica o arquivo de layout do aplicativo.

1. No painel **Projeto> Android** , expanda **res> layout** e clique duas vezes em **activity\_main.xml** . O editor de design de layout é aberto. O Android Studio inclui este editor, que permite construir o layout do seu aplicativo de uma forma visual e visualizar o design do layout. Você aprenderá mais sobre o editor de design em um codelab posterior.
2. Para visualizar o arquivo de layout como XML, clique na guia **Texto** na parte superior.



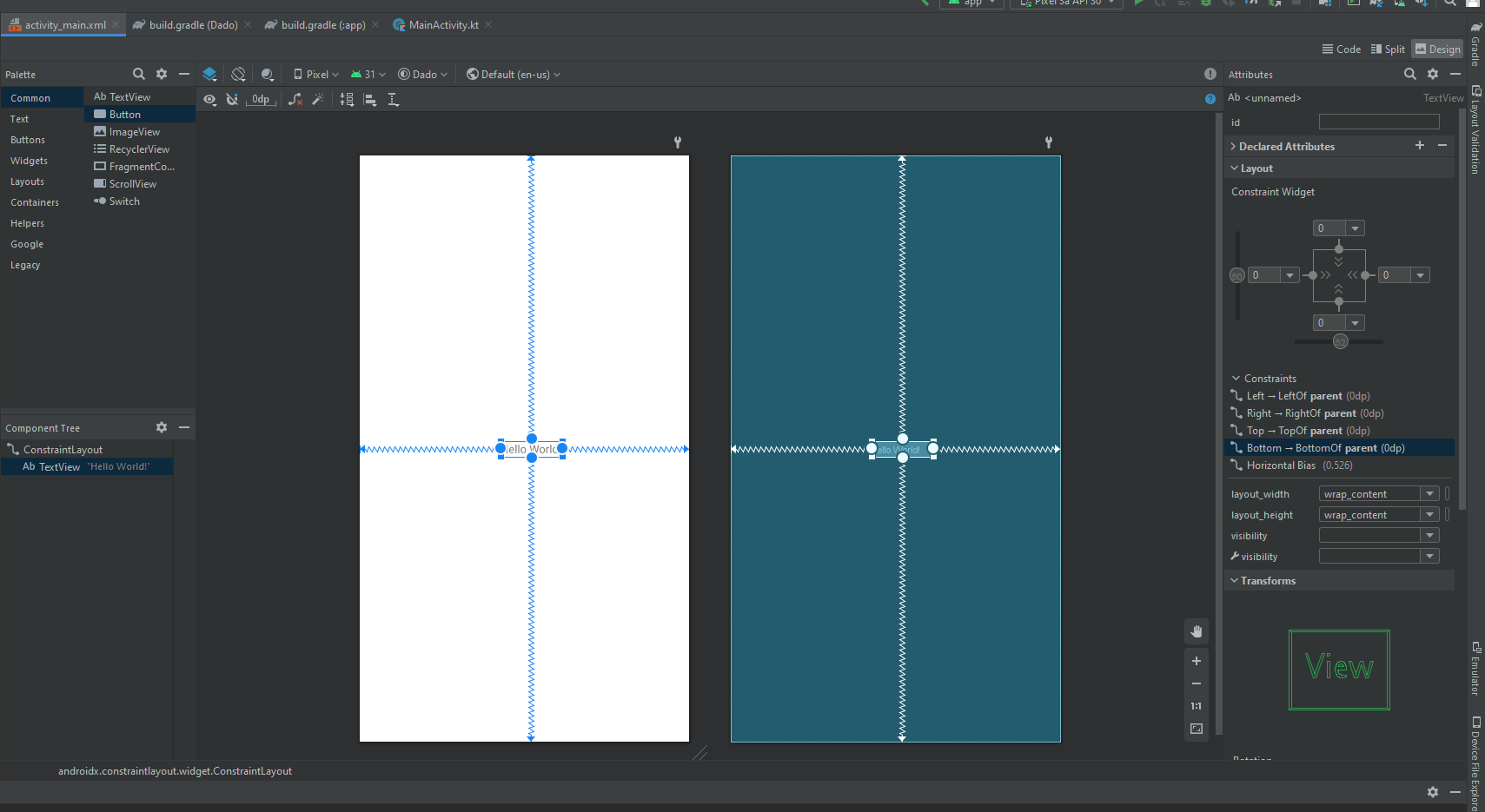


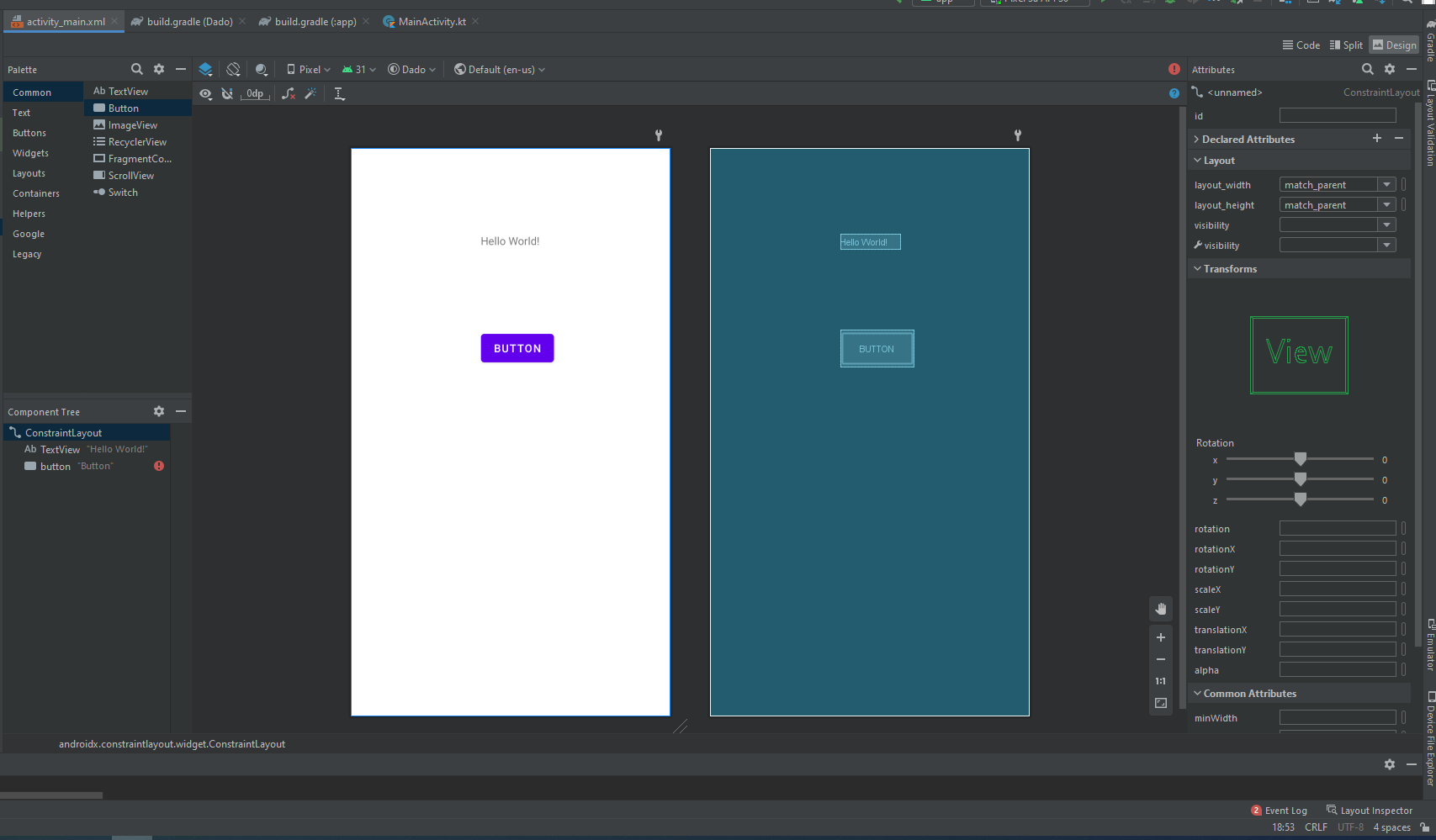
# [4. Tarefa: Adicionar um botão](https://developer.android.com/codelabs/kotlin-android-training-app-anatomy#3)

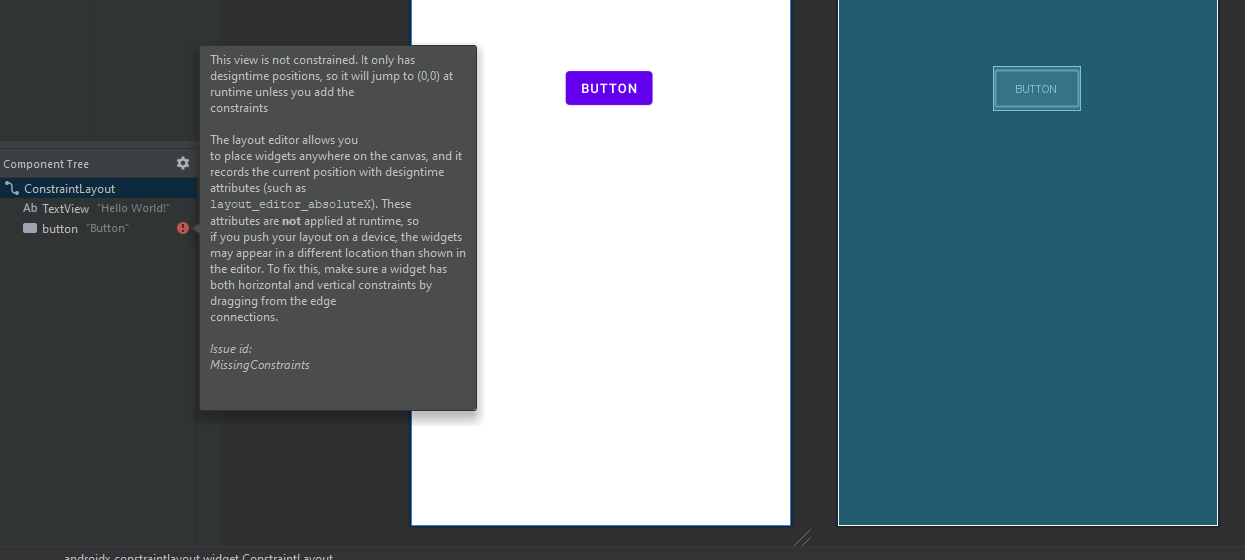
O aplicativo de lançamento de dados não é muito útil sem uma maneira do usuário rolar os dados e ver o que eles rolaram. Para começar, adicione um botão ao layout para lançar os dados e adicione um texto que mostre o valor dos dados que o usuário lançou.

# Etapa 1: adicionar um botão ao layout

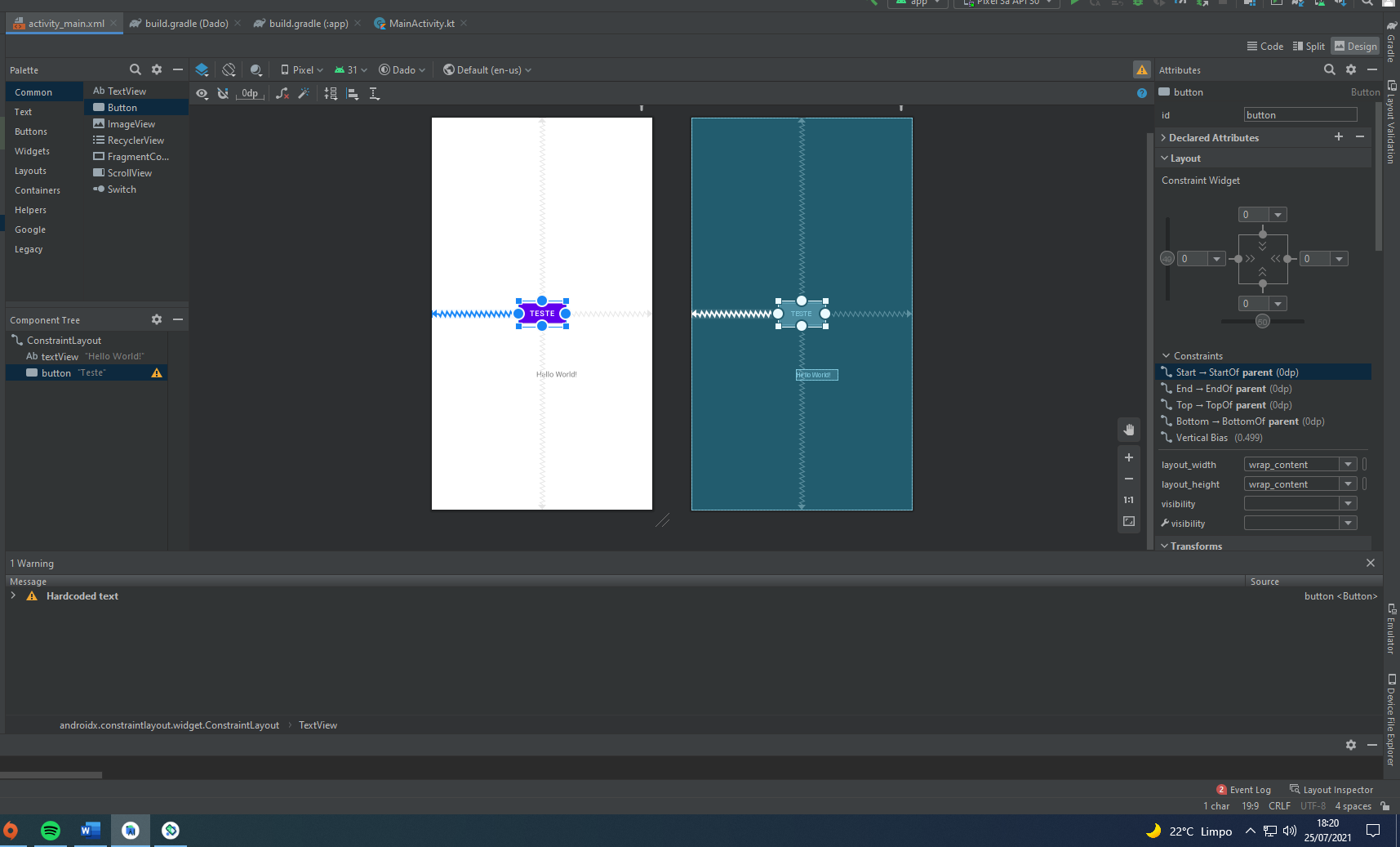
1. Adicione um Button ao layout abaixo da visualização de texto inserindo o **botão <** e pressione Return. É Buttonexibido um bloco que termina com />e inclui os atributos layout\_widthe layout\_height.

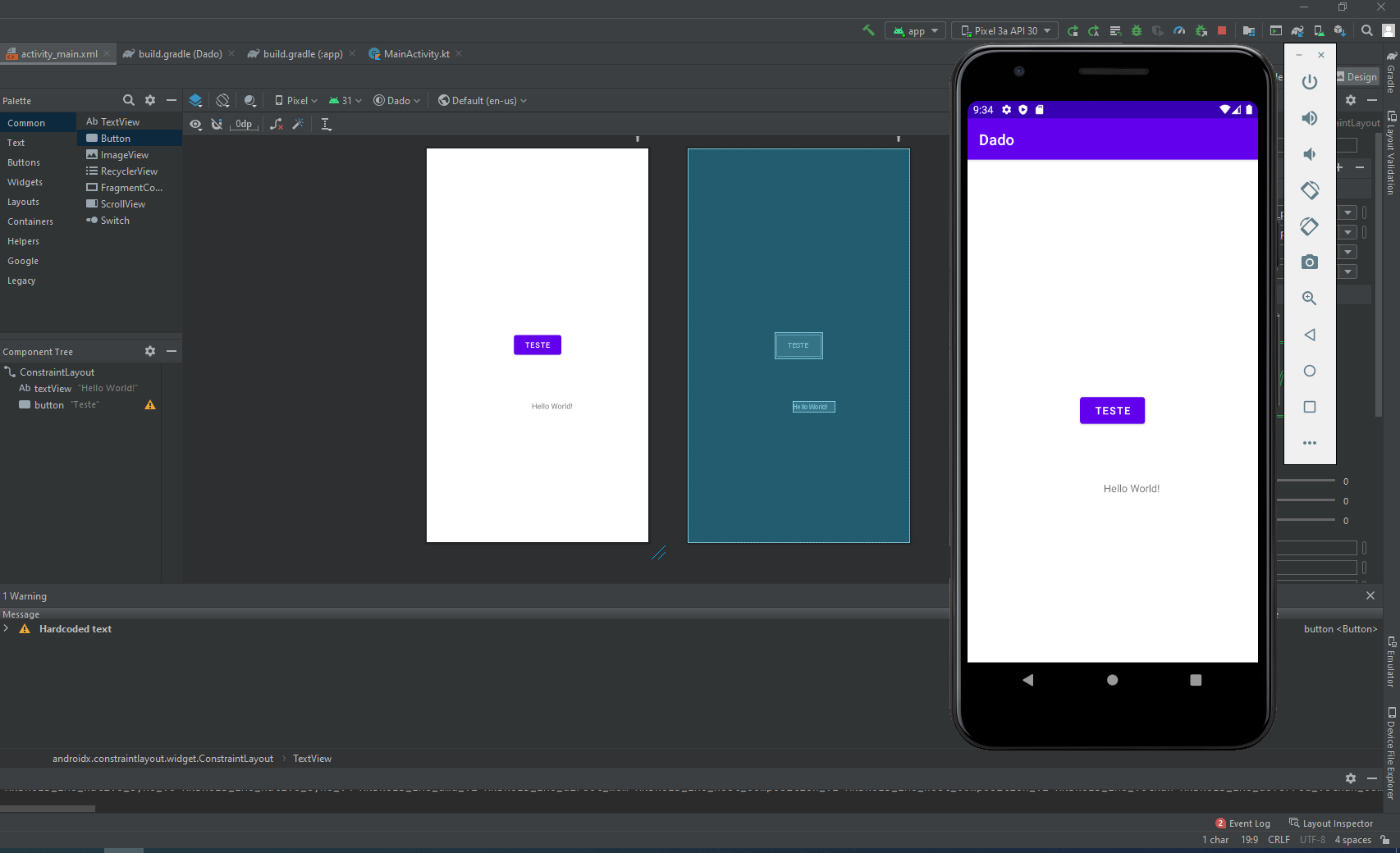






<https://www.youtube.com/watch?v=V20oodliuzA>





# Analisando o codigo



os atributos layout\_widthe layout\_heightcomo "wrap\_content". Com esses valores, o botão tem a mesma largura e altura que o rótulo de texto que contém.

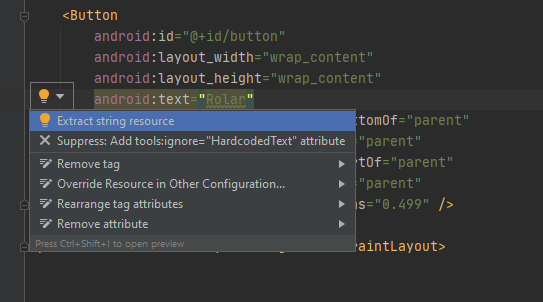
Adicione um android:textatributo ao botão e atribua a ele o valor "Rolar". O elemento Button agora se parece com isto:

Para Buttonvisualizações, o textatributo é o rótulo do botão. No editor de layout, o atributo é destacado em amarelo, o que indica uma dica ou aviso. Nesse caso, o destaque amarelo é porque a string "Roll"está codificada no rótulo do botão, mas a string deve ser um recurso. Você aprenderá sobre os recursos de string na próxima seção.

# Etapa 2: extrair recursos de string

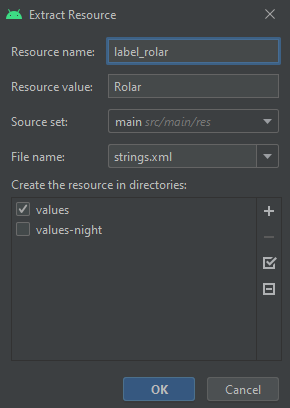
Em vez de codificar strings em seu layout ou arquivos de código, é uma prática recomendada colocar todas as strings de seu aplicativo em um arquivo separado. Este arquivo é chamado strings.xmle está localizado entre os recursos do aplicativo, no res/values/diretório.

Ter as strings em um arquivo separado torna mais fácil gerenciá-las, especialmente se você usar essas strings mais de uma vez. Além disso, os recursos de string são obrigatórios para traduzir e localizar seu aplicativo, porque você precisa criar um arquivo de recurso de string para cada idioma.

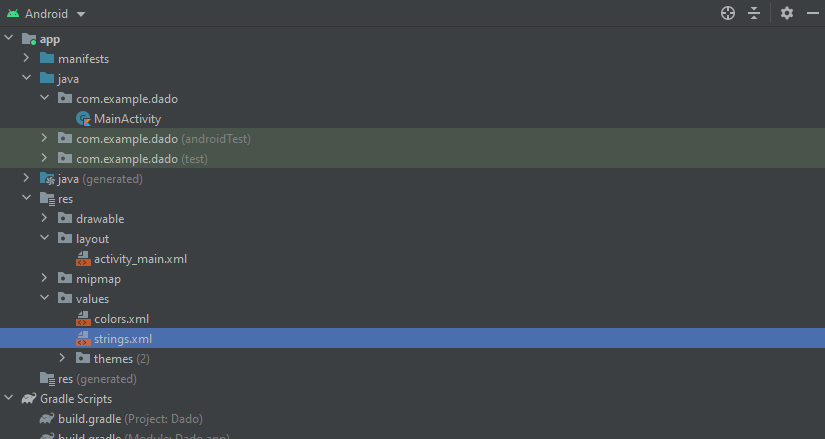


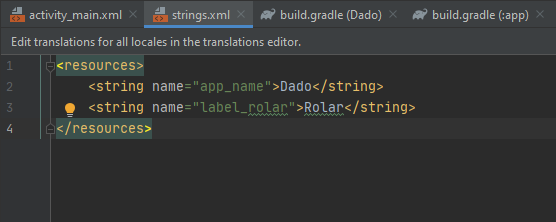
O Android Studio ajuda você a lembrar de colocar suas strings em um arquivo de recurso com dicas e avisos.

1. Clique uma vez na string "Roll" no android:text da <Button>tag.
2. Pressione Alt+Enter( Option+Enterno macOS) e selecione **Extrair recurso de string** no menu pop-up.
3. Insira roll\_labelo **nome** do **recurso** .
4. Clique **OK** . Um recurso de string é criado no res/values/string.xmlarquivo e a string no elemento Button é substituída por uma referência a esse recurso:android:text="@string/roll\_label"

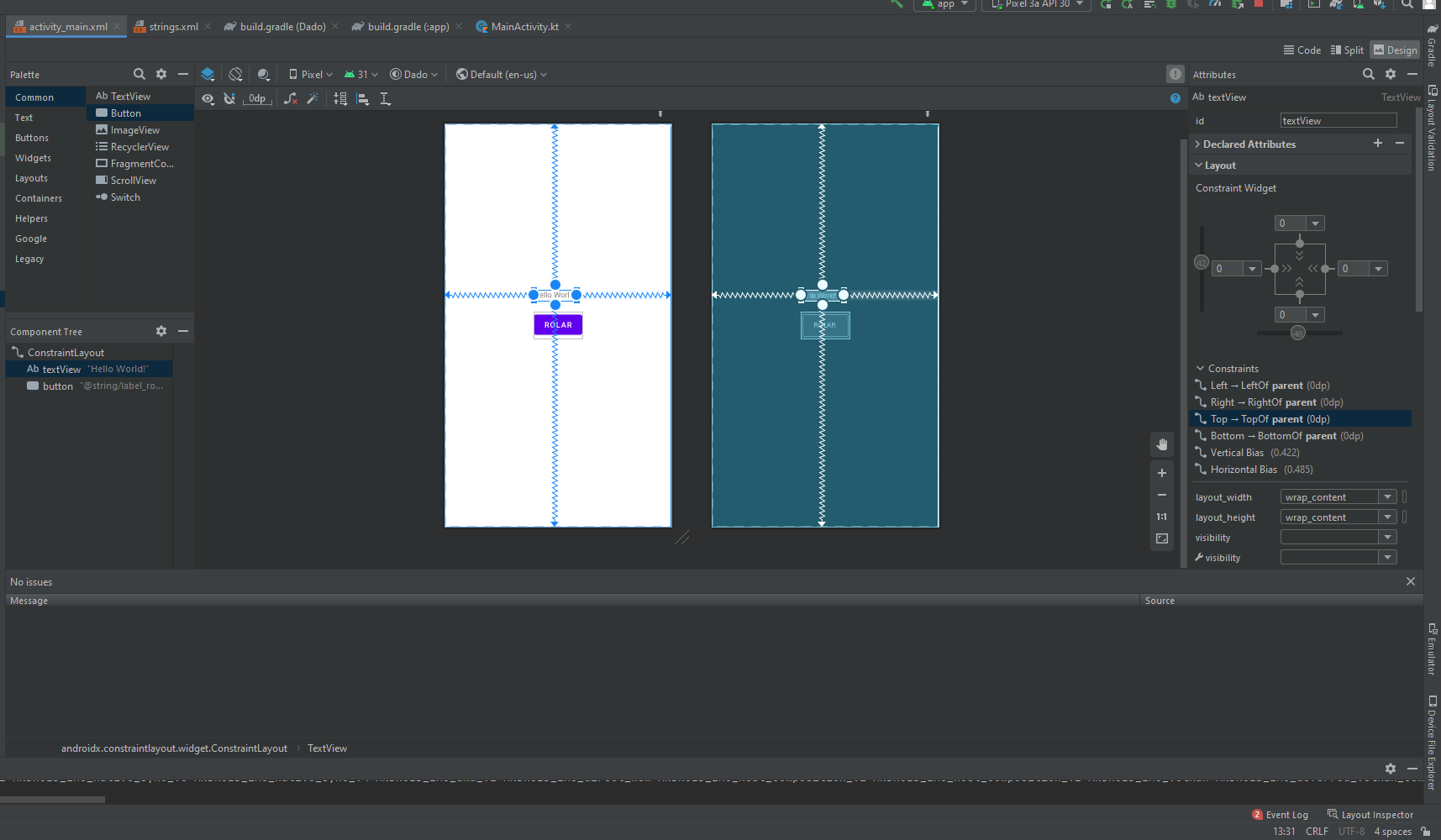


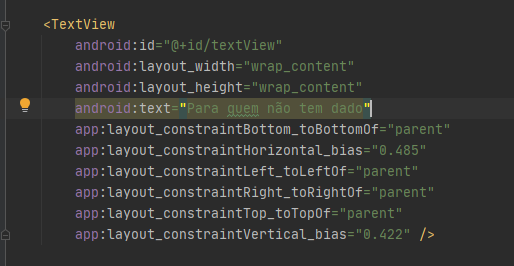
1. No painel **Projeto> Android** , expanda **res> valores** e clique duas vezes em **strings.xml** para ver seus recursos de string no strings.xml:



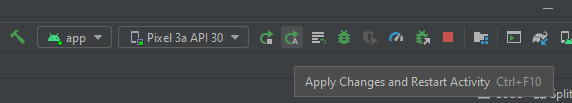


**Dica** : além da string que você acabou de adicionar, o **strings.xml**arquivo também inclui o nome do aplicativo. O nome do aplicativo aparecerá na barra de aplicativos na parte superior da tela se você iniciar o projeto do aplicativo usando o modelo vazio. Você pode alterar o nome do aplicativo editando o **app\_name**recurso.





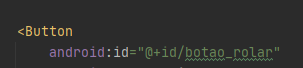




Veja o id do botão

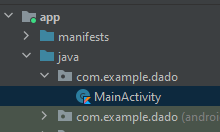


Vamos alterar para botao\_rolar



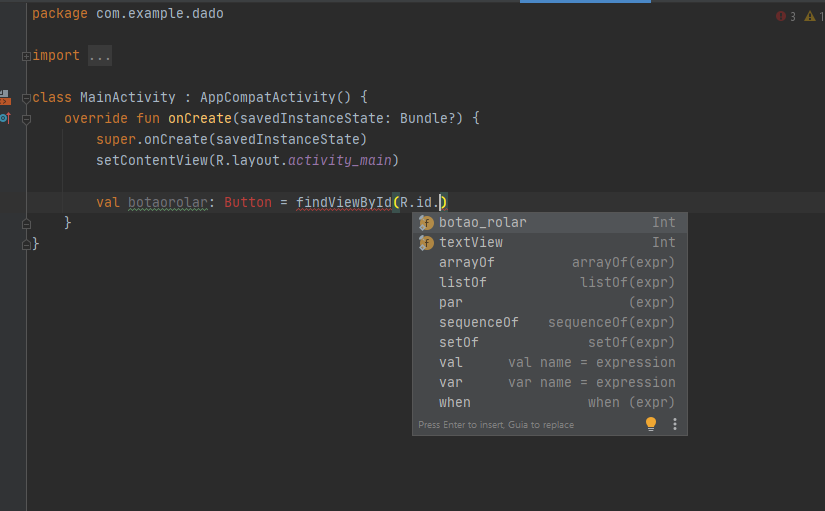
Quando você cria um ID para uma visualização no arquivo de layout XML, o Android Studio cria uma constante de número inteiro com o nome desse ID na Rclasse gerada . Portanto, se você nomear uma visualização roll\_button, o Android Studio gera e cria uma constante de número inteiro chamada roll\_buttonna Rclasse. O "@+id"prefixo para o nome do ID diz ao compilador para adicionar essa constante de ID à classe R. Todos os IDs de vista da propriedade em seu arquivo XML devem ter este prefixo.

1. Abra o MainActivityarquivo Kotlin. Dentro onCreate(), depois setContentView(), adicione esta linha:





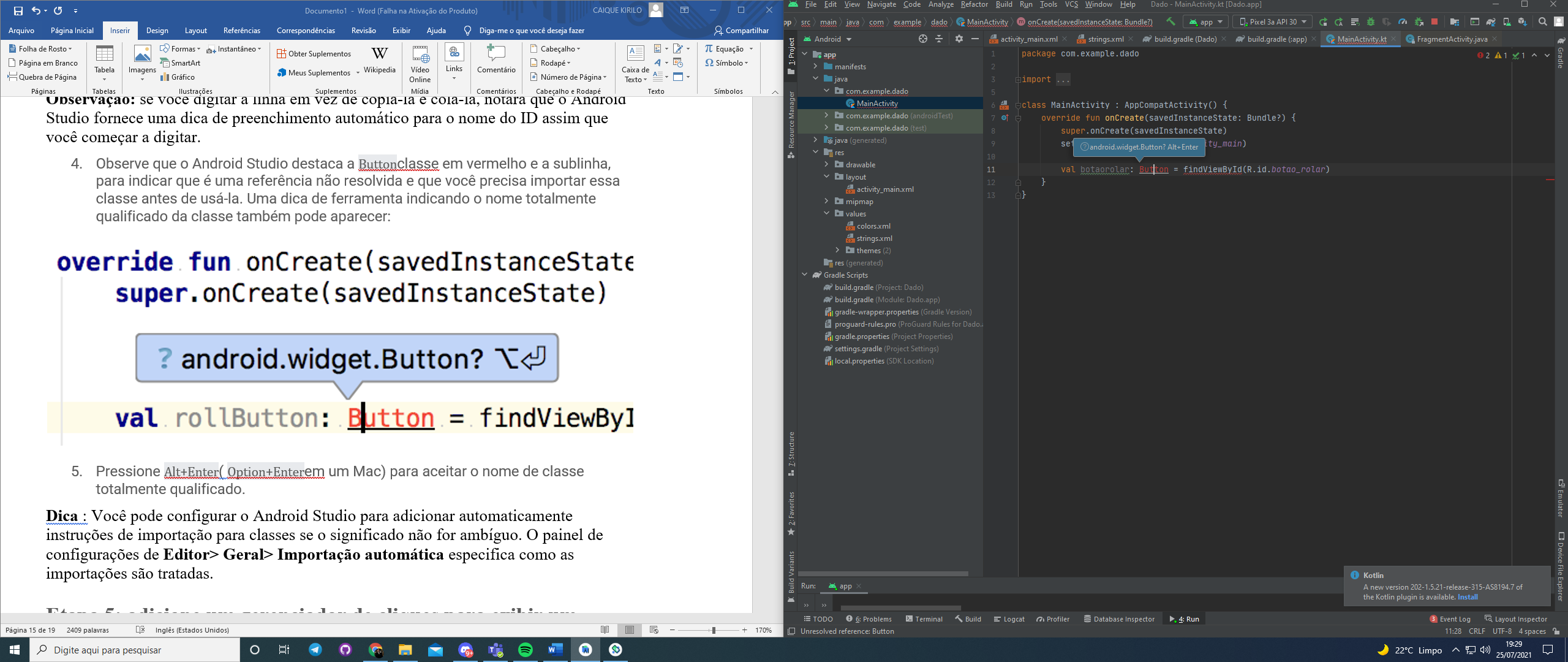
val botaorolar: Button = findViewById(R.id.*botao\_rolar*)



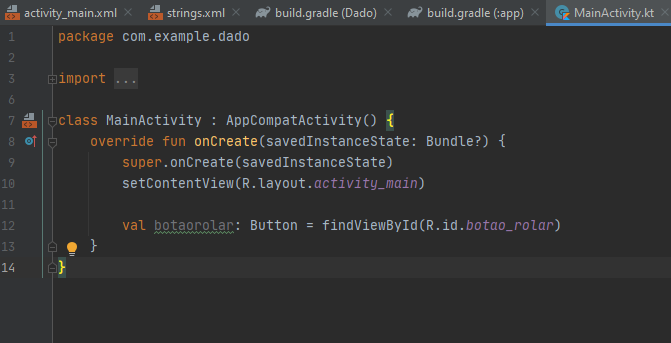
Use o findViewById() para obter uma View para a visualização que você definiu na classe XML.

**Observação:** se você digitar a linha em vez de copiá-la e colá-la, notará que o Android Studio fornece uma dica de preenchimento automático para o nome do ID assim que você começar a digitar.

1. Observe que o Android Studio destaca a classe Button em vermelho e a sublinha, para indicar que é uma referência não resolvida e que você precisa importar essa classe antes de usá-la. Uma dica de ferramenta indicando o nome totalmente qualificado da classe também pode aparecer:



1. Pressione Alt+Enter( Option+Enterem um Mac) para aceitar o nome de classe totalmente qualificado.



**Dica** : Você pode configurar o Android Studio para adicionar automaticamente instruções de importação para classes se o significado não for ambíguo. O painel de configurações de **Editor> Geral> Importação automática** especifica como as importações são tratadas.