

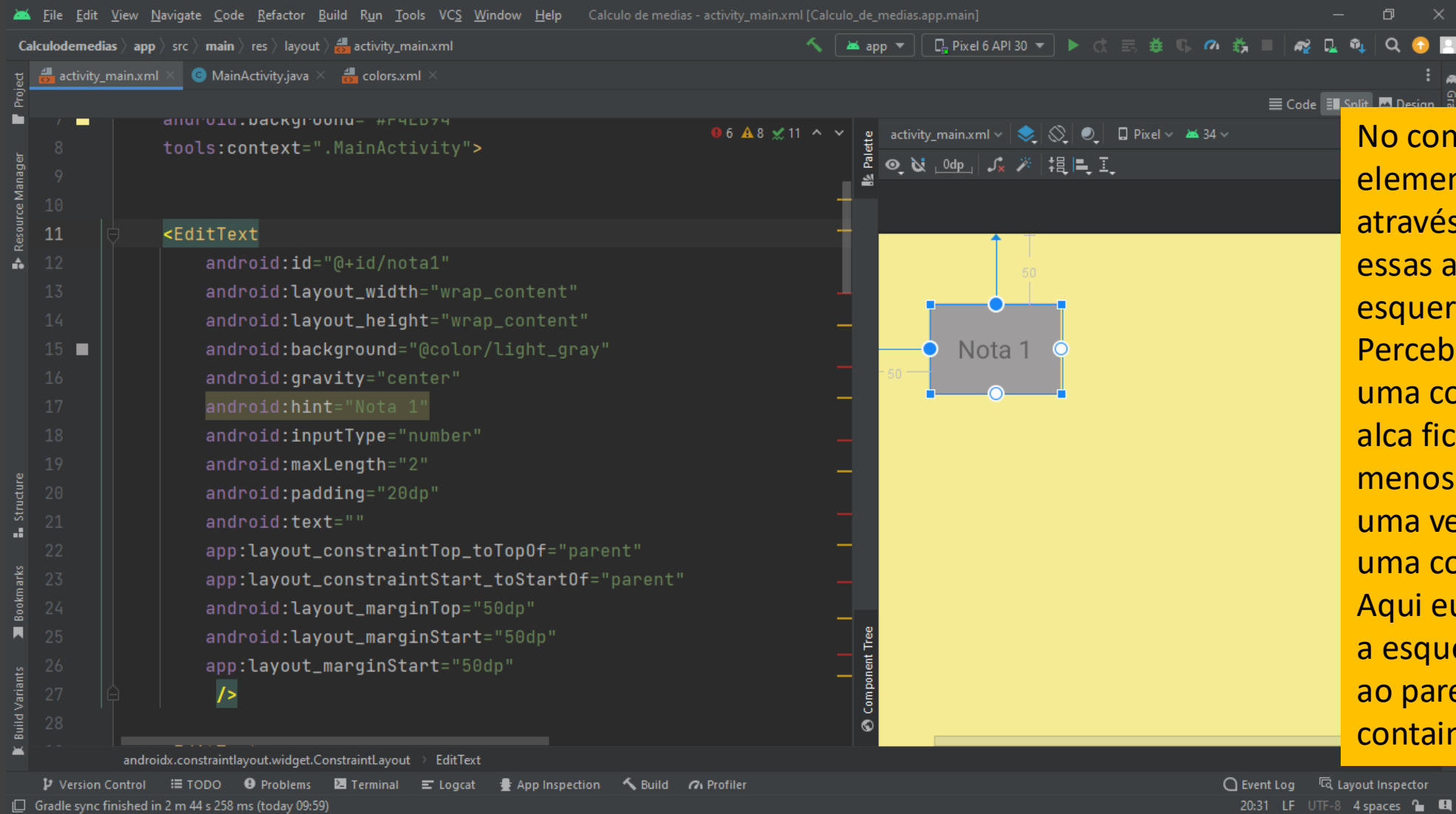
Aplicativo Media

Criar aplicativo calcular media com

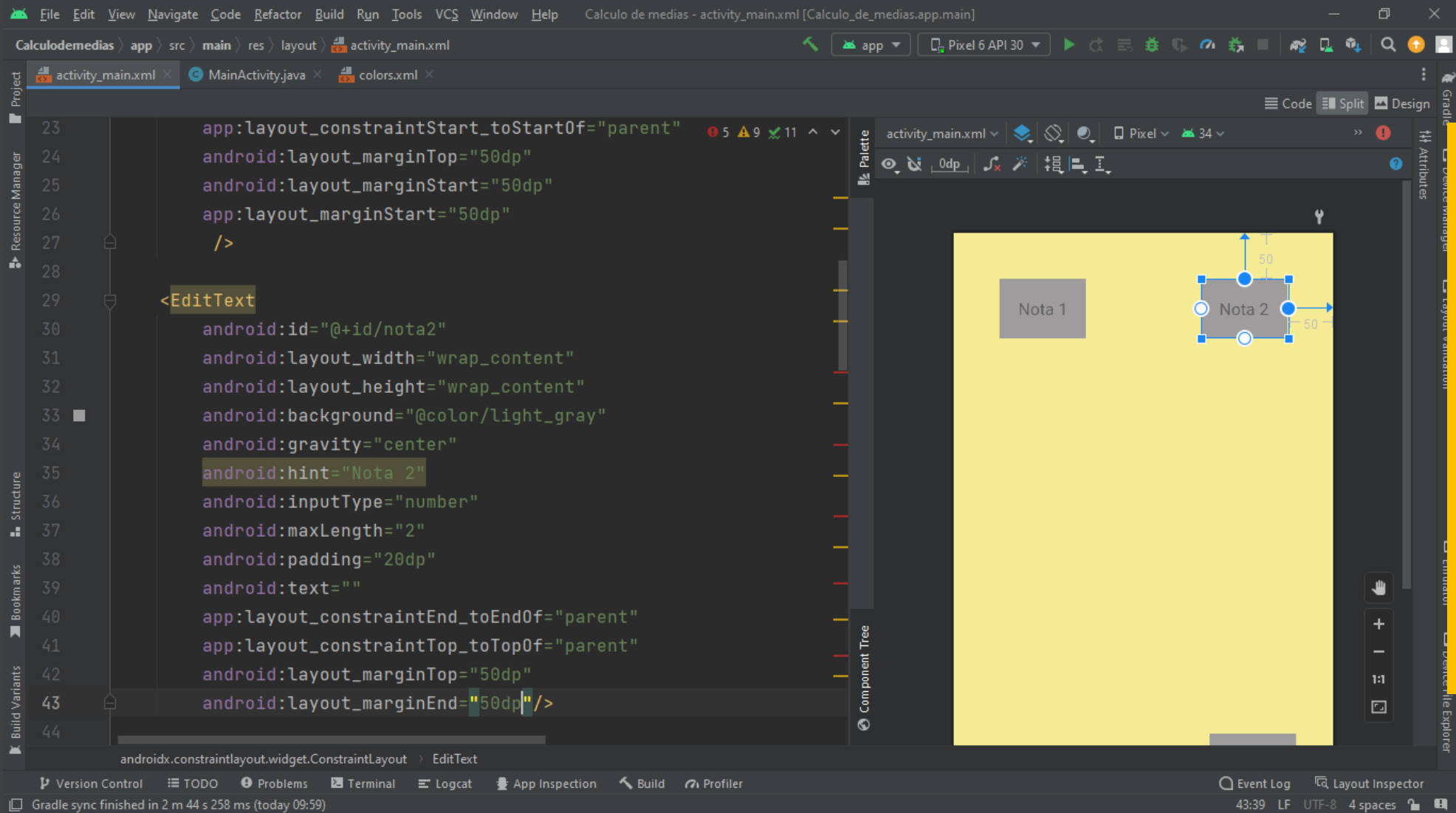
ConstraintLayout

Nome: ETIM PAMI Seu Nome Calculo de Media

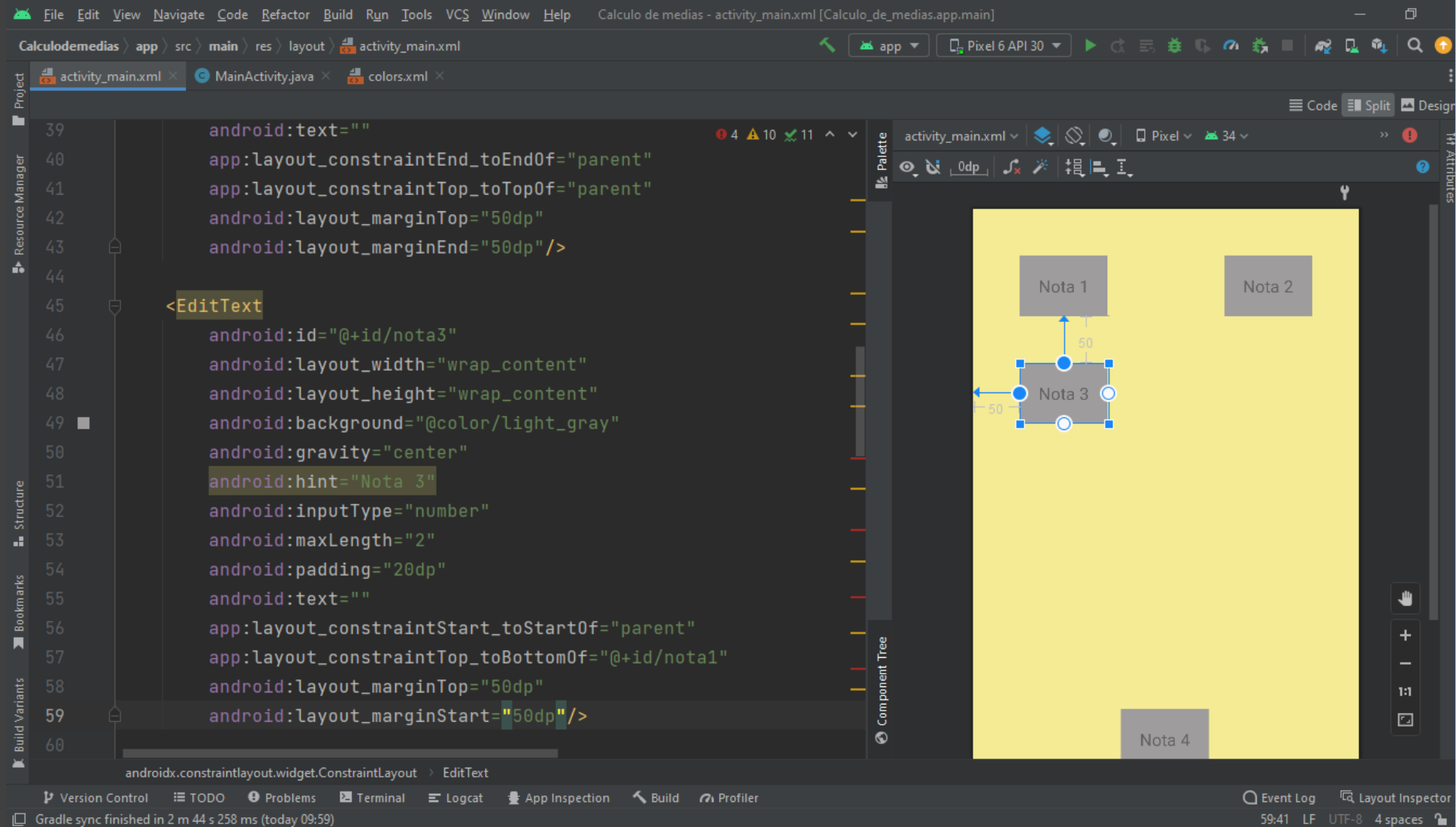
Package: com.android.seunome.calculodemedias



No constraint layout o elemento fica preso no lugar através de ancoras, que são essas alcas que ligam a esquerda e a direita. Percebam que ao colocar uma constraint ou ancora, a alca fica azul. Precisamos ao menos uma horizontal e uma vertical para conseguir uma coordenada de fixação. Aqui eu criei alcas de fixação a esquerda e no topo ligadas ao parent, ou seja ao container que eh o layout



Para a segunda nota, vou ligar acima no layout e a direita também no layout. O layout é o componente pai, onde estão todos os elementos dentro. Pai em inglês é parent, por isso que ligo em parent. Depois vamos colocar uma margem top e uma margem end de 50dp



A terceira nota vai ficar ligada a esquerda no container e acima vai ficar ligada a nota1. Iremos a parte de cima da nota3 a parte de baixo da nota1 e a parte esquerda da nota3 a parte esquerda do layout. O primeiro elemento, que eh a nota1 vai ligado ao layout ou pai. Os outros elementos abaixo dele vao ligados uns aos outros, assim se baixar o primeiro, todos os outros vao baixar junto

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Calculo de medias - activity_main.xml [Calculo_de_medias.app.main]

Calculodemedias > app > src > main > res > layout > activity_main.xml

activity_main.xml x MainActivity.java x colors.xml x

```
54 android:padding="20dp"
55 android:text=""
56 app:layout_constraintStart_toStartOf="parent"
57 app:layout_constraintTop_toBottomOf="@+id/nota1"
58 android:layout_marginTop="50dp"
59 android:layout_marginStart="50dp"/>
60
61 <EditText
62     android:id="@+id/nota4"
63     android:layout_width="wrap_content"
64     android:layout_height="wrap_content"
65     android:background="@color/light_gray"
66     android:gravity="center"
67     android:hint="Nota 4"
68     android:inputType="number"
69     android:maxLength="2"
70     android:padding="20dp"
71     android:text=""
72     app:layout_constraintEnd_toEndOf="parent"
73     android:layout_marginTop="50dp"
74     android:layout_marginEnd="50dp"
75     app:layout_constraintTop_toBottomOf="@+id/nota2"/>
```

activity_main.xml Pixel 6 API 30

0dp

Nota 1

Nota 2

Nota 3

Nota 4

50

50

Component Tree

Version Control TODO Problems Terminal Logcat App Inspection Build Profiler

Gradle sync finished in 2 m 44 s 258 ms (today 09:59)

Event Log Layout Inspector 61:5 LF UTF-8 4 spaces

Para a quarta nota, ligamos a parte de cima dela com a parte de baixo da nota2 e a parte direita dele a parte direita do layout

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Calculo de medias - activity_main.xml [Calculo_de_medias.app.main]

Calculodemedias > app > src > main > res > layout > activity_main.xml

activity_main.xml MainActivity.java colors.xml

```
70 android:padding="20dp"
71 android:text=""
72 app:layout_constraintEnd_toEndOf="parent"
73 android:layout_marginTop="50dp"
74 android:layout_marginEnd="50dp"
75 app:layout_constraintTop_toBottomOf="@+id/nota2" />
76
77 <EditText
78     android:id="@+id/numeroFalta"
79     android:layout_width="wrap_content"
80     android:layout_height="wrap_content"
81     android:background="@color/light_gray"
82     android:gravity="center"
83     android:hint="Numero de faltas"
84     android:inputType="number"
85     android:padding="20dp"
86     android:text=""
87     android:layout_marginTop="50dp"
88     app:layout_constraintEnd_toEndOf="parent"
89     app:layout_constraintStart_toStartOf="parent"
90     app:layout_constraintTop_toBottomOf="@+id/nota3" />
```

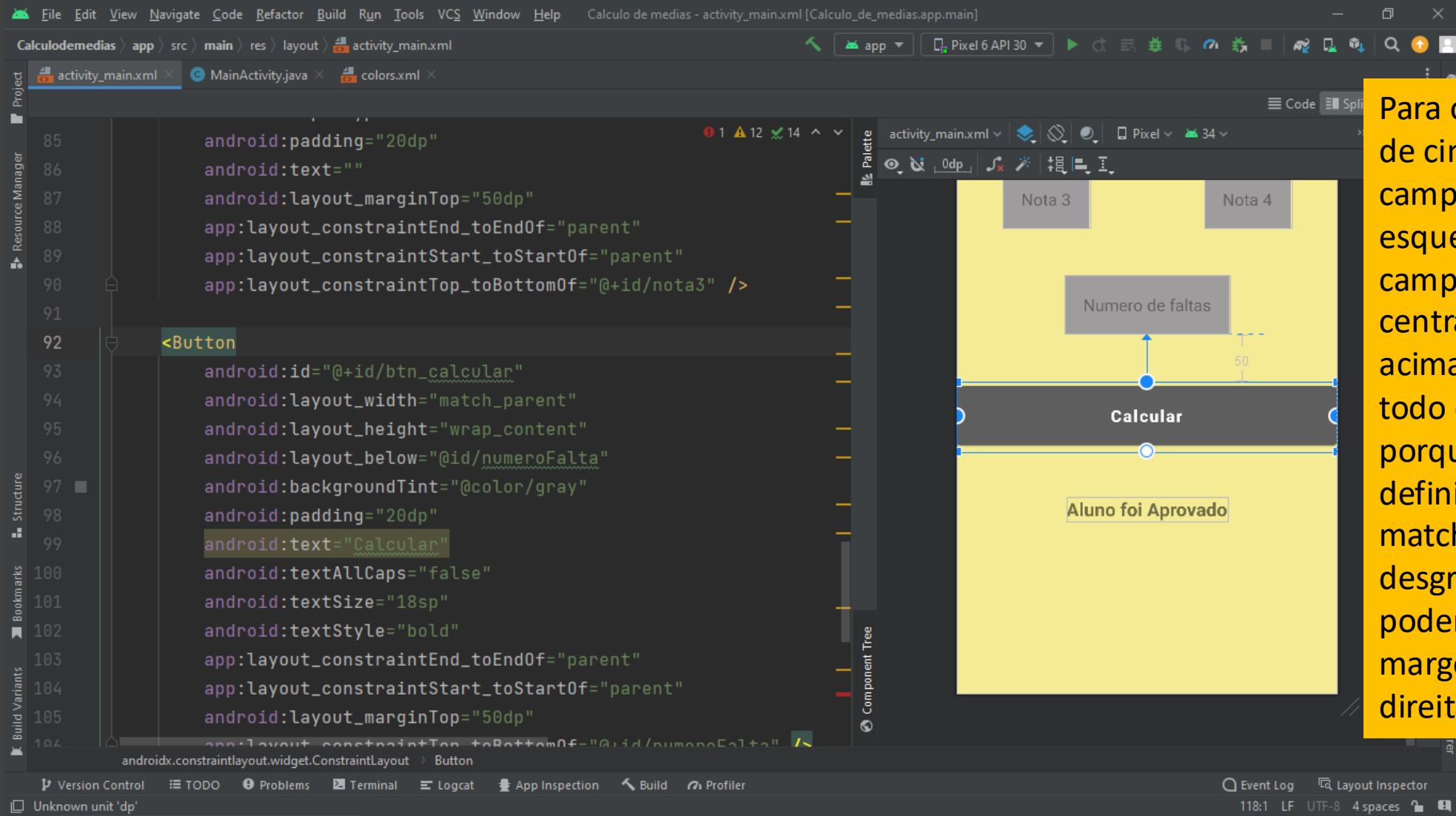
androidx.constraintlayout.widget.ConstraintLayout > EditText

Version Control TODO Problems Terminal Logcat App Inspection Build Profiler

Gradle sync finished in 2 m 44 s 258 ms (today 09:59)

Event Log Layout Inspector 87:39 LF UTF-8 4 spaces

Para o campo das faltas, ligamos a parte de cima do elemento a parte de baixo da nota3, e como quero que o elemento fique centralizado na tela, ligo a parte esquerda ao layout e a parte direita ao layout, assim crio uma mola puxando para um lado e outra puxando para o outro lado, e isso deixa o elemento centralizado



Para o botão, eu ligo a parte de cima a parte de baixo do campo das notas, e ligo a esquerda e a direita. O campo deveria ficar centralizado como o campo acima, mas ele esta pegando todo o tamanho da tela. Isso porque a largura esta definida como match_parent. Para desgrudar das margens podemos colocar uma margem esquerda e uma direita de uns 32dp

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Calculo de medias - activity_main.xml [Calculo_de_medias.app.main]

Calculodemedias > app > src > main > res > layout > activity_main.xml

activity_main.xml x MainActivity.java x colors.xml x

```
88 app:layout_constraintEnd_toEndOf="parent"
89 app:layout_constraintStart_toStartOf="parent"
90 app:layout_constraintTop_toBottomOf="@+id/nota3" />
91
92 <Button
93     android:id="@+id/btn_calcular"
94     android:layout_width="match_parent"
95     android:layout_height="wrap_content"
96     android:layout_below="@id/numeroFalta"
97     android:backgroundTint="@color/gray"
98     android:padding="20dp"
99     android:text="Calcular"
100     android:textAllCaps="false"
101     android:textSize="18sp"
102     android:textStyle="bold"
103     android:layout_marginEnd="32dp"
104     android:layout_marginStart="32dp"
105     app:layout_constraintEnd_toEndOf="parent"
106     app:layout_constraintStart_toStartOf="parent"
107     android:layout_marginTop="50dp"
108     app:layout_constraintTop_toBottomOf="@+id/numeroFalta" />
109
```

activity_main.xml v Pixel 6 API 30

0dp

Nota 1 Nota 2

Nota 3 Nota 4

Numero de faltas

Calcular

Aluno foi Aprovado

Component Tree

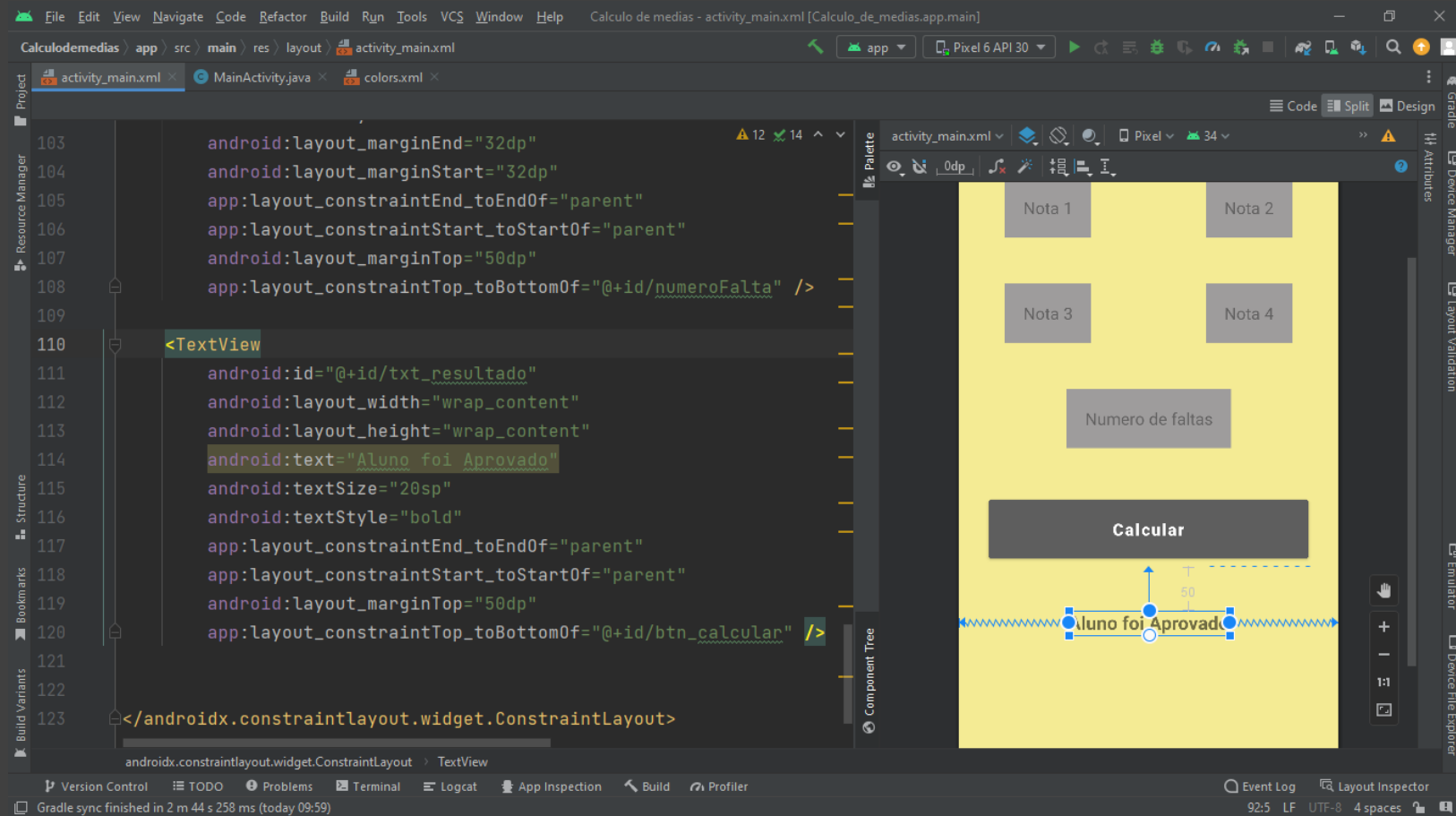
androidx.constraintlayout.widget.ConstraintLayout > Button

Version Control TODO Problems Terminal Logcat App Inspection Build Profiler

Gradle sync finished in 2 m 44 s 258 ms (today 09:59)

Event Log Layout Inspector 92:5 LF UTF-8 4 spaces

Agora o botão ficou mais legal sem estar colado nas extremidades do layout



O TextView vai ficar ligado pela parte de cima ao botão e ligo a esquerda e a direita, mas ele não estica porque o comprimento esta definido como wrap_content. Entao ele fica centralizado

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Calculo de medias - activity_main.xml [Calculo_de_medias.app.main]

Calculodemedias > app > src > main > res > layout > activity_main.xml

activity_main.xml x MainActivity.java x colors.xml x

```
88 app:layout_constraintEnd_toEndOf="parent"
89 app:layout_constraintStart_toStartOf="parent"
90 app:layout_constraintTop_toBottomOf="@+id/nota3" />
91
92 <Button
93     android:id="@+id/btn_calcular"
94     android:layout_width="match_parent"
95     android:layout_height="wrap_content"
96     android:layout_below="@id/numeroFalta"
97     android:backgroundTint="@color/gray"
98     android:padding="20dp"
99     android:text="Calcular"
100     android:textAllCaps="false"
101     android:textSize="18sp"
102     android:textStyle="bold"
103     android:layout_marginEnd="32dp"
104     android:layout_marginStart="32dp"
105     app:layout_constraintEnd_toEndOf="parent"
106     app:layout_constraintStart_toStartOf="parent"
107     android:layout_marginTop="50dp"
108     app:layout_constraintTop_toBottomOf="@+id/numeroFalta" />
109
```

activity_main.xml v Pixel 6 API 30

0dp

Nota 1 Nota 2

Nota 3 Nota 4

Numero de faltas

Calcular

Aluno foi Aprovado

Component Tree

androidx.constraintlayout.widget.ConstraintLayout > Button

Version Control TODO Problems Terminal Logcat App Inspection Build Profiler

Gradle sync finished in 2 m 44 s 258 ms (today 09:59)

Event Log Layout Inspector 92:5 LF UTF-8 4 spaces

Agora o botão ficou mais legal sem estar colado nas extremidades do layout

Tipos de medias para os elementos de visualizacao

Existem três tipos de medidas que podemos utilizar para configurar os elementos e essas medidas podem ser usadas tanto na horizontal quanto na vertical:

Wrap_content: nesse tipo de medida o tamanho do elemento se ajusta ao tamanho do conteúdo. Então se temos um texto pequeno dentro do elemento, o elemento também fica pequeno. Aumentando o texto, o elemento se ajusta para caber.

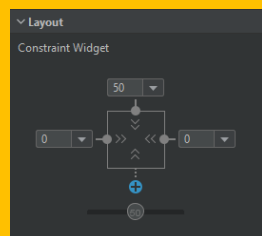
Match_parent: Nesse tipo de medida, o elemento ocupa todo espaço disponível do dispositivo.

Fixed: tamanho fixo.

No modo design podemos ver três tipos de ícones para as medidas:

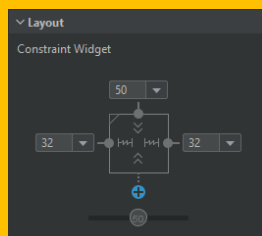
Existem três tipos de medidas que podemos utilizar para configurar os elementos e essas medidas podem ser usadas tanto na horizontal quanto na vertical:

Wrap_content



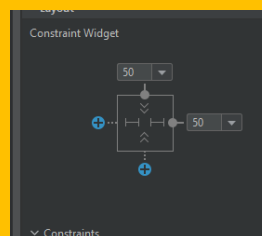
Aqui o ícone é duas setas na horizontal e duas na vertical o que indica que os dois estão ajustáveis

Match_parent:



Aqui o ícone é como uma mola indicando que ele vai se ajustar ao dispositivo não ao elemento

Fixed:



Aqui temos uma medida fixa e o ícone mostra um traco delimitado

Passos para criação de um app:

No arquivo xml:

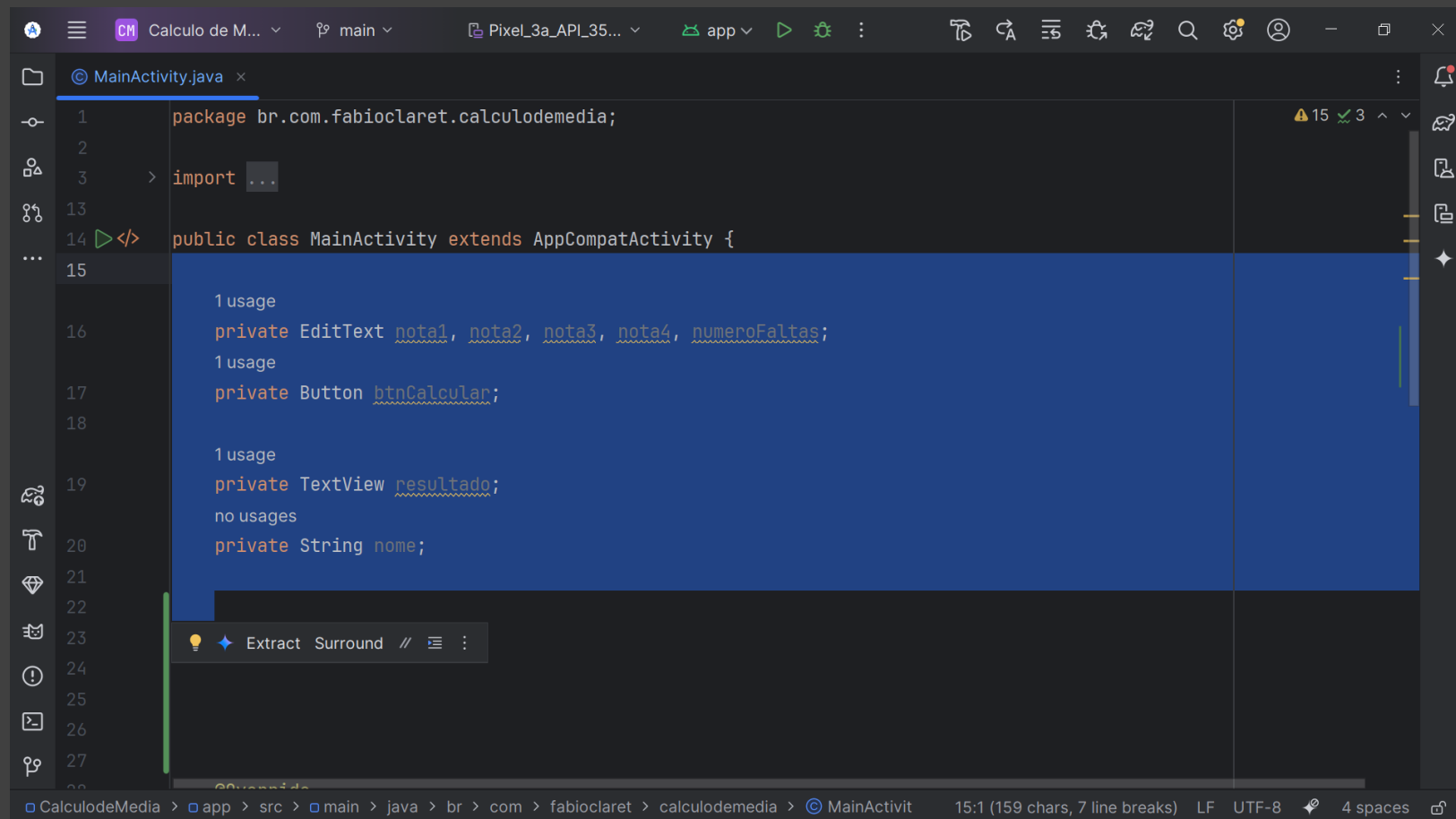
=====

1. Criação da tela (layout)
2. Identificação dos elementos (colocar um id em cada elemento)

Na Classe:

=====

1. Para cada elemento do layout, declarar um elemento do mesmo tipo na classe
2. Ligar o layout com a classe através do método findViewById()
3. Adicionar evento de clique para os botões.
4. Desenvolver a logica. (A logica será desenvolvida dentro do método onCreate, que é o primeiro método que roda ao abrirmos um app.



```
1 package br.com.fabioclaret.calculodemedias;  
2  
3 import ...  
4  
13  
14 public class MainActivity extends AppCompatActivity {  
15  
16     1 usage  
    private EditText nota1, nota2, nota3, nota4, numeroFaltas;  
    1 usage  
17     private Button btnCalcular;  
18  
19     1 usage  
    private TextView resultado;  
    no usages  
20     private String nome;  
21  
22  
23  
24  
25  
26  
27  
28
```

CalculodeMedia > app > src > main > java > br > com > fabioclaret > calculodemedias > MainActivity 15:1 (159 chars, 7 line breaks) LF UTF-8 4 spaces

Logo no inicio da classe, vamos declarar todos os elementos do layout que iremos usar. Esses elementos são classes.

E toda classe sempre começa com letra maiúscula

Aqui temos a classe MainActivity que esta herdando da classe AppCompatActivity

```
14 public class MainActivity extends AppCompatActivity {
15     private String nome;
16
17
18
19
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23
24         super.onCreate(savedInstanceState);
25         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
26         setContentView(R.layout.activity_main);
27
28         initComponents();
29
30
31
32         v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
33         return insets;
34     });
35 }
36
37
38
39
40
```

37 private void initComponents() {

38

39 } WindowInsetsCompat insets) -> {

});

v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);

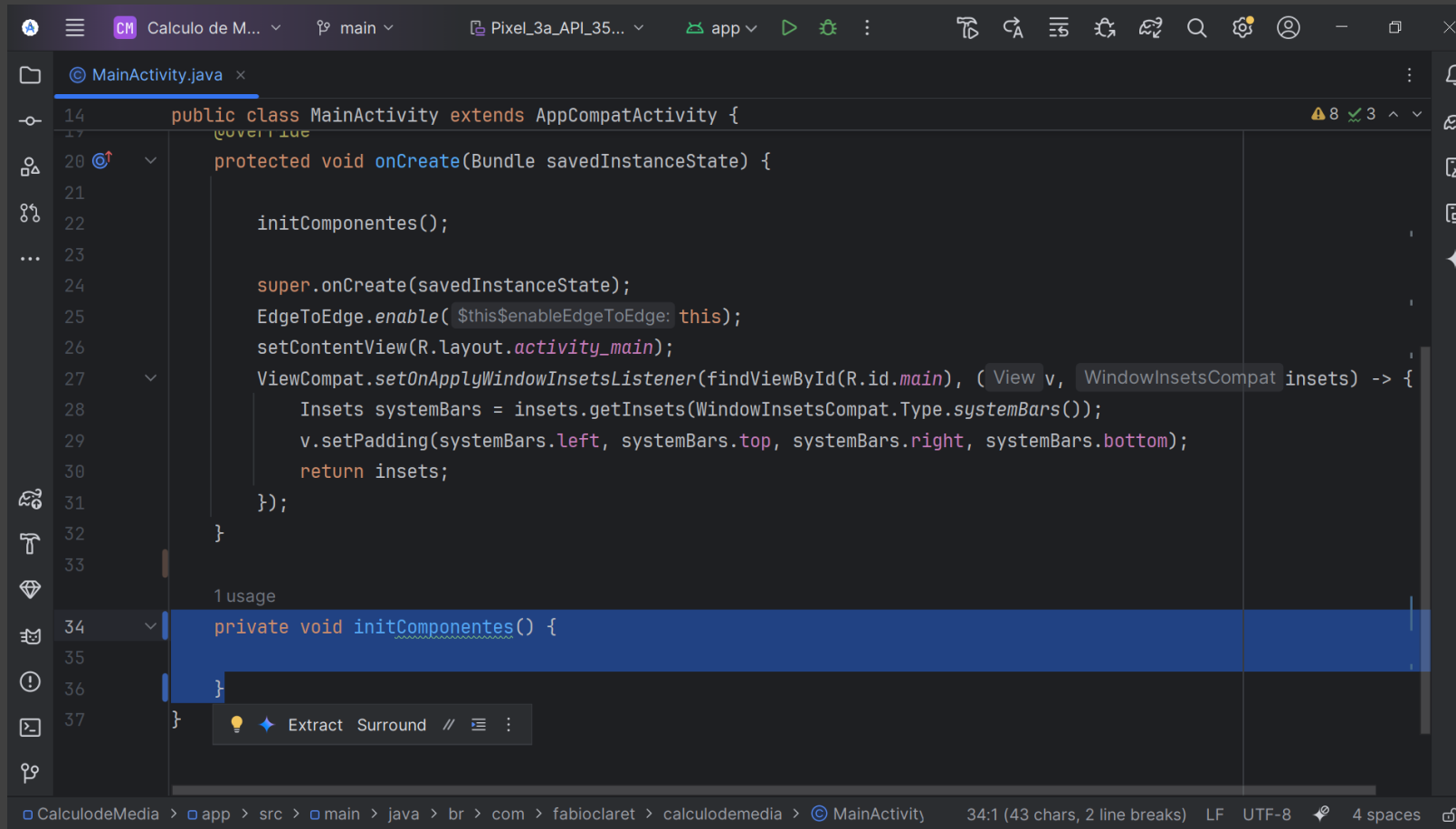
return insets;

});

}

CalculodeMedia > app > src > main > java > br > com > fabioclaire > calculodemedia > MainActivity > onCreate 28:19 LF UTF-8 4 spaces

Aqui temos o método `onCreate`, esse método é abstrato na classe `AppCompatActivity`, então aqui temos que sobrescreve-lo. E dentro dele (após o `setContentView`) vamos criar um método `initComponents()` Metodos e variáveis são sempre escritos com letra minúscula no inicio. Para criar basta escrever como esta aqui: `initComponents()`; Ele vai ficar vermelho, então clicamos na lâmpada e pedimos para criar o método.



```
14 public class MainActivity extends AppCompatActivity {  
17     @Override  
20     protected void onCreate(Bundle savedInstanceState) {  
21  
22         initComponents();  
23  
24         super.onCreate(savedInstanceState);  
25         EdgeToEdge.enable( $this$enableEdgeToEdge: this);  
26         setContentView(R.layout.activity_main);  
27         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), ( View v, WindowInsetsCompat insets) -> {  
28             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());  
29             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);  
30             return insets;  
31         });  
32     }  
33  
34     private void initComponents() {  
35  
36     }  
37 }
```

1 usage

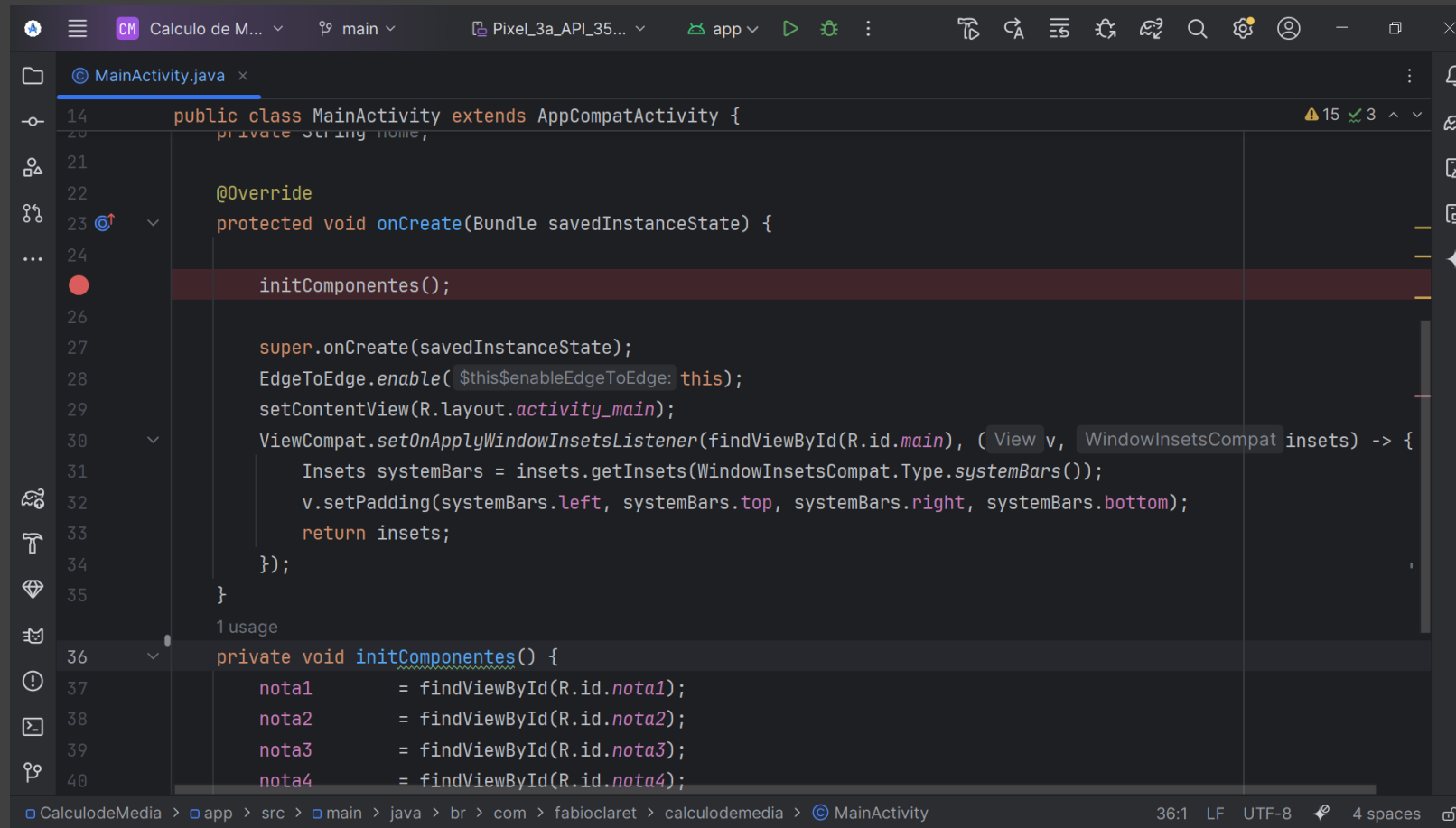
Extract Surround // ≡ ⋮

CalculodeMedia > app > src > main > java > br > com > fabioclaret > calculodemedia > MainActivity 34:1 (43 chars, 2 line breaks) LF UTF-8 4 spaces

Esse método será criado aqui dentro da MainActivity, depois do método onCreate. Todo método e classe sempre tem um escopo, que é o bloco entre chaves que delimitam esse métodos e classes.


```
14 public class MainActivity extends AppCompatActivity {
23     protected void onCreate(Bundle savedInstanceState) {
30         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (View v, WindowInsetsCompat insets) -> {
31             Insets systemBars = Insets.getInsets(WindowInsetsCompat.Type.systemBars());
32             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
33             return insets;
34         });
35     }
36
37
38
39     private void initComponents() {
40         nota1 = findViewById(R.id.nota1);
41         nota2 = findViewById(R.id.nota2);
42         nota3 = findViewById(R.id.nota3);
43         nota4 = findViewById(R.id.nota4);
44         numeroFaltas = findViewById(R.id.numero_faltas);
45         btnCalcular = findViewById(R.id.btn_calcular);
46         resultado = findViewById(R.id.txt_resultado);
47     }
48
49 }
```

Nesse método, a gente vai ligar todos os elementos do layout com a classe. Então criamos a variável nota1 que é do tipo EditText para receber o valor que será digitado lá no layout no campo nota1. findViewById() é o método que localiza um elemento pelo seu id, por isso temos que colocar o id nos elementos lá no layout.



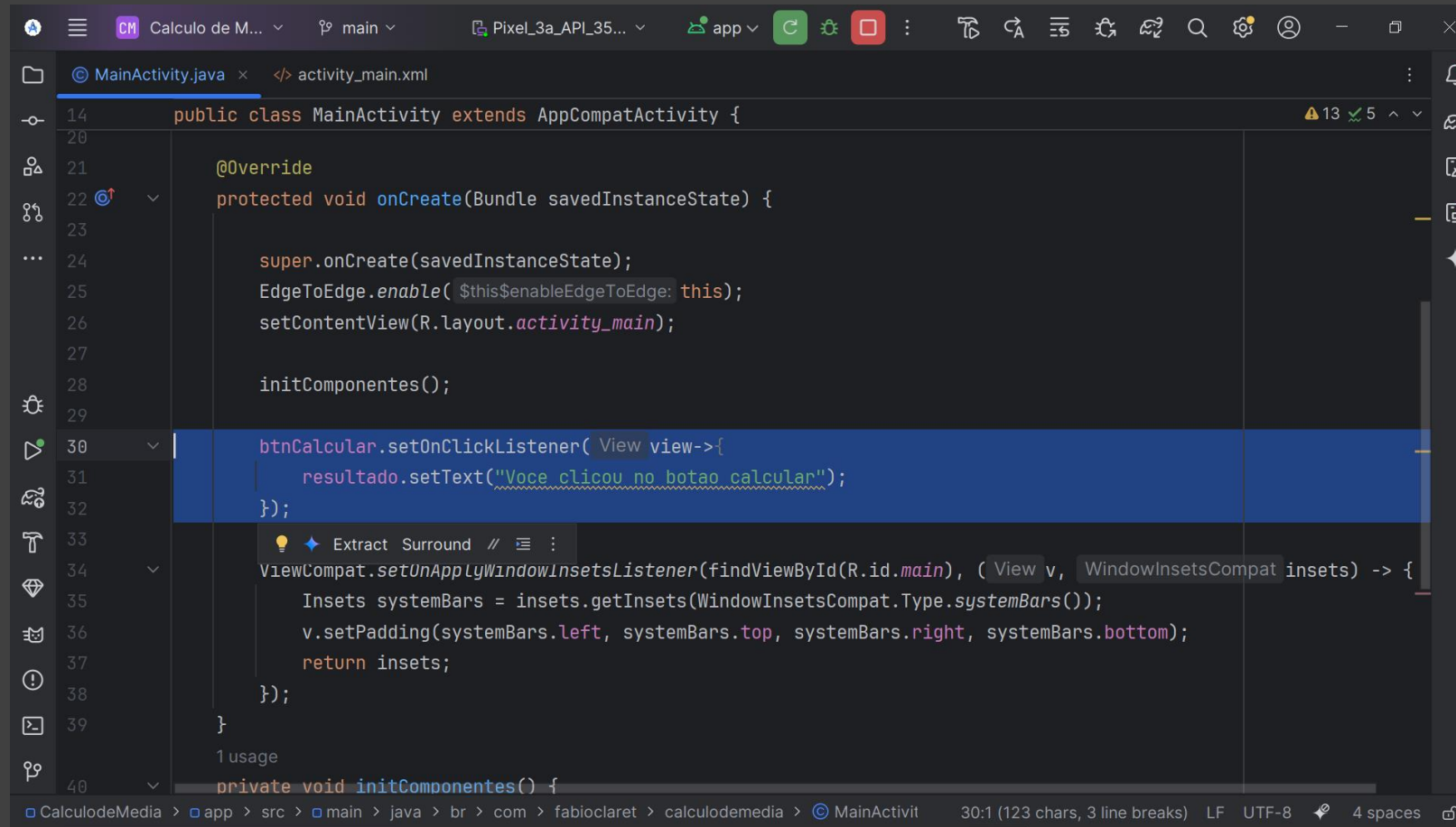
```
14 public class MainActivity extends AppCompatActivity {
20     private String nome;
21
22     @Override
23     protected void onCreate(Bundle savedInstanceState) {
24         initComponents();
25
26
27         super.onCreate(savedInstanceState);
28         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
29         setContentView(R.layout.activity_main);
30         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), ( View v, WindowInsetsCompat insets) -> {
31             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
32             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
33             return insets;
34         });
35     }
36     private void initComponents() {
37         nota1 = findViewById(R.id.nota1);
38         nota2 = findViewById(R.id.nota2);
39         nota3 = findViewById(R.id.nota3);
40         nota4 = findViewById(R.id.nota4);
```

15 3 ^

1 usage

CalculodeMedia > app > src > main > java > br > com > fabioclaire > calculodemedia > MainActivity 36:1 LF UTF-8 4 spaces

Aqui então na linha 25 o programa desvia para o método initComponents, que vai ligar o layout com a classe, depois volta na linha abaixo que eh a 26



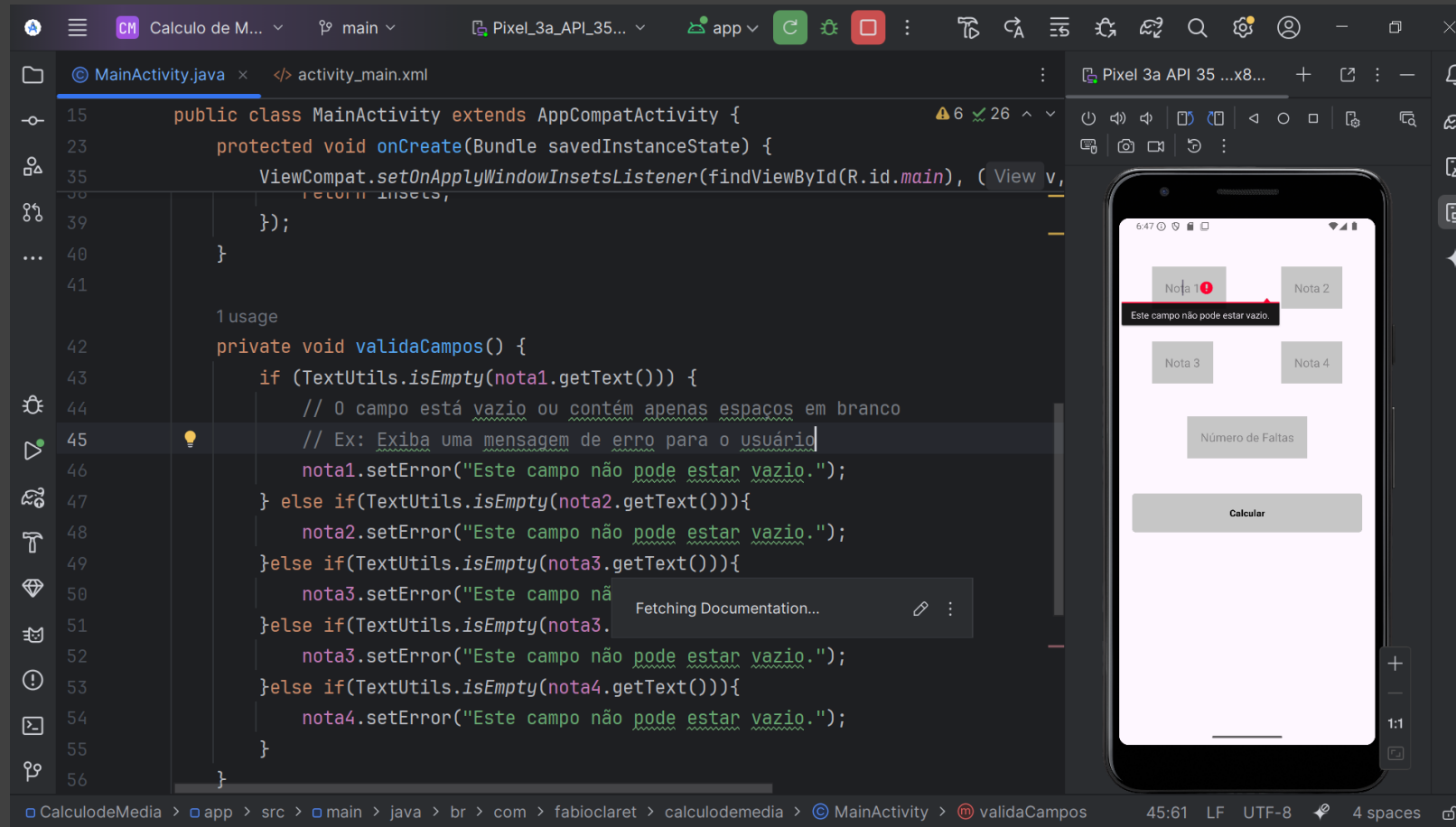
```
14 public class MainActivity extends AppCompatActivity {
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23
24         super.onCreate(savedInstanceState);
25         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
26         setContentView(R.layout.activity_main);
27
28         initComponents();
29
30         btnCalcular.setOnClickListener( View view->{
31             resultado.setText("Voce clicou no botao calcular");
32         });
33
34         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), ( View v, WindowInsetsCompat insets) -> {
35             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
36             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
37             return insets;
38         });
39     }
40     private void initComponents() {
```

Fizemos os passos 1, 2 e 3, agora vamos criar o evento de click do botão calcular. Fazemos isso dentro do método onCreate. Esse método tem que ficar depois do setContentView que é o método que desenha a tela.

The screenshot shows an IDE window with the file `MainActivity.java` open. The code is in Java and extends `AppCompatActivity`. The `onCreate` method is visible, and a lambda expression is being used to set an `onClick` listener for `btnCalcular`. A code completion menu is open, suggesting the method `validaCampos()`. The menu options are: `Create method 'validaCampos' in 'MainActivity'`, `Rename reference`, and `Extract to method reference`. The `validaCampos` method is also visible in the code, defined as a private void method. The IDE's status bar at the bottom shows the file path: `lodeMedia > app > src > main > java > br > com > fabioclairet > calculodemedia > MainActivity > onCreate > Lambda`. The status bar also shows the time `31:28`, the file encoding `UTF-8`, and the number of spaces `4`.

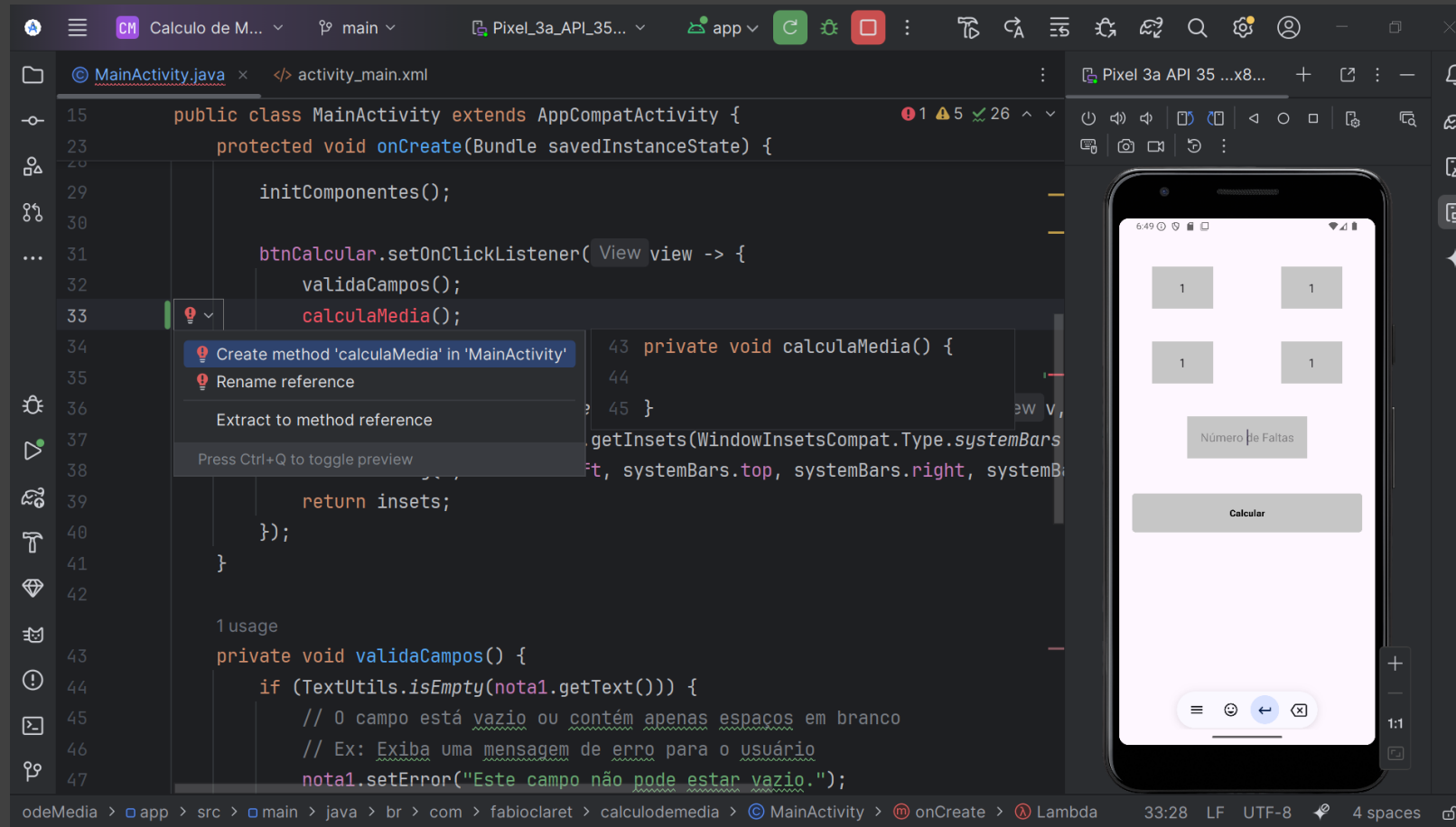
```
14 public class MainActivity extends AppCompatActivity {
22     protected void onCreate(Bundle savedInstanceState) {
23
24         super.onCreate(savedInstanceState);
25         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
26         setContentView(R.layout.activity_main);
27
28         initComponents();
29
30         btnCalcular.setOnClickListener( View view -> {
31             validaCampos();
32
33             // Create method 'validaCampos' in 'MainActivity'
34             // Rename reference
35             // Extract to method reference
36             // Press Ctrl+Q to toggle preview
37
38             return insets;
39         });
40
41     private void initComponents() {
42
43         // FindViewById(R.id.btnCalcular)
44         btnCalcular = findViewById(R.id.btnCalcular);
45         btnCalcular.setOnClickListener( View view, WindowInsetsCompat insets) -> {
46             getInsets(WindowInsetsCompat.Type.systemBars());
47             insets = WindowInsetsCompat.systemBars().toWindowInsets(systemBars.top, systemBars.right, systemBars.bottom);
48         });
49     }
50
51     private void initComponents() {
52         // FindViewById(R.id.btnCalcular)
53         btnCalcular = findViewById(R.id.btnCalcular);
54         btnCalcular.setOnClickListener( View view, WindowInsetsCompat insets) -> {
55             getInsets(WindowInsetsCompat.Type.systemBars());
56             insets = WindowInsetsCompat.systemBars().toWindowInsets(systemBars.top, systemBars.right, systemBars.bottom);
57         });
58     }
59 }
```

Quando a gente programa, temos que pensar em quem vai usar o app (usuário). E temos que prever que ele pode fazer algo errado, e contornar o erro. Aqui a gente vai calcular a media, então todos os campos de nota tem que estar preenchidos. Temos que garantir isso, criando um método para forçar o usuário digitar tudo. Fazemos isso criando o `validaCampos` e pedindo para a IDE criar.

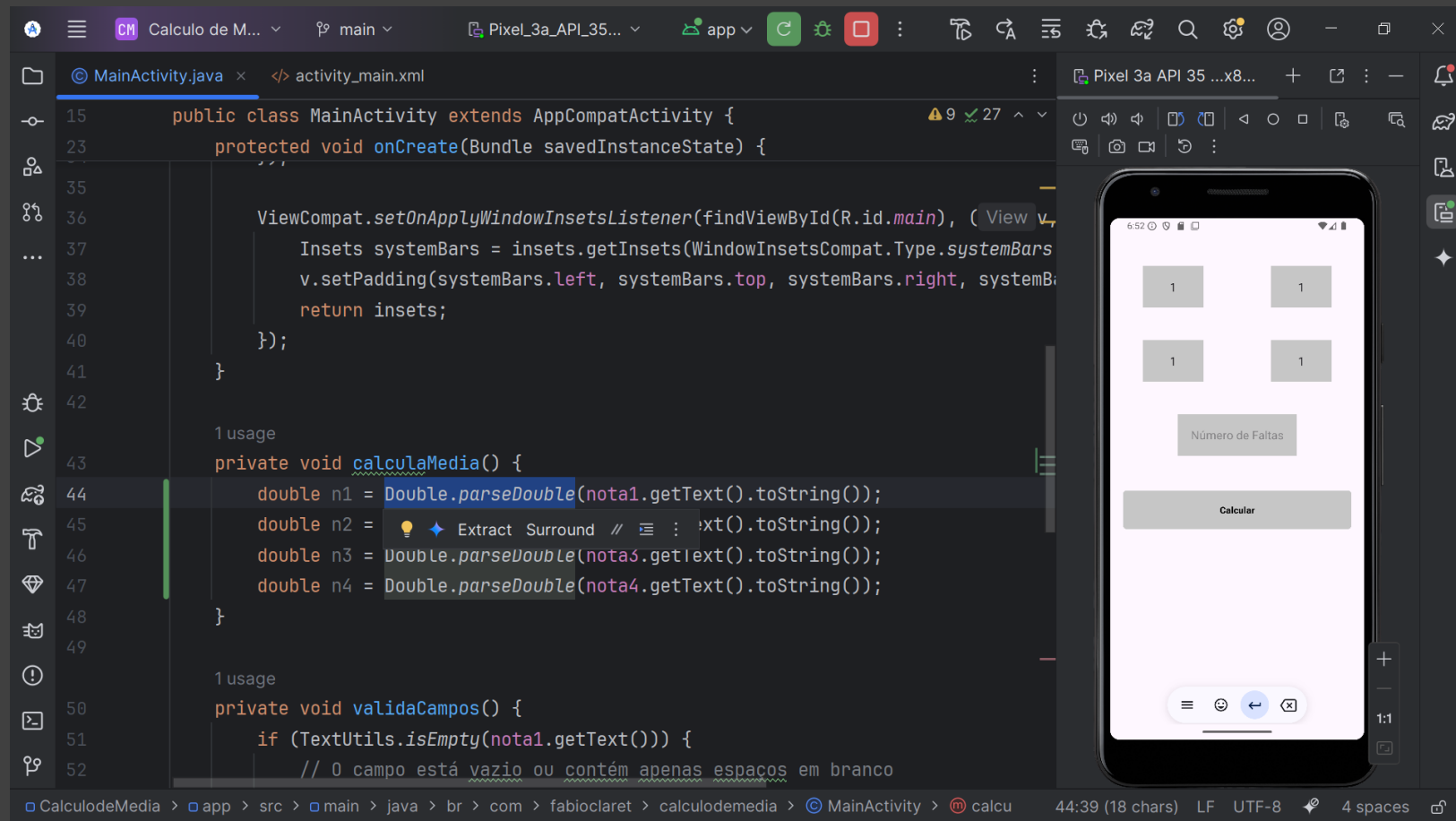


A classe `TextUtils.isEmpty` checa se o campo está vazio ou com espaços.

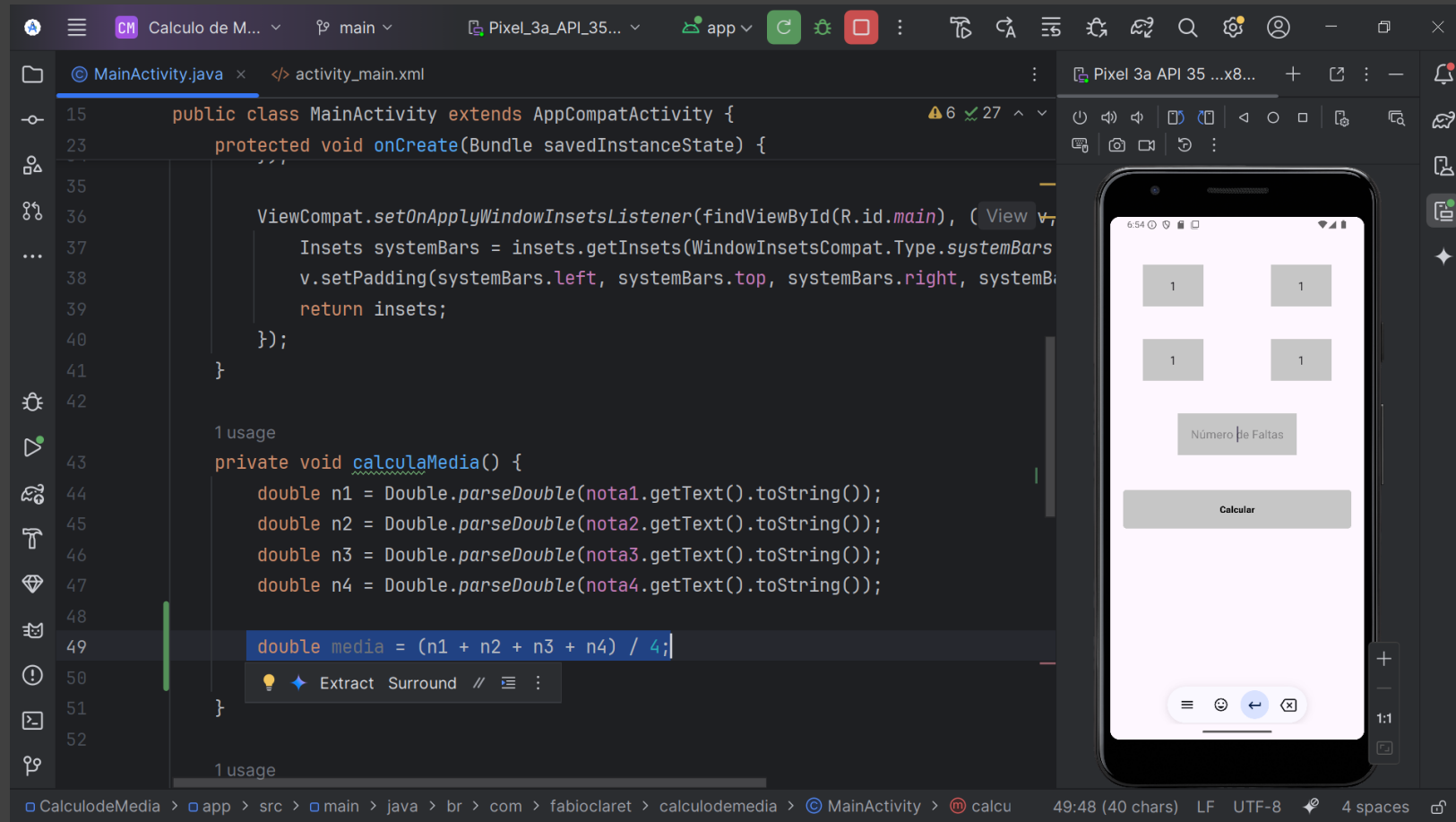
Se estiver, ao clicar no botão calcular, ele não vai e fica esperando até que todos os campos estejam preenchidos.



Digitamos o nome do método para calcular a media e pedimos para o Android criar o metodo



Toda vez que digitamos qualquer numero em uma caixa de texto, ele vem como texto, então temos que converter para o que queremos. E aqui queremos números do tipo double, pois podem ter decimais. Esse processo de converter se chama casting



Aqui eu calculo a media


```
16 public class MainActivity extends AppCompatActivity {
44     private void calculaMedia() {
45         double n1 = Double.parseDouble(nota1.getText().toString());
46         double n2 = Double.parseDouble(nota2.getText().toString());
47         double n3 = Double.parseDouble(nota3.getText().toString());
48         double n4 = Double.parseDouble(nota4.getText().toString());
49         double media = (n1 + n2 + n3 + n4) / 4;
50         double faltas = Double.parseDouble(numeroFaltas.getText().toString());
51
52         if( media > 7 ){
53             if( faltas < 20 ) {
54                 resultado.setTextColor(Color.parseColor( colorString: "#437845"));
55                 resultado.setText("Aluno Aprovado com media " + media);
56             }else{
57                 resultado.setTextColor(Color.parseColor( colorString: "#F44336"));
58                 resultado.setText("Excesso de falta " + faltas);
59             }
60         }else{
61             resultado.setTextColor(Color.parseColor( colorString: "#F44336"));
62             resultado.setText("Aluno Retido com media " + media);
63         }
64     }
}
```

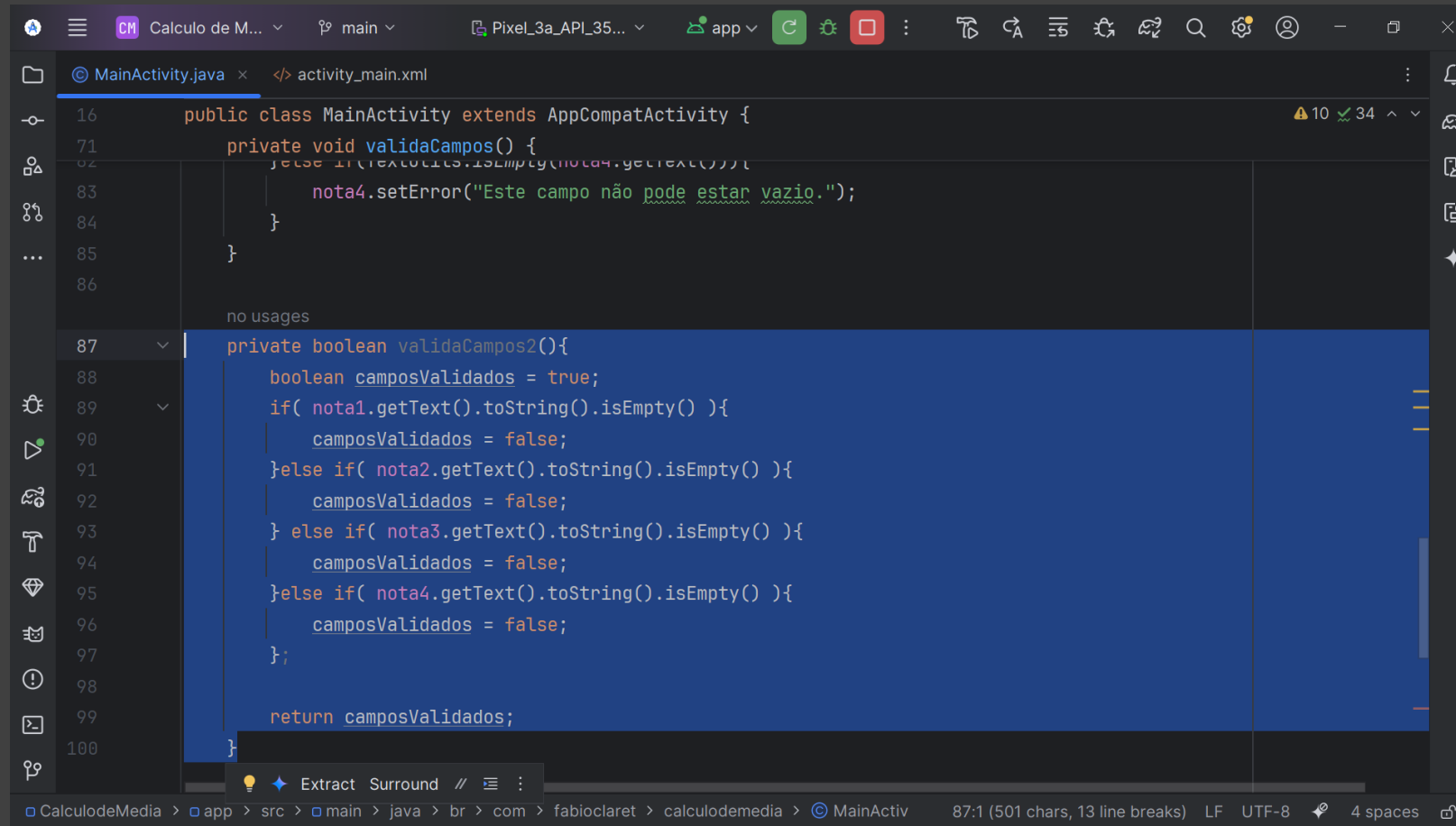
Aqui é pura logica:

Se a media eh maior que 7 e o numero de falta menor que 20, aprovado.

Se a media eh maior que 7 mas o numero de falta eh maior que 20, reprovado

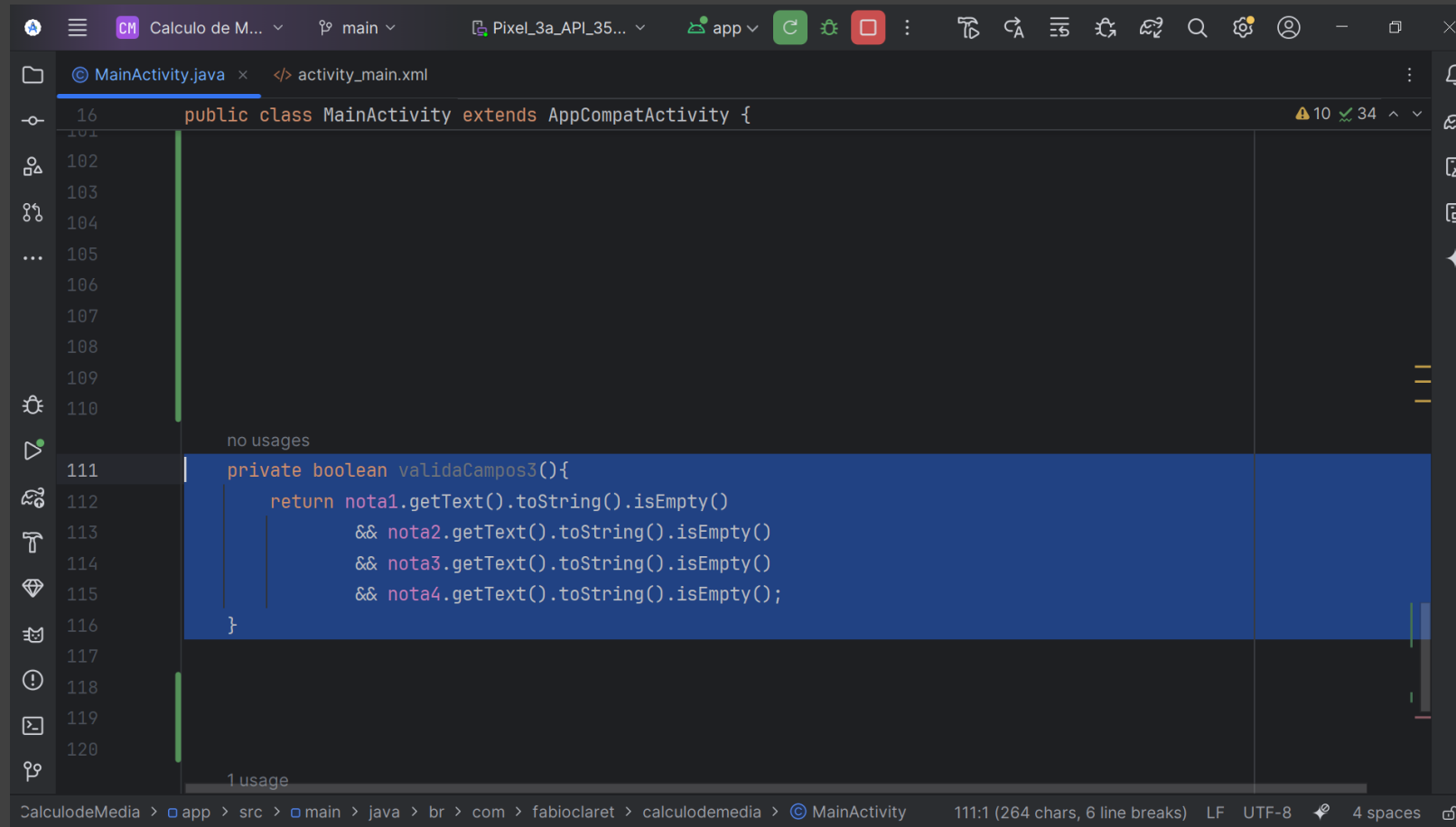
Se media eh menor que 7, aluno reprovado por media

E se o numero de falta for maior que 20, aluno retido por falta



```
16 public class MainActivity extends AppCompatActivity {
71     private void validaCampos() {
82         if (TextUtils.isEmpty(nota4.getText())) {
83             nota4.setError("Este campo não pode estar vazio.");
84         }
85     }
86
87     private boolean validaCampos2(){
88         boolean camposValidados = true;
89         if( nota1.getText().toString().isEmpty() ){
90             camposValidados = false;
91         }else if( nota2.getText().toString().isEmpty() ){
92             camposValidados = false;
93         } else if( nota3.getText().toString().isEmpty() ){
94             camposValidados = false;
95         }else if( nota4.getText().toString().isEmpty() ){
96             camposValidados = false;
97         };
98
99         return camposValidados;
100     }
```

Outro jeito de validar campos, crio uma variável verdadeira e se algum campo for vazio e torno essa variável falsa E no final, retorno a variável. La em cima, basta checar se o validaCampos2 eh verdadeiro ou falso



```
16 public class MainActivity extends AppCompatActivity {  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111 private boolean validaCampos3(){  
112     return nota1.getText().toString().isEmpty()  
113         && nota2.getText().toString().isEmpty()  
114         && nota3.getText().toString().isEmpty()  
115         && nota4.getText().toString().isEmpty();  
116 }  
117  
118  
119  
120
```

no usages

1 usage

CalculodeMedia > app > src > main > java > br > com > fabioclairet > calculodemedi... > MainActivity 111:1 (264 chars, 6 line breaks) LF UTF-8 4 spaces

Aqui otimizei a logica para não precisar de usaro o if

Apostila para o curso de PAMII

Professores:
Fabio Claret
Cleiton Silva