

Desenvolvendo com Android Studio: compraCerta

Giovanni S. Lobo

Programação para Dispositivos Móveis

Sistemas de Informação

Unisul

Estrutura da Apresentação

- **Introdução**
- **Telas**
- **Estrutura de dados**
- **Casos de uso**
- **Diagrama de classes**
- **Diagrama de sequência**
- **Blocos de códigos**
- **Demonstração**
- **Conclusão**

10 minutos ???



compraCerta



20:07

Compra certa ✓

\$ Carteira

Quanto você pode gastar?

limpar

ok

Lista de Compras

Incluir

SUBTOTAL

0

Apagar lista

Fechar

0:01

Produto

Item

7

Salvar

Produto

sabão em pó

Alterar

Quantidade

1

Excluir

Preço unitário

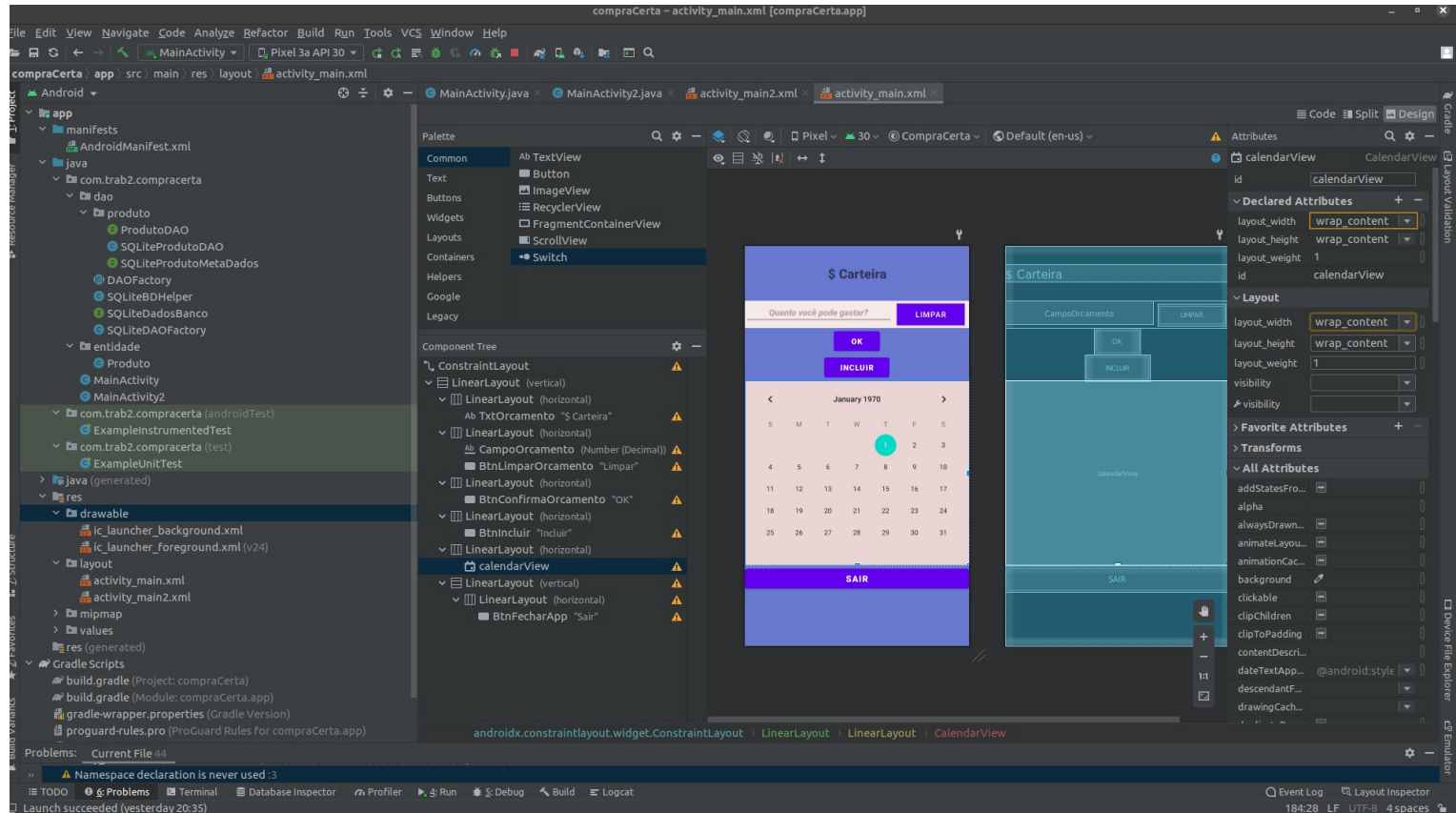
16.48

Limpar

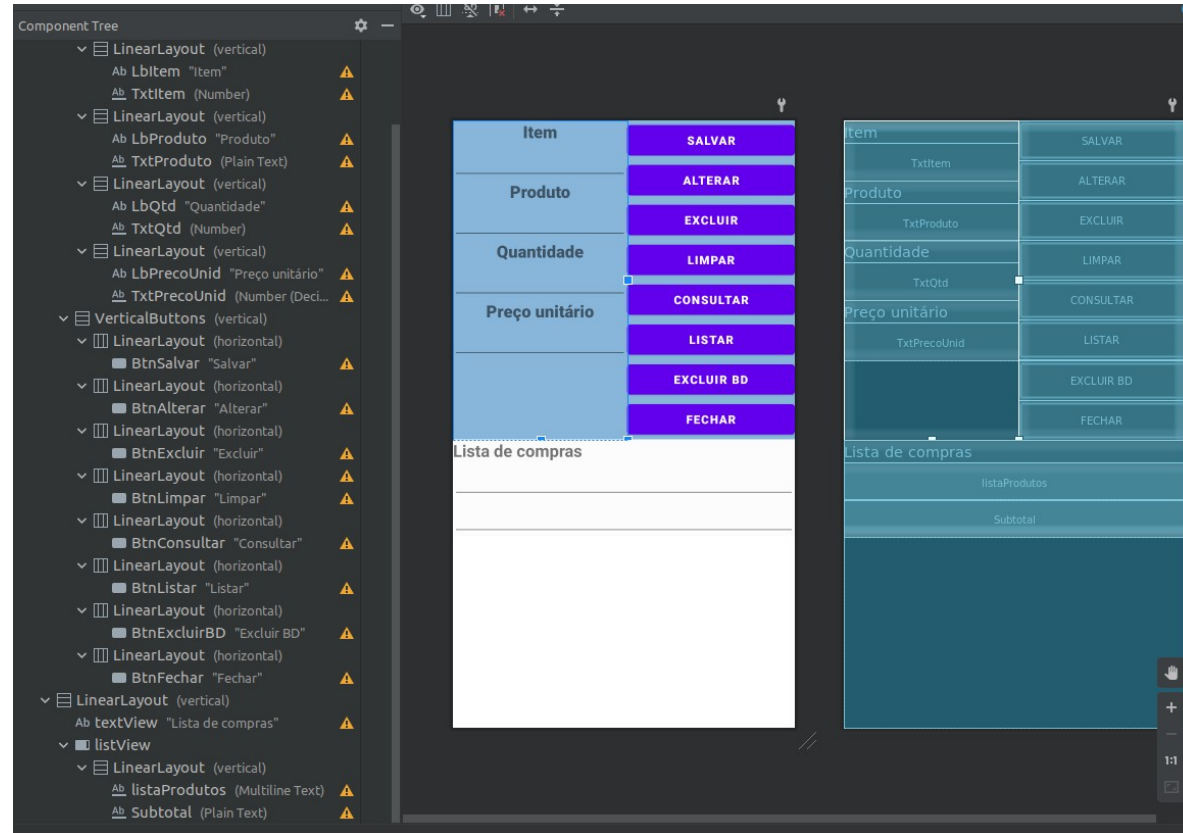
Fechar

Compra certa!
App Inventor

Android Studio: compraCerta

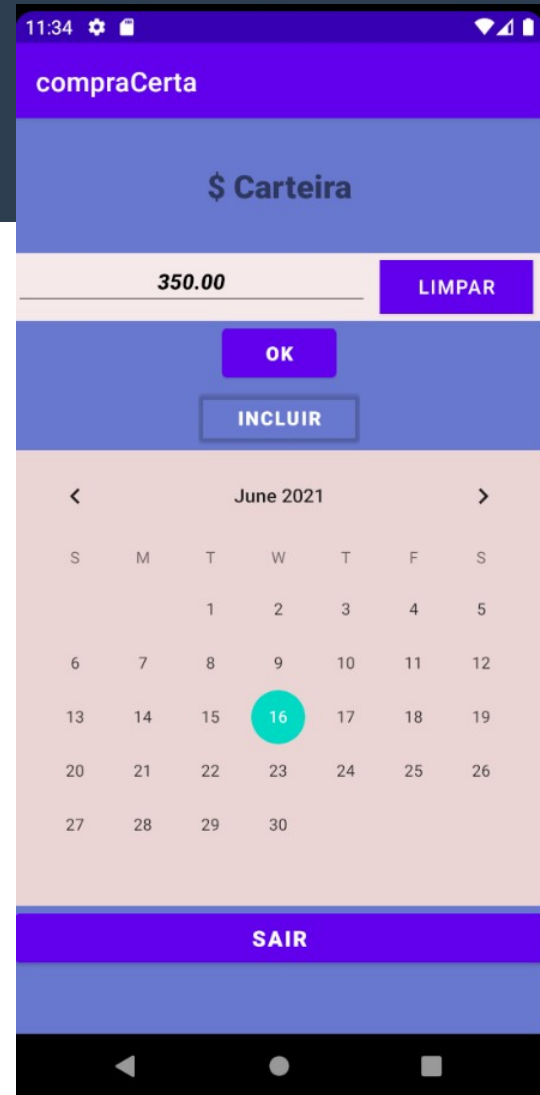
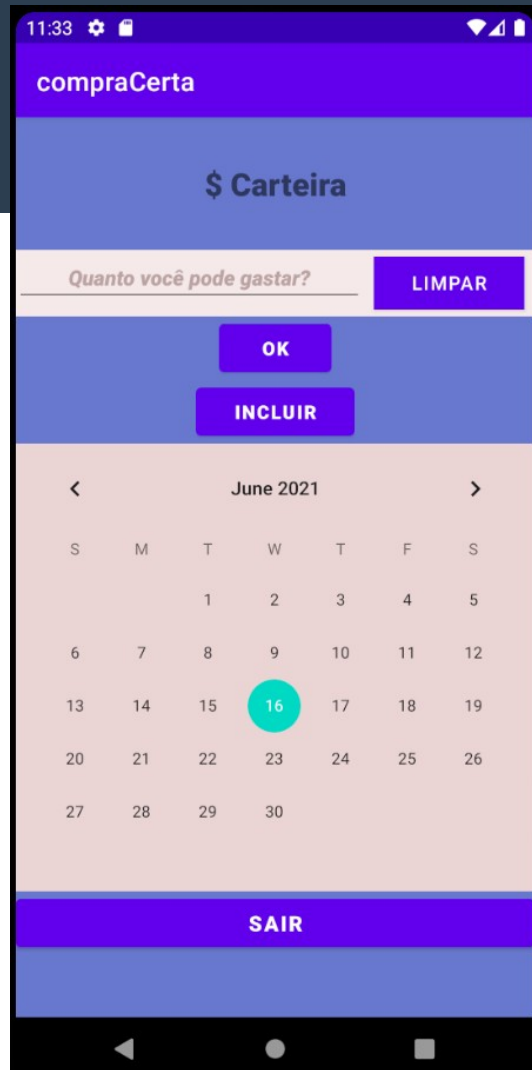


Android Studio: compraCerta



compraCerta

- Tela 1



compraCerta

- Tela 2

11:34

compraCerta

| | |
|----------------|------------|
| Item | SALVAR |
| Produto | ALTERAR |
| Quantidade | EXCLUIR |
| Preço unitário | LIMPAR |
| | CONSULTAR |
| | LISTAR |
| | EXCLUIR BD |
| | FECHAR |

Lista de compras

12:09

compraCerta

| | |
|----------------|------------|
| Item | SALVAR |
| 9 | ALTERAR |
| Produto | EXCLUIR |
| detergente | LIMPAR |
| Quantidade | CONSULTAR |
| 3 | LISTAR |
| Preço unitário | EXCLUIR BD |
| 2.45 | FECHAR |

Lista de compras

1. cafe -> 1 * 11.28 = R\$ 11.28
2. leite -> 9 * 3.49 = R\$ 31.41
3. arroz -> 2 * 4.75 = R\$ 9.5
4. pizza -> 2 * 8.99 = R\$ 17.98
5. carne -> 2 * 29.80 = R\$ 59.6
6. bolacha -> 2 * 3.28 = R\$ 6.56
7. agua -> 4 * 4.99 = R\$ 19.96
8. sabonete -> 5 * 1.99 = R\$ 9.95

SUBTOTAL: R\$ 166.24

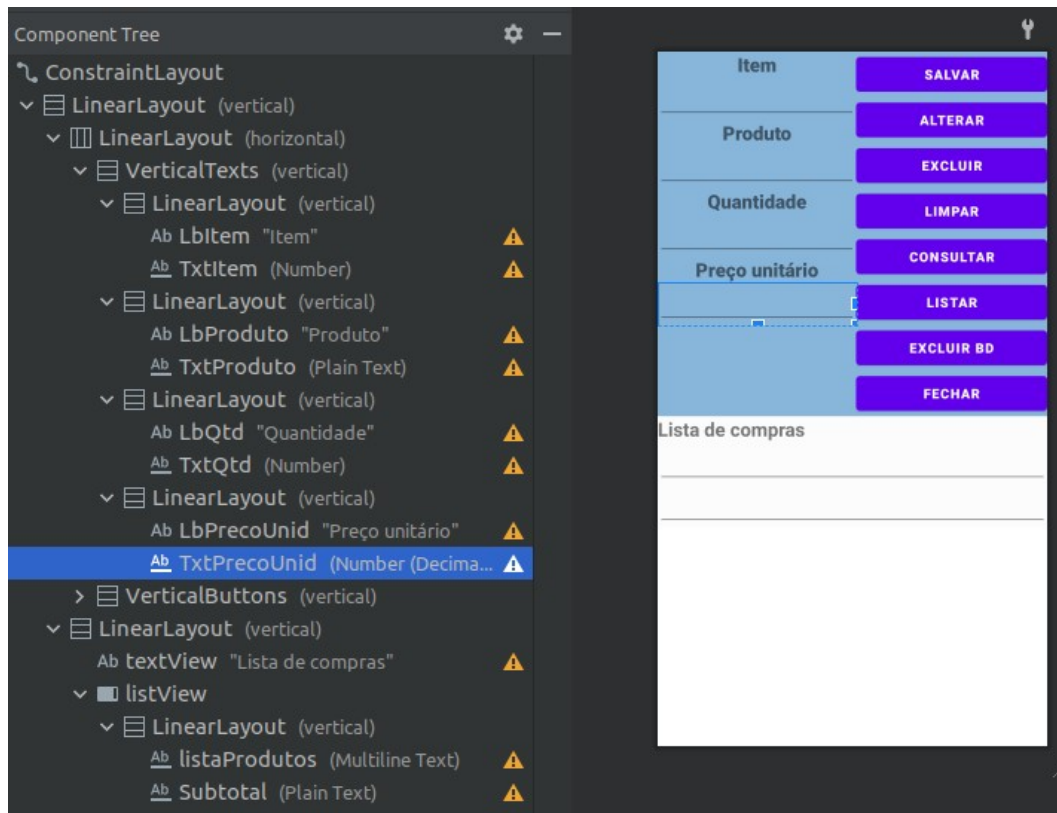
compraCerta

Estrutura de dados

- **TxtOrcamento (decimal, interno)**
- **TxtItem (inteiro)**
- **TxtProduto (text)**
- **TxtQtd (inteiro)**
- **TxtPrecoUnid (decimal)**
- **listaProdutos (multiline text)**
- **Subtotal (text)**

- **Objeto -> Produto**

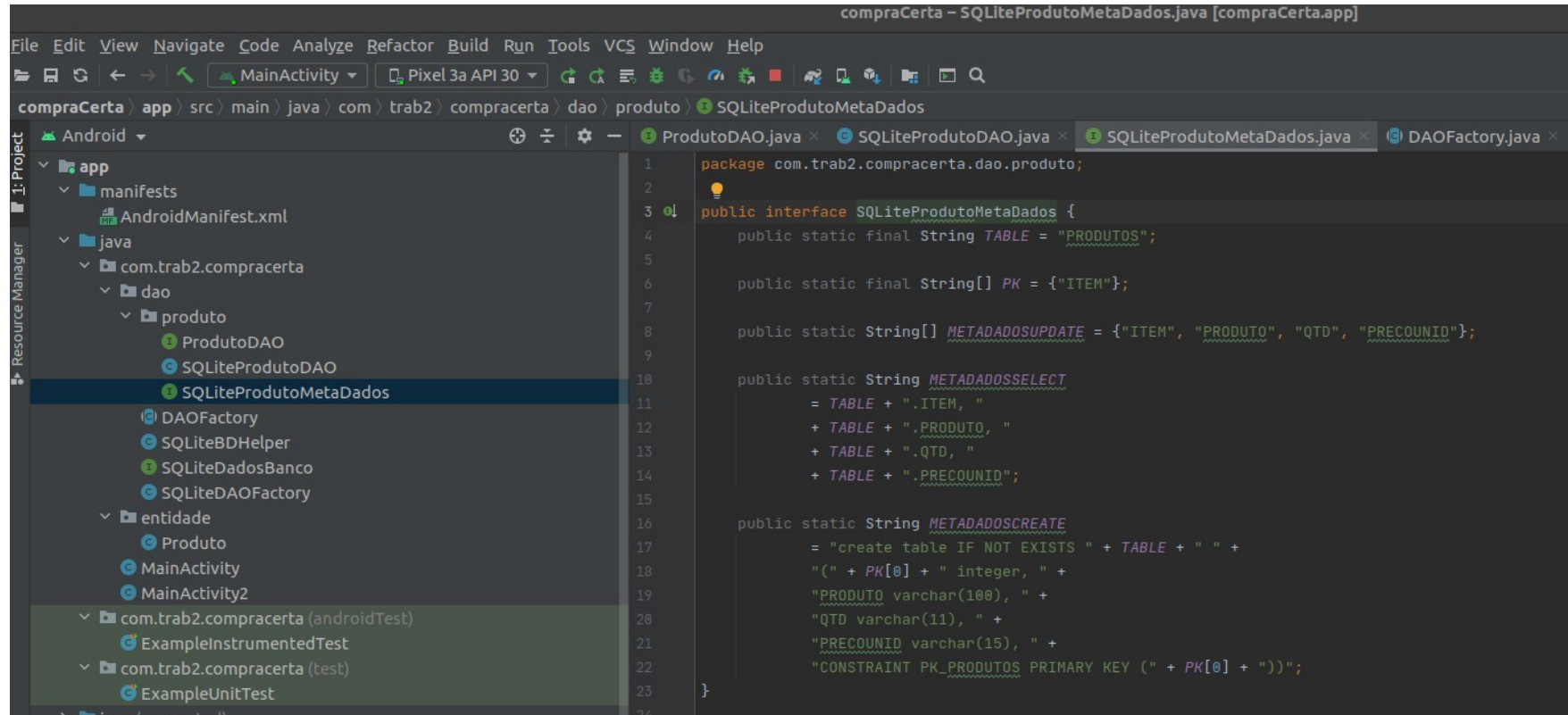
(String item, String produto, String qtd, String precoUnid)



compraCerta

Persistência dos dados

SQLite

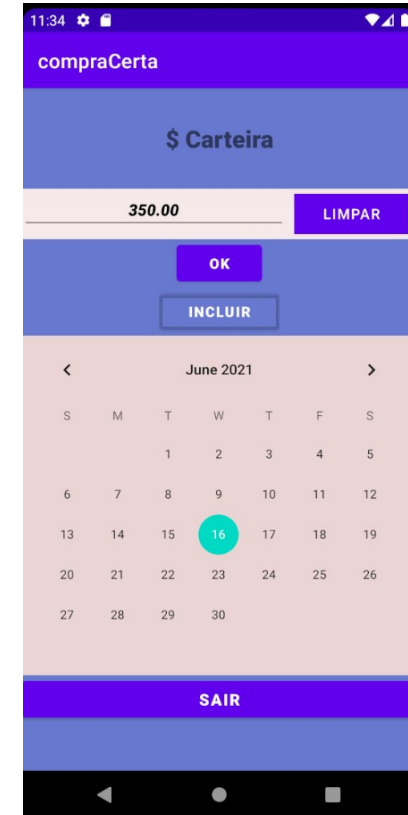


```
compraCerta – SQLiteProdutoMetaDados.java [compraCerta.app]
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MainActivity Pixel 3a API 30
compraCerta > app > src > main > java > com > trab2 > compracerta > dao > produto > SQLiteProdutoMetaDados
Android
  app
    manifests
      AndroidManifest.xml
    java
      com.trab2.compracerta
        dao
          produto
            ProdutoDAO
            SQLiteProdutoDAO
            SQLiteProdutoMetaDados
          DAOFactory
          SQLiteBDHelper
          SQLiteDadosBanco
          SQLiteDAOFactory
        entidade
          Produto
          MainActivity
          MainActivity2
        com.trab2.compracerta (androidTest)
          ExampleInstrumentedTest
        com.trab2.compracerta (test)
          ExampleUnitTest
  resources
    drawable
    mipmap
    values
    xml
  SQLiteProdutoMetaDados.java
  ProdutoDAO.java
  SQLiteProdutoDAO.java
  DAOFactory.java

1 package com.trab2.compracerta.dao.produto;
2
3 public interface SQLiteProdutoMetaDados {
4     public static final String TABLE = "PRODUTOS";
5
6     public static final String[] PK = {"ITEM"};
7
8     public static String[] METADADOSUPDATE = {"ITEM", "PRODUTO", "QTD", "PRECOUNID"};
9
10    public static String METADADOSSELECT
11        = TABLE + ".ITEM, "
12        + TABLE + ".PRODUTO, "
13        + TABLE + ".QTD, "
14        + TABLE + ".PRECOUNID";
15
16    public static String METADADOSCREATE
17        = "create table IF NOT EXISTS " + TABLE + " " +
18        "(" + PK[0] + " integer, " +
19        "PRODUTO varchar(100), " +
20        "QTD varchar(11), " +
21        "PRECOUNID varchar(15), " +
22        "CONSTRAINT PK_PRODUTOS PRIMARY KEY (" + PK[0] + "))";
23
24 }
```

Caso de Uso

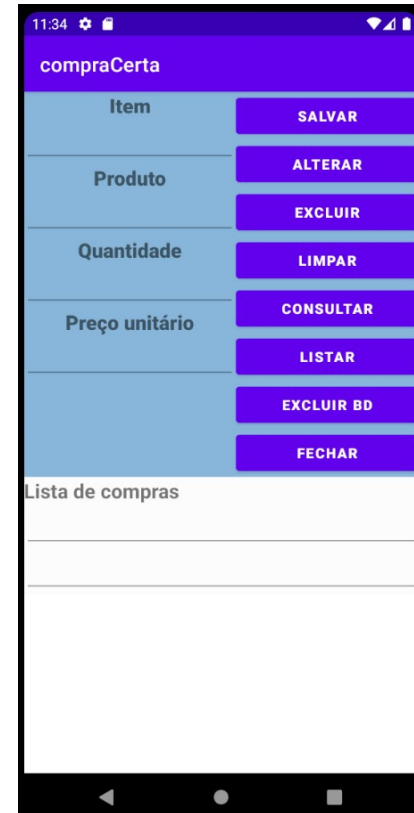
| | |
|-------------------------------|---|
| UC número | UC001 - Inserir o valor do orçamento |
| Objetivo | O usuário deve ser capaz de inserir o valor do orçamento disponível para compra no campo indicado |
| Pré-Condições | Possuir o software instalado |
| Ator | Usuário |
| Gatilho | Clicar na caixa de texto <i>"Quanto você pode gastar?"</i> |
| Procedimento das ações | <ol style="list-style-type: none">1. Clicar no campo para entrada do orçamento2. Digitar o valor que pode gastar3. Confirmar a ação clicando no botão "OK" |
| Fluxo de exceção 1 | <p>Se no passo 2 o usuário digitar um valor errado:</p> <ol style="list-style-type: none">3. Limpar o valor errado clicando no botão "Limpar" ao lado4. Digitar o valor correto que pode gastar5. Confirmar a ação clicando no botão "OK" |



compraCerta

| | |
|-------------------------------|--|
| UC número | UC002 - Incluir um produto na lista de compras |
| Objetivo | O usuário deve ser capaz de inicializar a inclusão de produtos na lista de compras e adicionar produtos adicionais |
| Pré-Condições | Estar com a tela 1 do aplicativo aberta e ser direcionado para a tela 2 do aplicativo |
| Ator | Usuário |
| Gatilho | Selecionar a opção de incluir produtos |
| Procedimento das ações | <ol style="list-style-type: none">1. Abrir o software2. Clicar no botão de "INCLUIR" na tela 13. Inserir o número do item na caixa de texto do "Item"4. Inserir o nome do produto na caixa de texto do "Produto"5. Inserir a quantidade de itens desejados na caixa de texto "Quantidade"6. Inserir o preço do produto na caixa de texto do "Preço unitário"7. Clicar no botão "SALVAR"8. O sistema deverá apresentar a mensagem "Item incluído com sucesso!" |
| Fluxo alternativo 1 | <p>Se o usuário já estiver na tela 2 e desejar incluir um outro produto na lista de compras, então:</p> <ol style="list-style-type: none">1. Retorna ao passo 3 do fluxo de base |
| Fluxo de exceção 1 | <p>Se em qualquer um dos passos 3, 4, 5 ou 6 do fluxo base o usuário digitar um valor errado, então:</p> <ol style="list-style-type: none">1. O usuário poderá limpar todos os valores de todos os campos clicando no botão "LIMPAR"2. Retorna ao passo 3 do fluxo base |

Caso de Uso



compraCerta

| | |
|-------------------------------|--|
| UC número | UC003 - Consultar um produto na lista de compras |
| Objetivo | O usuário deve ser capaz de consultar todos os valores referentes a qualquer produto presente na lista de produtos |
| Pré-Condições | Estar com a tela 2 do aplicativo aberta e possuir algum produto cadastrado no banco de dados PRODUTOS |
| Ator | Usuário |
| Gatilho | Selecionar a opção de consultar produtos |
| Procedimento das ações | <ol style="list-style-type: none">1. Abrir a tela 2 do aplicativo2. Clicar no botão "LISTAR" para listar todos os produtos cadastrados no banco de dados3. Identificar o valor do "Item" referente ao produto que deseja pesquisar4. Inserir o valor do "Item" na caixa de texto "Item"5. Clicar no botão "CONSULTAR"6. O sistema deverá resgatar e apresentar nas respectivas caixas de texto todos os valores cadastrados para o produto7. O sistema deverá apresentar a mensagem "Item encontrado!" |
| Fluxo alternativo 1 | <p>Após o passo 1 do fluxo base o sistema deverá apresentar a lista de produtos cadastrados previamente, então:</p> <ol style="list-style-type: none">1. O usuário poderá passar para o passo 3 do fluxo base |
| Fluxo de exceção 1 | <p>Se no passo 4 do fluxo base o usuário digitar um valor não presente na lista de produtos, então após executar o passo 5 do fluxo base:</p> <ol style="list-style-type: none">1. O sistema não deverá alterar os valores das caixas de texto2. O sistema deverá apresentar a mensagem "Item não encontrado!" |

Caso de Uso

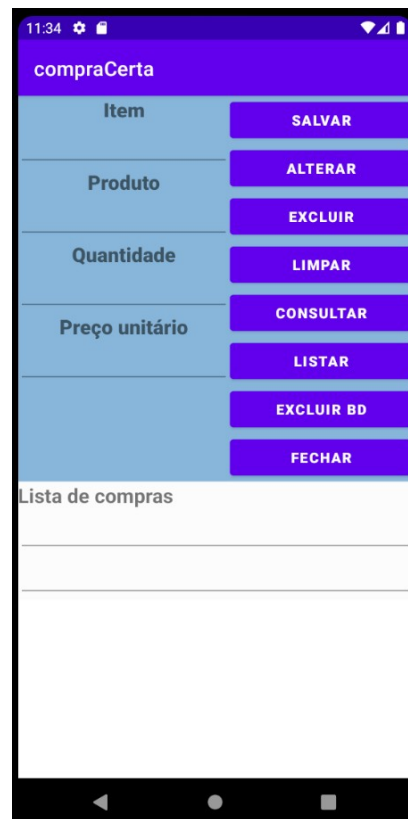
The screenshot shows the 'compraCerta' mobile application. The top section has a purple header with the app name. Below it, there are input fields and buttons for searching a product. The 'Item' field contains '9', 'Produto' contains 'detergente', 'Quantidade' contains '3', and 'Preço unitário' contains '2.45'. To the right of these fields are buttons: SALVAR, ALTERAR, EXCLUIR, LIMPAR, CONSULTAR, LISTAR, EXCLUIR BD, and FECHAR. Below this is a section titled 'Lista de compras' which displays a list of items with their quantities and prices. At the bottom, there is a 'SUBTOTAL: R\$ 166.24'.

| Item | Quantidade | Preço unitário | Total |
|----------|------------|----------------|-----------|
| cafe | 1 | 11.28 | R\$ 11.28 |
| leite | 9 | 3.49 | R\$ 31.41 |
| arroz | 2 | 4.75 | R\$ 9.5 |
| pizza | 2 | 8.99 | R\$ 17.98 |
| carne | 2 | 29.80 | R\$ 59.6 |
| bolacha | 2 | 3.28 | R\$ 6.56 |
| agua | 4 | 4.99 | R\$ 19.96 |
| sabonete | 5 | 1.99 | R\$ 9.95 |

SUBTOTAL: R\$ 166.24

| | |
|-------------------------------|---|
| UC número | UC004 - Excluir um produto da lista de compras |
| Objetivo | O usuário deve ser capaz de excluir todos os valores referentes a qualquer produto presente na lista de produtos |
| Pré-Condições | Estar com a tela 2 do aplicativo aberta e possuir algum produto cadastrado no banco de dados PRODUTOS |
| Ator | Usuário |
| Gatilho | Selecionar a opção de excluir produto |
| Procedimento das ações | <ol style="list-style-type: none">1. Abrir a tela 2 do aplicativo2. Clicar no botão "LISTAR" para listar todos os produtos cadastrados no banco de dados3. Identificar o valor do "Item" referente ao produto que deseja excluir4. Inserir o valor do "Item" na caixa de texto "Item"5. Clicar no botão "EXCLUIR"6. O sistema deverá apresentar a mensagem: "Deseja excluir o produto?"7. Clicar no botão "SIM"8. O sistema deverá excluir do banco de dados todos os valores referentes ao produto9. O sistema deverá apresentar a mensagem "Exclusão realizada com sucesso!"10. O sistema deverá listar todos os produtos cadastrados sem os valores do produto excluído |
| Fluxo alternativo 1 | <p>Após o passo 1 do fluxo base o sistema deverá apresentar a lista de produtos cadastrados previamente, então:</p> <ol style="list-style-type: none">1. O usuário poderá passar para o passo 3 do fluxo base |
| Fluxo alternativo 2 | <p>Se após o passo 1 do fluxo base o usuário não realizar o passo 4 e executar o passo 5 do fluxo base, então:</p> <ol style="list-style-type: none">1. O sistema deverá apresentar a mensagem "Digite um item!" |
| Fluxo alternativo 3 | <p>Se no passo 7 do fluxo de base o usuário clicar no botão "NÃO", então:</p> <ol style="list-style-type: none">1. O sistema não excluirá o produto da lista de compras |
| Fluxo de exceção 1 | <p>Se no passo 4 do fluxo base o usuário digitar um valor não presente na lista de produtos, então após executar o passo 5 do fluxo base:</p> <ol style="list-style-type: none">1. O sistema deverá apresentar a mensagem "Item não encontrado!" |

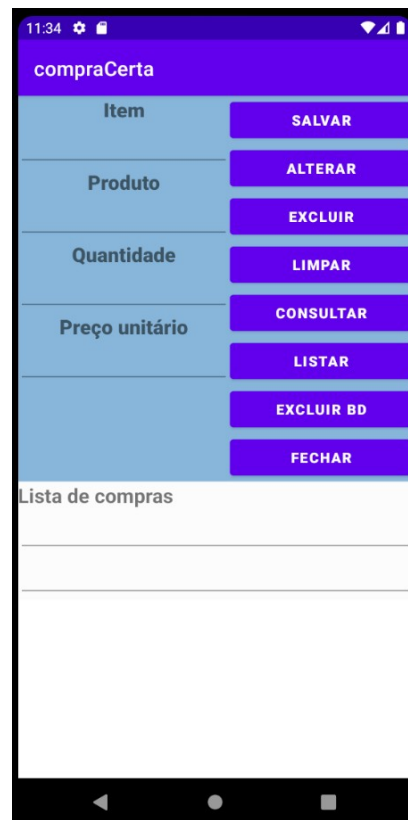
Caso de Uso



compraCerta

| | |
|-------------------------------|--|
| UC número | UC005 - Alterar um produto na lista de compras |
| Objetivo | O usuário deve ser capaz de alterar qualquer valor referente a qualquer produto presente na lista de compras |
| Pré-Condições | Estar com a tela 2 do aplicativo aberta e possuir algum produto cadastrado no banco de dados PRODUTOS |
| Ator | Usuário |
| Gatilho | Selecionar a opção de alterar produto |
| Procedimento das ações | <ol style="list-style-type: none">1. Abrir a tela 2 do aplicativo2. Clicar no botão "LISTAR" para listar todos os produtos cadastrados no banco de dados3. Identificar o valor do "Item" referente ao produto que deseja alterar4. Inserir o valor do "Item" na caixa de texto "Item"5. Clicar no botão "CONSULTAR"6. O sistema deverá resgatar e apresentar nas respectivas caixas de texto todos os valores cadastrados para o produto7. Ajustar o valor na caixa de texto que deseja alterar8. Clicar no botão "ALTERAR"7. O sistema deverá apresentar a lista de compras com o referido produto apresentando os seus valores alterados |
| Fluxo alternativo 1 | <p>Após o passo 1 do fluxo base o sistema deverá apresentar a lista de produtos cadastrados previamente, então:</p> <ol style="list-style-type: none">1. O usuário poderá passar para o passo 3 do fluxo base |
| Fluxo de exceção 1 | <p>Se após o passo 1 do fluxo de base o usuário passar direto para o passo 8 do fluxo de base, então:</p> <ol style="list-style-type: none">2. O sistema deverá apresentar a mensagem "Digite um item!" |

Caso de Uso



Caso de Uso

| | |
|-------------------------------|--|
| UC número | UC006 - Excluir todos os produtos da lista de compras |
| Objetivo | O usuário deve ser capaz de excluir a lista de compras |
| Pré-Condições | Estar com a tela 2 do aplicativo aberta e possuir algum produto cadastrado no banco de dados PRODUTOS |
| Ator | Usuário |
| Gatilho | Selecionar a opção de excluir banco de dados |
| Procedimento das ações | <ol style="list-style-type: none">1. Abrir a tela 2 do aplicativo2. Clicar no botão "EXCLUIR BD"3. O sistema deverá apresentar a mensagem: "Deseja apagar a lista de compras?"4. Clicar no botão "SIM"5. O sistema deverá excluir todos os produtos da lista de compras7. O sistema deverá apresentar a mensagem "Lista apagada!" |
| Fluxo alternativo 1 | <p>Se no passo 4 do fluxo de base o usuário clicar no botão "NÃO", então:</p> <ol style="list-style-type: none">1. O sistema não deverá alterar a lista de compras |

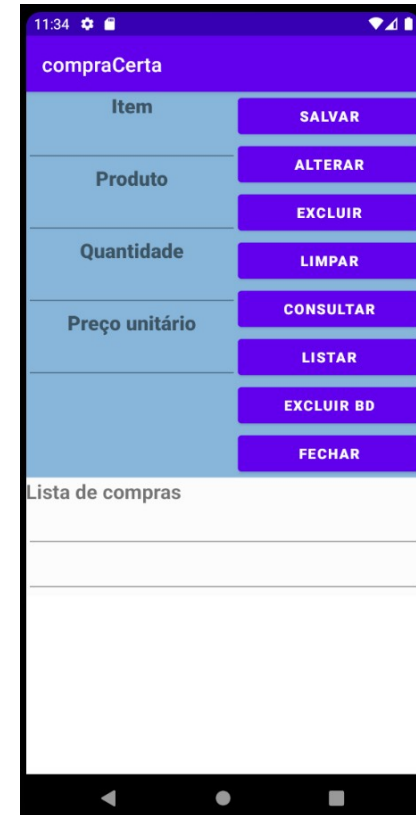


Diagrama de Classes

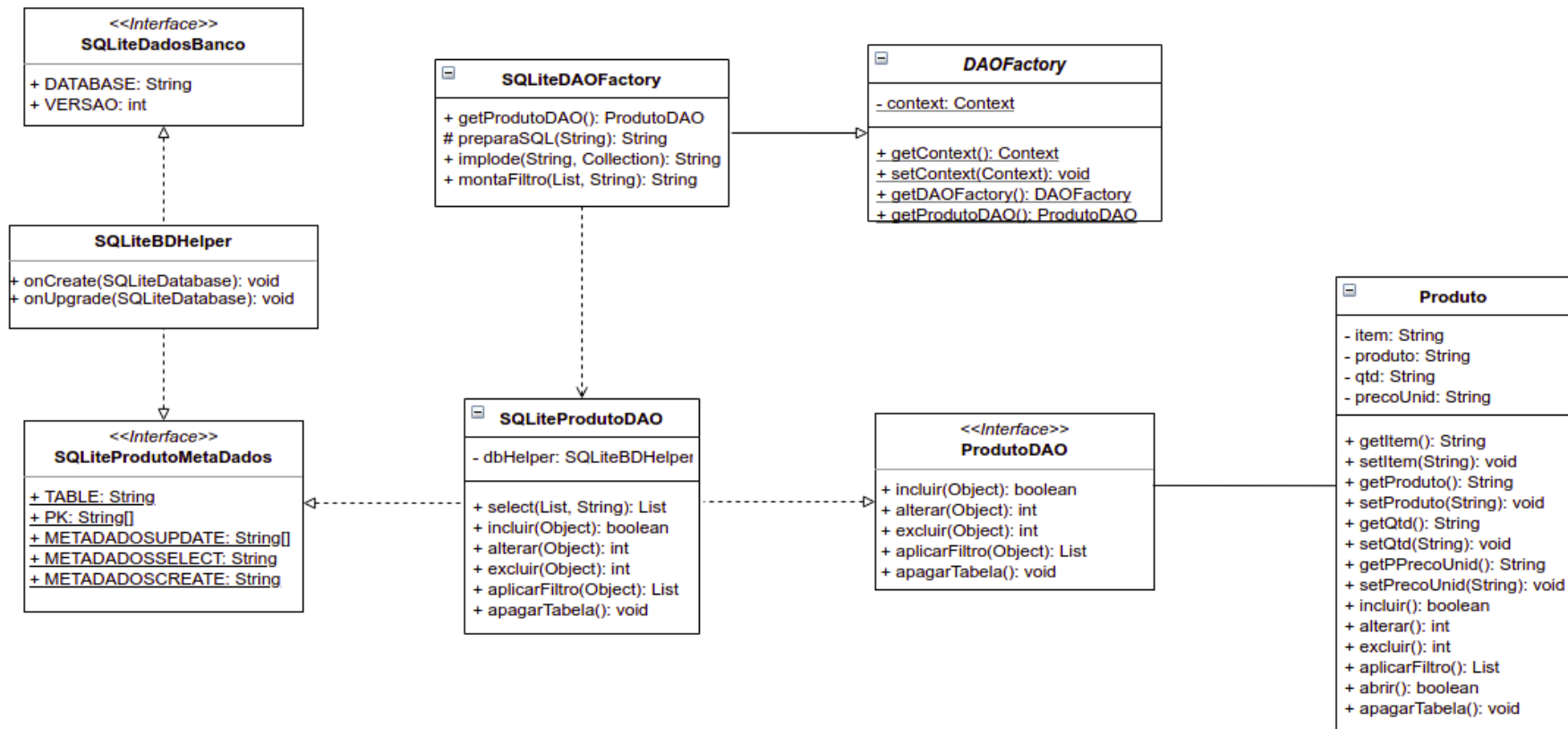
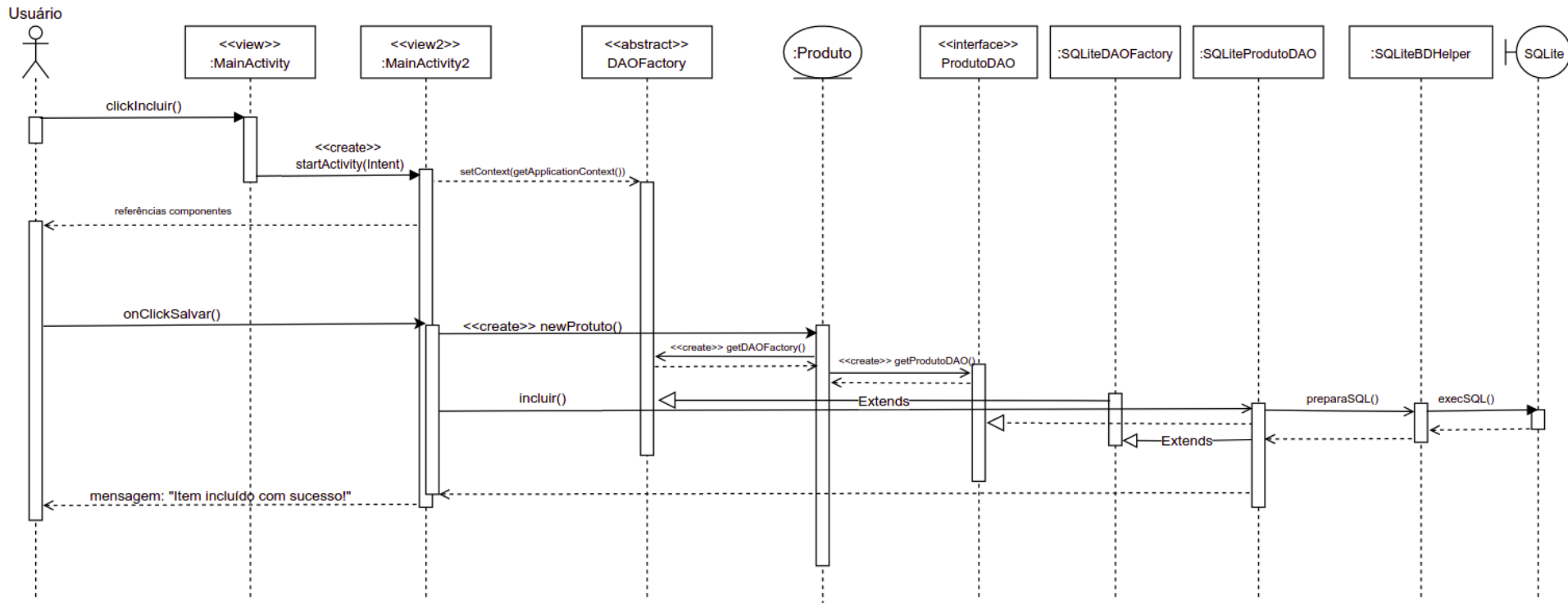


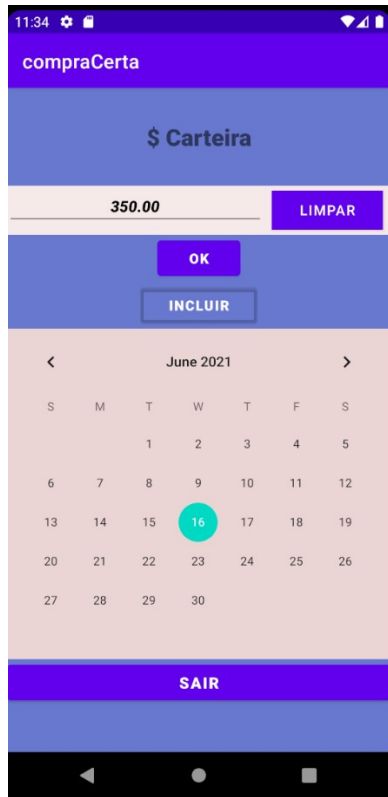
Diagrama de Sequência

UC002



compraCerta

Tela 1: Código



```
MainActivity.java x activity_main2.xml x
1 package com.trab2.compracerta;
2
3 import ...
17
18 public class MainActivity extends AppCompatActivity {
19
20     private TextView txtOrcamento;
21     private EditText campoOrcamento;
22     private Button btnLimparOrcamento;
23     private Button btnConfirmaOrcamento;
24     private CalendarView calendario;
25     private Button btnIncluir;
26     private Button btnFecharApp;
27
28     @Override
29     protected void onCreate(Bundle savedInstanceState) {
30         super.onCreate(savedInstanceState);
31         setContentView(R.layout.activity_main);
32
33         txtOrcamento = findViewById(R.id.TxtOrcamento);
34         campoOrcamento = findViewById(R.id.CampoOrcamento);
35         btnLimparOrcamento = findViewById(R.id.BtnLimparOrcamento);
36         btnConfirmaOrcamento = findViewById(R.id.BtnConfirmaOrcamento);
37         calendario = findViewById(R.id.calendarView);
38         btnIncluir = findViewById(R.id.BtnIncluir);
39         btnFecharApp = findViewById(R.id.BtnFecharApp);
40     }
41
42     public void clickBtnLimparOrcamento (View v) { campoOrcamento.setText(""); }
43
44     public void clickConfirmaOrcamento (View v){
45         campoOrcamento.clearFocus();
46         btnIncluir.setBackgroundColor(259);
47         hideSoftKeyboard(v);
48     }
49
50     public void clickIncluir (View v){
51         Intent intent = new Intent( packageContext: this, MainActivity2.class);
52         startActivity(intent);
53     }
54
55 }
```

compraCerta

Tela 2: Código

12:09

compraCerta

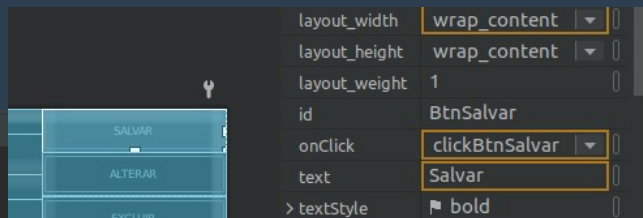
| Item | |
|----------------|------------|
| 9 | SALVAR |
| Produto | ALTERAR |
| detergente | EXCLUIR |
| Quantidade | LIMPAR |
| 3 | CONSULTAR |
| Preço unitário | LISTAR |
| 2.45 | EXCLUIR BD |
| | FECHAR |

Lista de compras

- 1. cafe -> 1 * 11.28 = R\$ 11.28
- 2. leite -> 9 * 3.49 = R\$ 31.41
- 3. arroz -> 2 * 4.75 = R\$ 9.5
- 4. pizza -> 2 * 8.99 = R\$ 17.98
- 5. carne -> 2 * 29.80 = R\$ 59.6
- 6. bolacha -> 2 * 3.28 = R\$ 6.56
- 7. agua -> 4 * 4.99 = R\$ 19.96
- 8. sabonete -> 5 * 1.99 = R\$ 9.95

SUBTOTAL: R\$ 166.24

```
MainActivity2.java x activity_main2.xml x
71 }
72
73 public void clickBtnSalvar(View v) {
74     if (item.getText().toString().equals("")) {
75         Produto prod = new Produto();
76         prod.setItem(item.getText().toString());
77         prod.setProduto(produto.getText().toString());
78         prod.setQtd(qtd.getText().toString());
79         prod.setPrecoUnid(precoUnid.getText().toString());
80         boolean resultado = prod.incluir();
81         setResult(RESULT_CANCELED);
82         if (resultado == true) {
83             Toast.makeText(context: MainActivity2.this, text: "Item incluído com sucesso!", Toast.LENGTH_SHORT).show();
84             hideSoftKeyboard(v);
85             item.setText("");
86             produto.setText("");
87             qtd.setText("");
88             precoUnid.setText("");
89         } else {
90             Toast.makeText(context: MainActivity2.this, text: "Inclusão não realizada!", Toast.LENGTH_SHORT).show();
91         }
92     } else {
93         item.requestFocus();
94     }
95     clickListar(v);
96 }
97
98 public void clickBtnAlterar(View v) {
99     Produto prod = new Produto();
100     prod.setItem(item.getText().toString());
101     prod.setProduto(produto.getText().toString());
102     prod.setQtd(qtd.getText().toString());
103     prod.setPrecoUnid(precoUnid.getText().toString());
104     prod.alterar();
105     clickListar(v);
106 }
```



compraCerta

Código

```
Produto.java x
1 package com.trab2.compracerta.entidade;
2
3 import ...
4
5
6
7
8 public class Produto {
9
10     private String item;
11
12     private String produto;
13
14     private String qtd;
15
16     private String precoUnid;
17
18     public Produto() { this( item: "", produto: "", qtd: "", precoUnid: ""); }
19
20
21
22     public Produto(String item, String produto, String qtd, String precoUnid) {
23         setItem(item);
24         setProduto(produto);
25         setQtd(qtd);
26         setPrecoUnid(precoUnid);
27     }
28
29     public String getItem() { return item; }
30
31
32     public void setItem(String item) { this.item = item; }
33
34
35
36
37     public String getProduto() { return produto; }
38
39
40     public void setProduto(String produto) { this.produto = produto; }
41
42
43
44
45     public String getQtd() { return qtd; }
46
47
48     public void setQtd(String qtd) { this.qtd = qtd; }
49
50
51
52
53     public String getPrecoUnid() { return precoUnid; }
54
55
56     public void setPrecoUnid(String precoUnid) { this.precoUnid = precoUnid; }
57
58
59
60
61
62
63     public boolean incluir() {
64         DAOFactory factory = DAOFactory.getDAOFactory();
65         ...
66     }
67 }
```

compraCerta

Código

```
SQLiteBDHelper.java x
1  package com.trab2.compracerta.dao;
2
3  import ...
4
5  public class SQLiteBDHelper extends SQLiteOpenHelper implements SQLiteDadosBanco, SQLiteProdutoMetaDados {
6
7      public SQLiteBDHelper(Context context) {
8          super(context, SQLiteDadosBanco.DATABASE, factory: null, SQLiteDadosBanco.VERSAO);
9      }
10
11      @Override
12      public void onCreate(SQLiteDatabase db) { db.execSQL(SQLiteProdutoMetaDados.METADADOSCREATE); }
13
14      @Override
15      public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
16
17      }
18  }
```

compraCerta

Código

```
package com.trab2.compracerta.dao.produto;

import ...

public class SQLiteProdutoDAO extends SQLiteDAOFactory implements ProdutoDAO, SQLiteProdutoMetaDados {

    private SQLiteBDHelper dbHelper;

    public SQLiteProdutoDAO() { dbHelper = new SQLiteBDHelper(getContext()); }

    private List select(List<String> filtros, String ordem) {
        List lista = new LinkedList();
        String[] columnas = METADADOSSELECT.split( regex: " ");
        SQLiteDatabase db = null;
        Cursor cursor = null;
        try {
            db = dbHelper.getReadableDatabase();
            cursor = db.query(TABLE, columnas, montaFiltro(filtros, separador: " and ", selectionArgs: null,
            while (cursor.moveToNext()) {
                Produto prod = new Produto();
                prod.setItem(cursor.getString(cursor.getColumnIndex( columnName: "PRODUTO")));
                prod.setProduto(cursor.getString(cursor.getColumnIndex( columnName: "PRODUTO")));
                prod.setQtd(cursor.getString(cursor.getColumnIndex( columnName: "QTD")));
                prod.setPrecoUnid(cursor.getString(cursor.getColumnIndex( columnName: "PRECOUNID")));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return lista;
    }
}
```

```
DAOFactory.java
1 package com.trab2.compracerta.dao;
2
3 import ...
4
5
6
7 public abstract class DAOFactory {
8     private static Context context;
9
10    public static Context getContext() { return context; }
11
12
13
14    public static void setContext(Context context) { DAOFactory.context = context; }
15
16
17
18    public static DAOFactory getDAOFactory() { return new SQLiteDAOFactory(); }
19
20
21
22    public abstract ProdutoDAO getProdutoDAO();
23 }
```

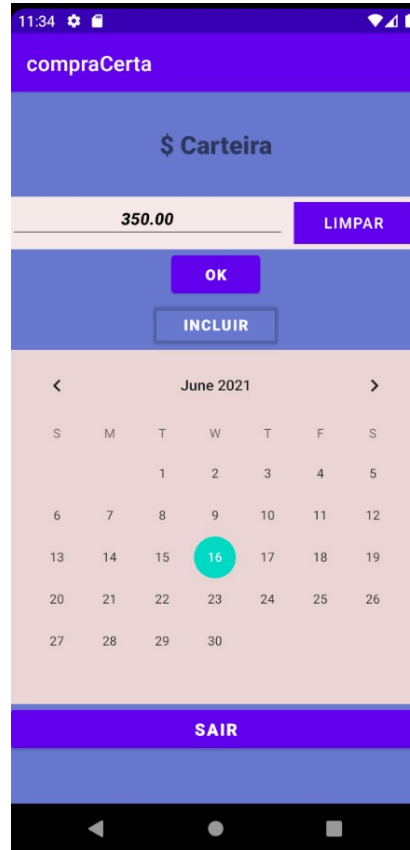
```
SQLiteDAOFactory.java MainActivity2.java activity_main2.xml
1 package com.trab2.compracerta.dao;
2
3 import ...
4
5
6
7
8
9
10 public class SQLiteDAOFactory extends DAOFactory implements SQLiteDadosBanco {
11     public ProdutoDAO getProdutoDAO() { return new SQLiteProdutoDAO(); }
12
13
14     protected String preparaSQL(String valor) {
15         if (valor != null) {
16             return valor.replaceAll( regex: " ", replacement: "");
17         } else {
18             return "";
19         }
20     }
21
22
23
24    public String implode(String separator, Collection collection) {
25        StringBuffer textBufferReturn = new StringBuffer();
26        //rawtypes/
27        Iterator it = collection.iterator();
28        while (it.hasNext()) {
29            String text = (String) it.next();
30            textBufferReturn.append(text);
31            if (it.hasNext()) {
32                textBufferReturn.append(separator);
33            }
34        }
35        return textBufferReturn.toString();
36    }
37
38    public String montaFiltro(List<String> lista, String separador) {
39        StringBuffer filtro = new StringBuffer();
40        Iterator<String> filtroIt = lista.iterator();
41        while (filtroIt.hasNext()) {
42            String texto = filtroIt.next();
43            filtro.append(texto);
44            if (filtroIt.hasNext()) {
45                filtro.append(" " + separador + " ");
46            }
47        }
48        return filtro.toString();
49    }
50 }
```


Demonstração

compraCerta

compraCerta

- Conclusão
- Android Studio
- compraCerta
- Versões futuras





FIM