Introdução

Na aula anterior, como salvar dados no Firebase Realtime Database. Nesta aula, você verá como recuperar os dados salvos.

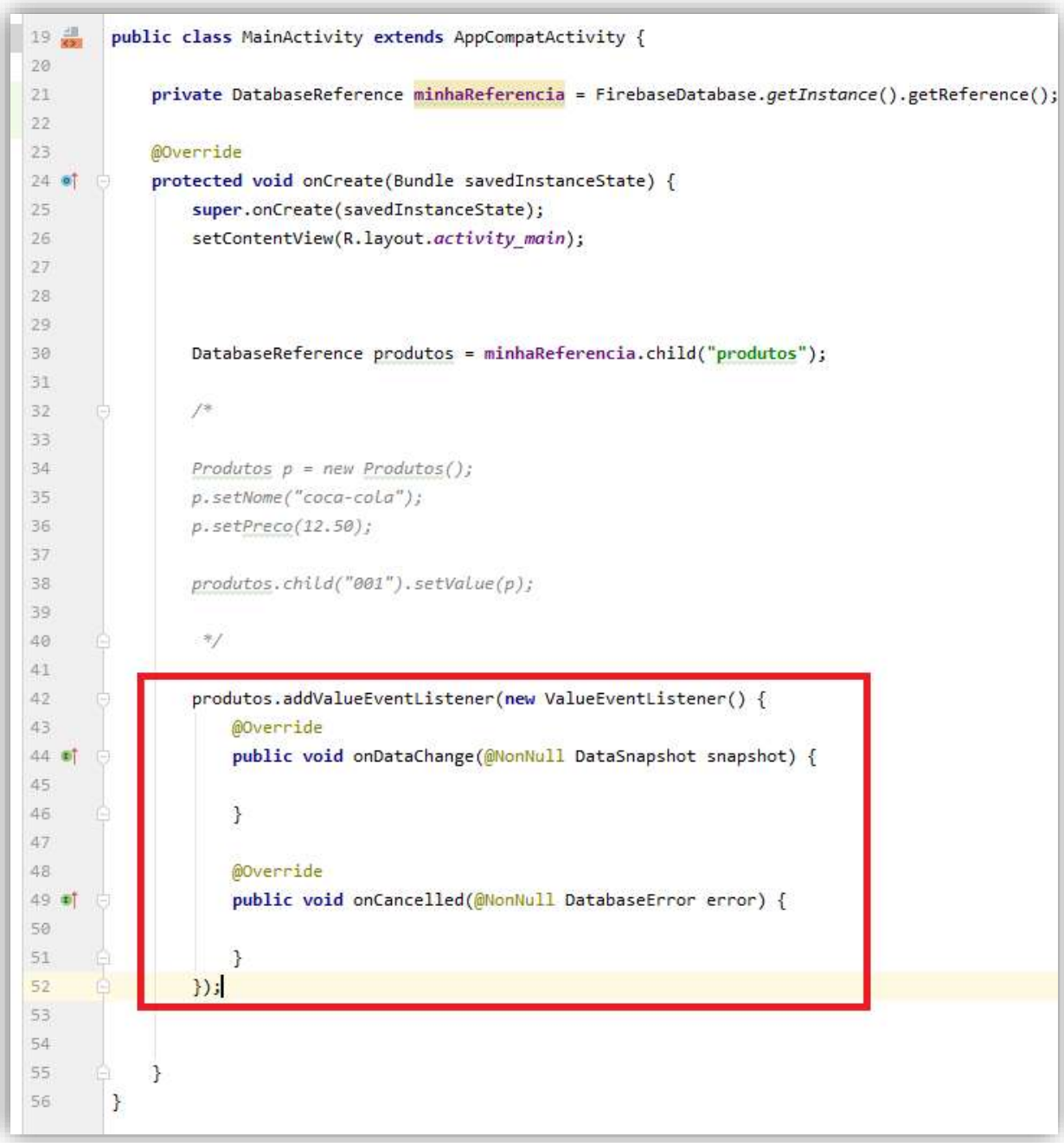
Recuperando Dados

Primeiramente, comente o bloco de código em que salvamos os dados.

```
16  public class MainActivity extends AppCompatActivity {
17
18      private DatabaseReference minhaReferencia = FirebaseDatabase.getInstance().getReference();
19
20      @Override
21      protected void onCreate(Bundle savedInstanceState) {
22          super.onCreate(savedInstanceState);
23          setContentView(R.layout.activity_main);
24
25
26
27          DatabaseReference produtos = minhaReferencia.child("produtos");
28
29          /*
30
31          Produtos p = new Produtos();
32          p.setNome("coca-cola");
33          p.setPreco(12.50);
34
35          produtos.child("001").setValue(p);
36
37           */
38
39
40
41      }
42  }
```

Nós iremos recuperar os produtos salvos utilizando um listener. Um listener de eventos é uma interface na classe View que contém um único método de callback. Esses métodos serão chamados pelo framework do Android quando a View a que o listener estiver registrado for ativada pela interação do usuário com o item na IU. Não irei entrar em detalhes sobre listeners porque não é este o objetivo deste material. Para saber mais sobre isto, acesse a documentação oficial em: https://developer.android.com/guide/topics/ui/ui-events?hl=pt-br

Crie o listener da seguinte maneira:



```
19  public class MainActivity extends AppCompatActivity {
20
21      private DatabaseReference minhaReferencia = FirebaseDatabase.getInstance().getReference();
22
23      @Override
24      protected void onCreate(Bundle savedInstanceState) {
25          super.onCreate(savedInstanceState);
26          setContentView(R.layout.activity_main);
27
28
29
30          DatabaseReference produtos = minhaReferencia.child("produtos");
31
32          /*
33
34          Produtos p = new Produtos();
35          p.setNome("coca-cola");
36          p.setPreco(12.50);
37
38          produtos.child("001").setValue(p);
39
40           */
41
42          produtos.addValueEventListener(new ValueEventListener() {
43              @Override
44              public void onDataChange(@NonNull DataSnapshot snapshot) {
45
46              }
47
48              @Override
49              public void onCancelled(@NonNull DatabaseError error) {
50
51              }
52          });
53
54
55      }
56  }
```

Observe que ao implementar esta classe, são criados automaticamente dois métodos, onDataChange e onCancelled. O primeiro é chamado sempre que você conseguir recuperar os dados com sucesso. Quando algum dado for alterado no firebase, você conseguirá usar este método.

Se por algum motivo a requisição for cancelada, o tratamento desse erro pode ser feito através do método onCancelled.

Observe que o método onDataChange possui um objeto do tipo DataSnapshot. Esse método é o retorno que temos do firebase. Ou seja, ele armazena o resultado da consulta feita.

Ao adicionar este listener em produtos, se houver qualquer mudança dentro do nó produtos, seremos notificados automaticamente. E isto que é um banco de dados em tempo real!

Observe a linha em destaque a seguir:

```
produtos.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {

        Log.i( tag: "FIREBASE", snapshot.getValue().toString());

    }

    @Override
    public void onCancelled(@NonNull DatabaseError error) {

    }
});
```

Nela, eu utilizo um log para recuperar o objeto snapshot com o método getValue. Preciso apenas fazer a conversão usando toString().

Execute o projeto e vá até o ***logcat*** alterando a visualização para ***info*** e você verá que os dados dos produtos serão recuperados através do log. Porém, você poderia recuperar de diversas formas, como em um listView, por exemplo.

Tente alterar um valor diretamente no firebase e você verá que o ***logcat*** será atualizado automaticamente com o novo valor. Isso porque o listener é chamado cada vez que qualquer valor for alterado dentro do nó.