Buse Çarık 27/06/18

INTERRA R&D Material Design

Material Design

Text Fields

https://github.com/InterraMaterialDesign/MaterialTextField

**INDEX**

1. **INTRODUCTION………………………………………………………………………3**
   1. **Purpose of this document…………………………………………………………..3**
   2. **Overview…………………………………………………………………………….3**
2. **WHAT IS TEXTFIELD………………………………………………………………...4**
3. **REQUIREMENTS FOR IMPLEMENTATION…………………………………….10**
4. **AREAS OF USAGE…………………………………………………………………... 10**
5. **MATERIAL TEXTFIELD...…………………………………………………………..11**
   1. **Types………………………………………………………………………………..11**
   2. **Height Variations…………………………………………………………………..11**
   3. **Styles………………………………………………………………………………...11**
6. **REFERENCES………………………………………………………………………...26**

**INTRODUCTION**

**PURPOSE OF THIS DOCUMENT**

The aim of this project is analyzing material design text fields’ types, styles and attributes.

**OVERVIEW**

The project that is described in this document, includes material text fields’ types and styles. The types are filled text fields and outlined text fields and the styles are

Base.Widget.MaterialComponents.TextInputLayout

Widget.Design.TextInputLayout

Widget.MaterialComponents.TextInputLayout.FilledBox

Widget.MaterialComponents.TextInputLayout.FilledBox.Dense

Widget.MaterialComponents.TextInputLayout.OutlineBox

Widget.MaterialComponents.TextInputLayout.OutlineBox.Dense

**Text Field**

Text fields allows users to enter texts. When you touch the text field, the keyboard shows up automatically. Text fields also allows other actions such as text selection (copy, paste, cut) and auto completion. To take the inputs, EditText is used in layouts.

Text fields can take different input types with ***android:inputType*** attribute. The input types are **text, textPassword, email, phone, date, textUri, number, numberDecimal, numberPassword, time**… But you cannot paste different types to the field. The keyboard’s type is changed according to your input type selection. For example, if you choose email as input type, the keyboard has *.com* and @ symbols (Examples can be found below). inputType attribute allows multiple options too. Besides the keyboard type, you can also specify keyboard behaviors by separating these with '**|**'. These behaviors are **textCapSentences, textCapWords, textAutoCorrect** … Similarly, with inputType, it can be single line or multiple line.

Besides the type of the keyboard, inputType also allows to specify an action to use when the input is completed. For example, you can search your input, or you can send your input (like a mail). This option is set with ***android:imeOptions*** attribute.

android:imeOptions="actionNext"

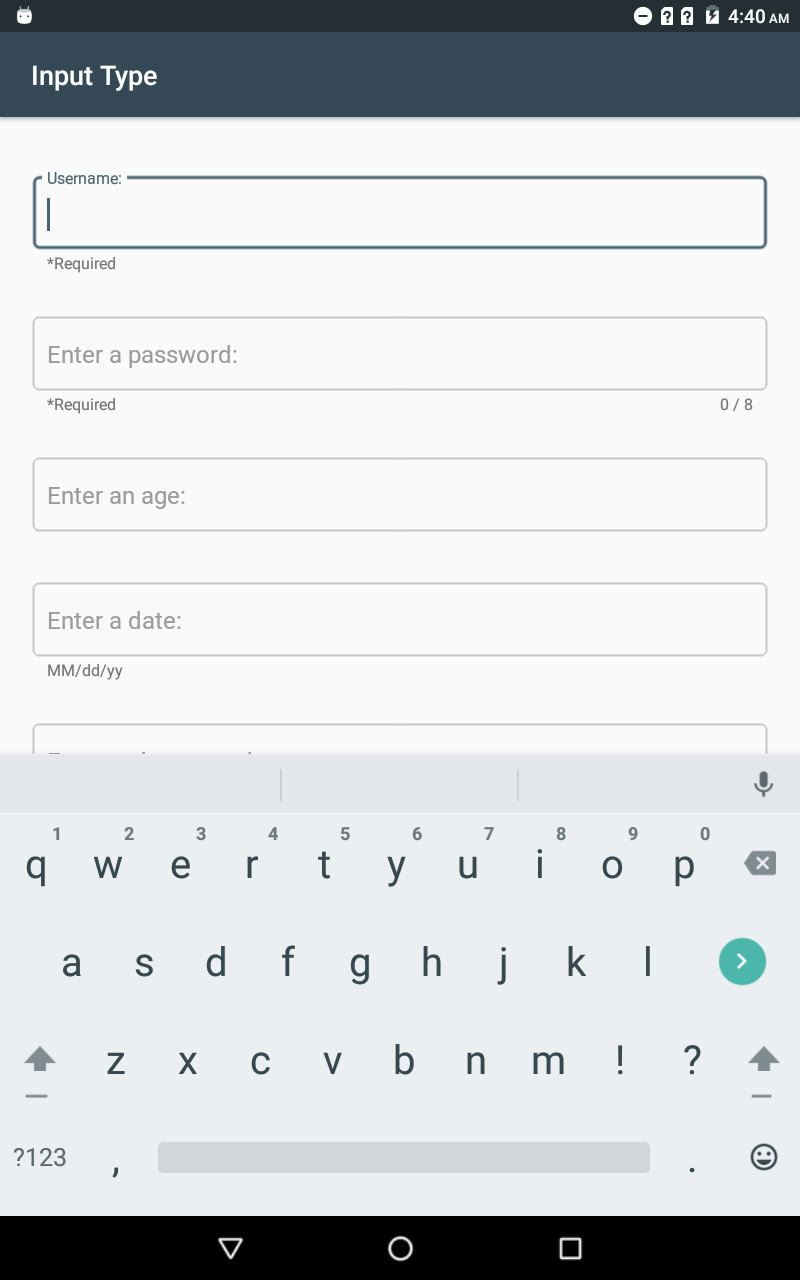
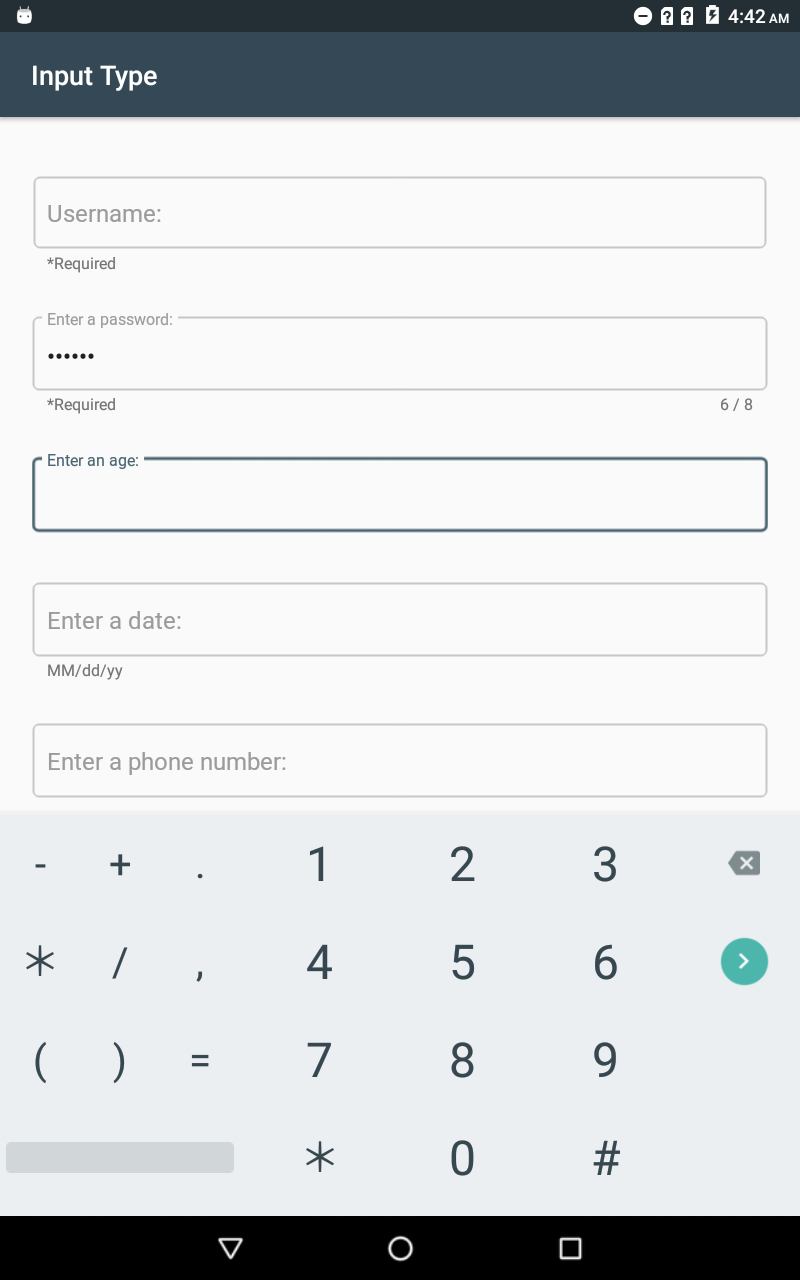
android:imeOptions="actionDone"

android:imeOptions="actionSend"

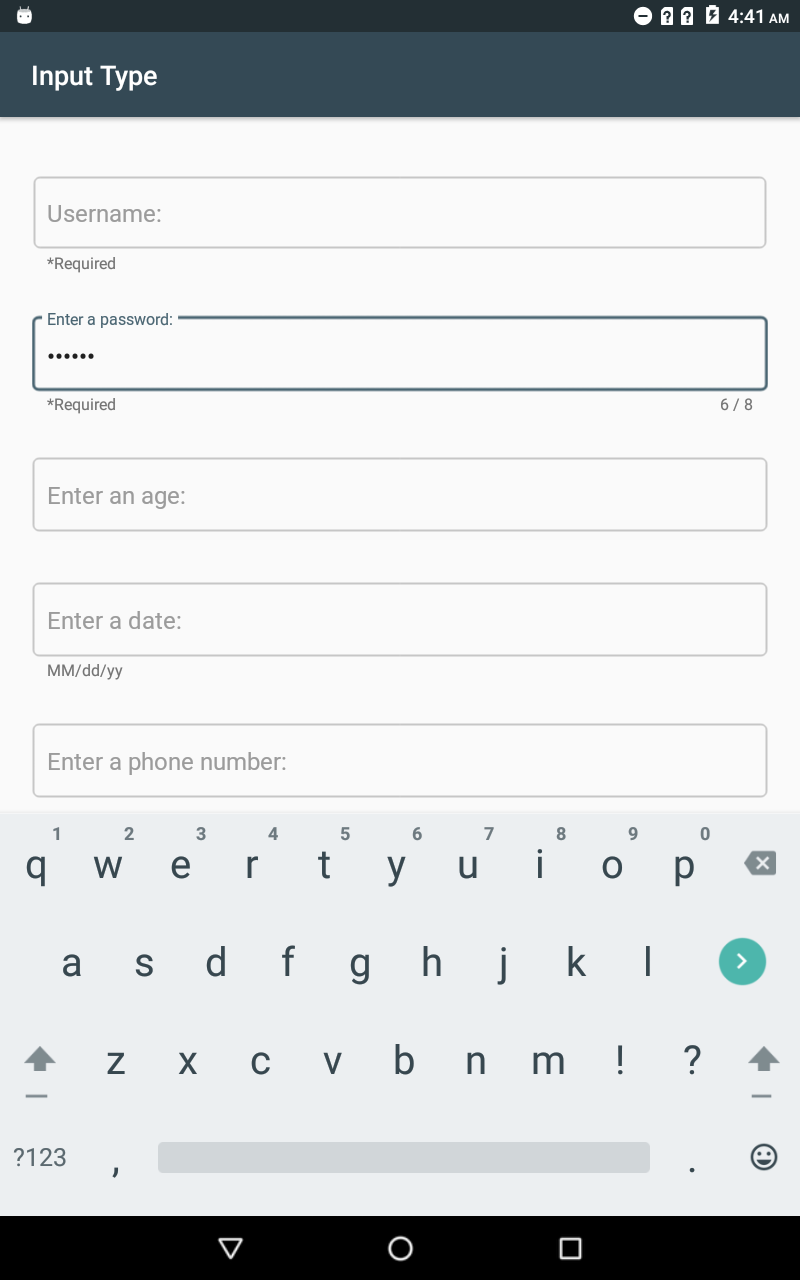
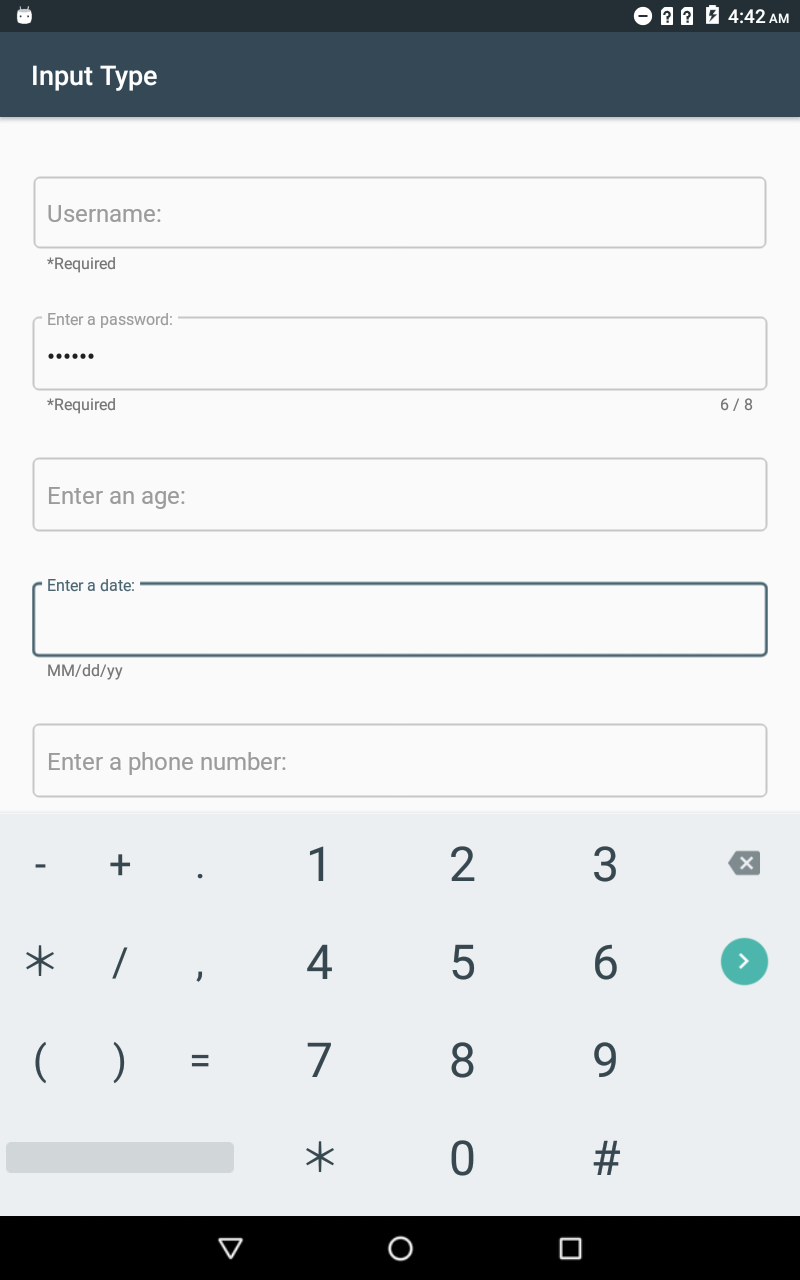
 

android:imeOptions="actionSearch"

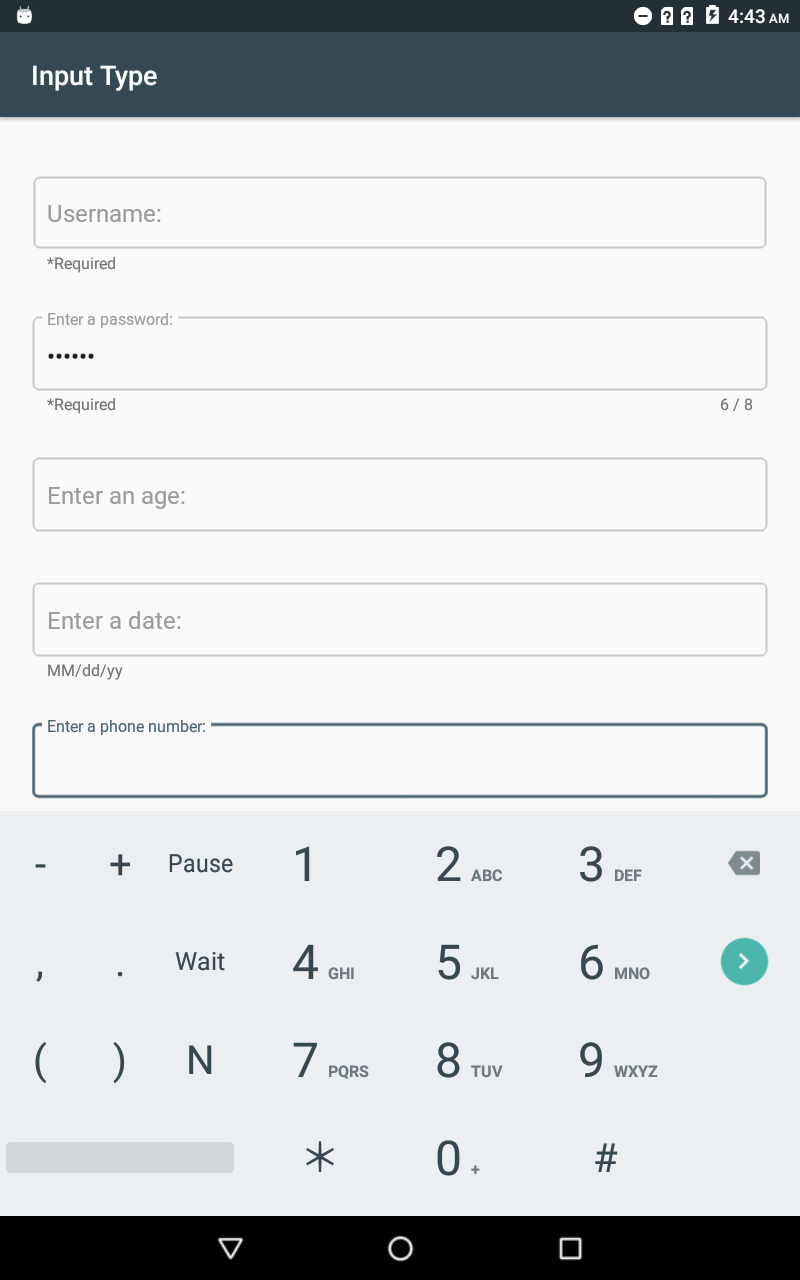
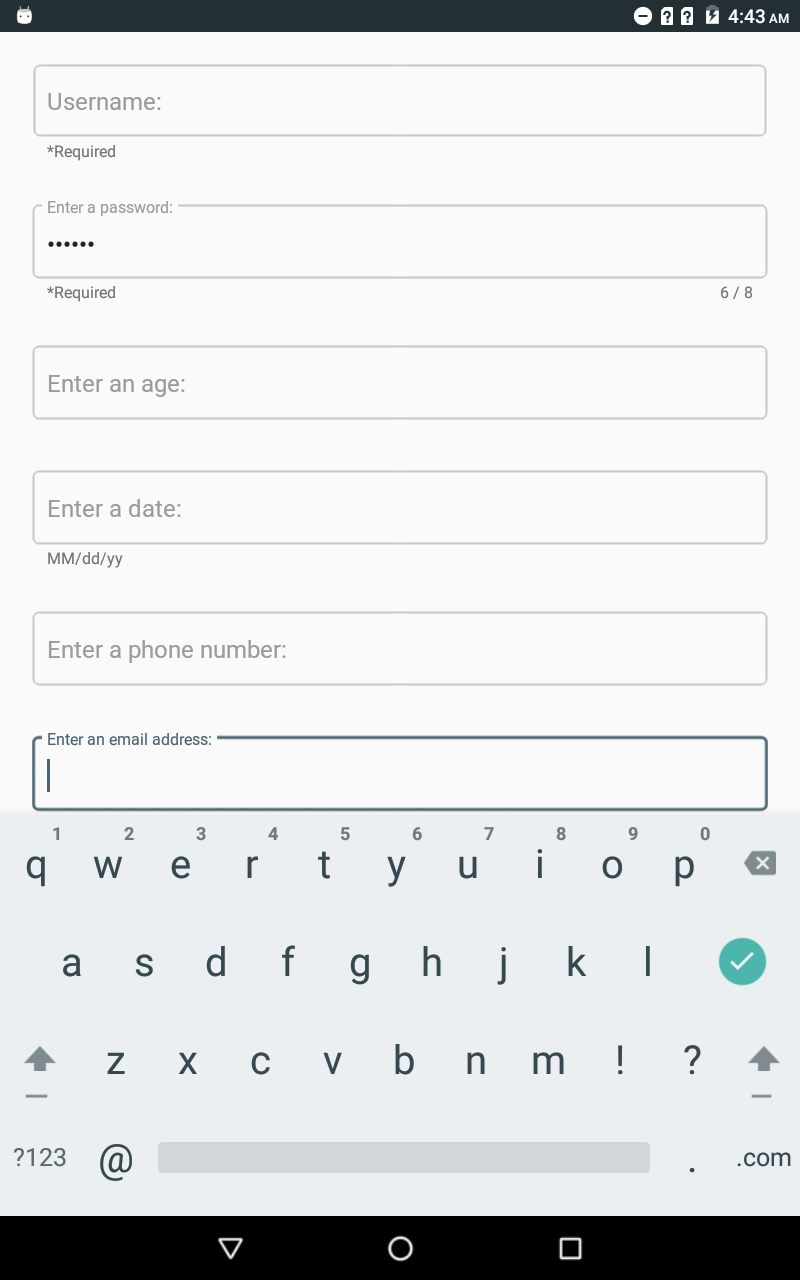
android:imeOptions="actionPrevious"

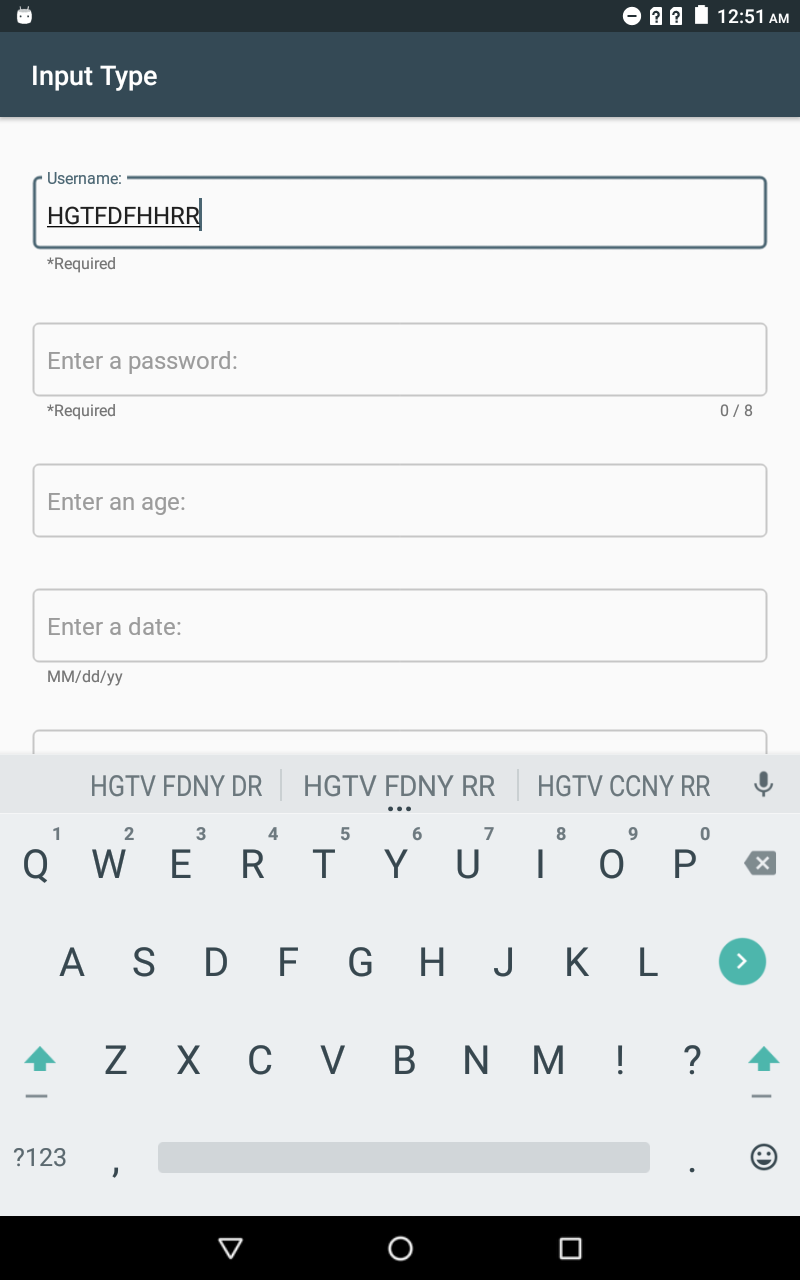
 

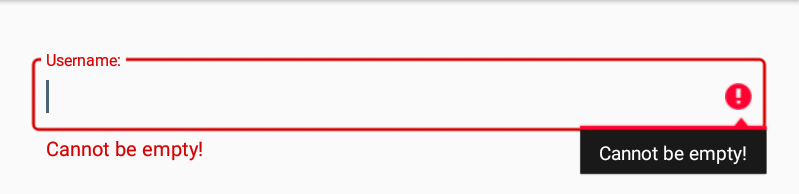
 



android:inputType="text|textCapCharacters"

**Error Text**: error text can be customized with errorTextAppearance style.



**Other attributes related with EditText:**

* ***autoLink***: for email, phone, map or web; controls whether links such as URLs and email addresses are automatically found and converted to clickable links.
* ***lineSpacingExtra***: extra spacing between lines
* ***maxLines***: defines the maximum number of lines. When used on an edit text, **inputType** attribute’s value must be combined with the textMultiLine flag.
* ***shadowDx***: horizontal offset of the text shadow
* ***shadowDy***: vertical offset of the text shadow
* ***tooltipText***: defines text displayed in a small popup window on hover or long press
* ***ems***: font size
* ***allowUndo***: whether undo should be allowed. Default is true

**Requirements for Implementation**

To use Text Field of Material Design, add the following requirement to the build.gradle file in the project:

dependencies {  
 implementation 'com.android.support: design:28.0.0-alpha1 }

In XML file, edit text view is called as EditText. If the Material Design’s text field is preferred, android.support.design.widget.TextInputEditText should be used.

Also, to use all the features of Material Text Fields, you should implement TextInputEditText under the **TextInputLayout**. In XML file, this layout is called as android.support.design.widget.TextInputLayout.

**Areas of Usage**

Text fields are one of the most important interaction method with users. Text fields allows users to send inputs like emails, messages or search their requests. Also, they are useful for sign in and sign up in apps.

**Material Text Field**

Combination of ***textInputLayout*** and ***textInputEditText*** makes easier to apply material text field. *textInputLayout* is parent of the *textInputEditText*. Instead of textInputEditText, you can use EditText as the child of textInputLayout but with textInputEditText, you get more control over the visual aspects of the input text. However, for EditText, you can add material styles such as *Widget.MaterialComponents.TextInputLayout.FilledBox*, attributes that belong to the textInputLayout, also work on EditText. And when you use EditText inside of the textInputLayout, the hint text does not disappear.

\*\*\*Only one text can be placed inside of the textInputLayout (It does not matter whether EditText or textInputEditText).

**Types**

* ***Filled Box*:** (default) that type has a solid background. It is used to draw more attention. To use in your text fields, you should apply FilledBox style. To change the background color of field, use *boxBackgroundColor*.
* ***Outline Box*:** that type has stroked border, its background is transparent. It is used for less emphasized situation. The stroke’s color and width can change with *boxStrokeColor* and *boxStrokeWidth*. To use this, apply the OutlineBox style.

***Height Variations***

There are two height options for both filled and outline box text fields which are standard and dense. Both filled, and outline’s default is standard. Dense option is reduced the height of the text field. It reduces height with reducing vertical padding within the text. You can apply the dense option with appropriate styles according to your text fields’ type.

**Styles**

* Base.Widget.MaterialComponents.TextInputLayout
* Widget.Design.TextInputLayout
* Widget.MaterialComponents.TextInputLayout.FilledBox
* Widget.MaterialComponents.TextInputLayout.FilledBox.Dense
* Widget.MaterialComponents.TextInputLayout.OutlineBox
* Widget.MaterialComponents.TextInputLayout.OutlineBox.Dense

\*While a custom style was creating, you must extend one of those existing styles as parent.

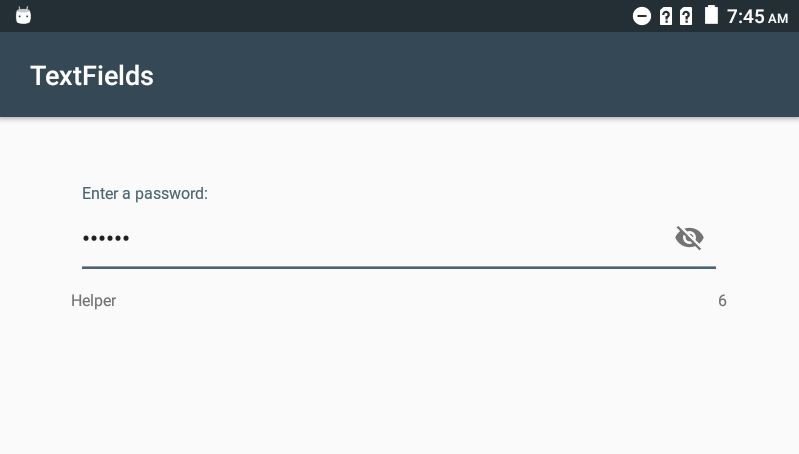
**Widget.Design.TextInputLayout**

Its parent is android:Widget (it customizes only text appearance). Its customized attributes are:

* ***boxBackgroundMode***: the options are filled, outline or none (which are the types that are explained above)
* ***hintTextAppearance***: change the appearance of hint with another style (TextAppearance.Design.Hint (default)). It changes size and color of text.
* ***errorTextAppearance***: change the appearance of error text with another style (TextAppearance.Design.Error (default))
* ***counterTextAppearance***: change the appearance of character counter with another style (TextAppearance.Design.Counter (default))
* ***counterOverflowTextAppearance***: change the appearance of character counter when number of character cross the limit with another style (TextAppearance.Design.Counter.Overflow (default))

\*\*\***password toggle**: it sets the appearance of password. In default, it uses an icon which is the visibility (look like an eye). When the icon is clicked it displays the input. And when it is clicked again, it hides. While it hides the input, it uses the visibility off (the eye with the diagonal line) but while it displays the input, it uses the visibility.

* ***passwordToggleContentDescription***: a string can be set as content description
* ***passwordToggleDrawable***: in default, the icon is visibility but with this attribute, the icon can be changed (when hide the input, no diagonal line over the icon).
* ***passwordToggleTint***: change the color of icon.



<android.support.design.widget.TextInputLayout

……

style="@style/Widget.Design.TextInputLayout"

app:helperText="@string/helper"

app:counterEnabled="true"

android:hint="@string/password"

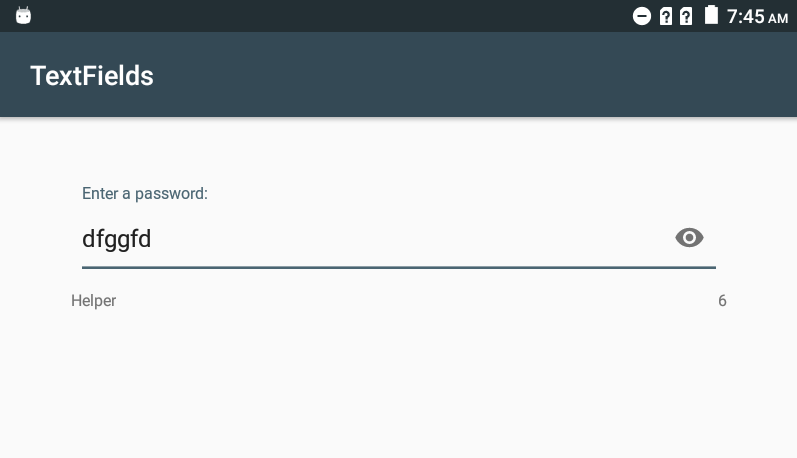
app:passwordToggleEnabled="true">

<android.support.design.widget.TextInputEditText

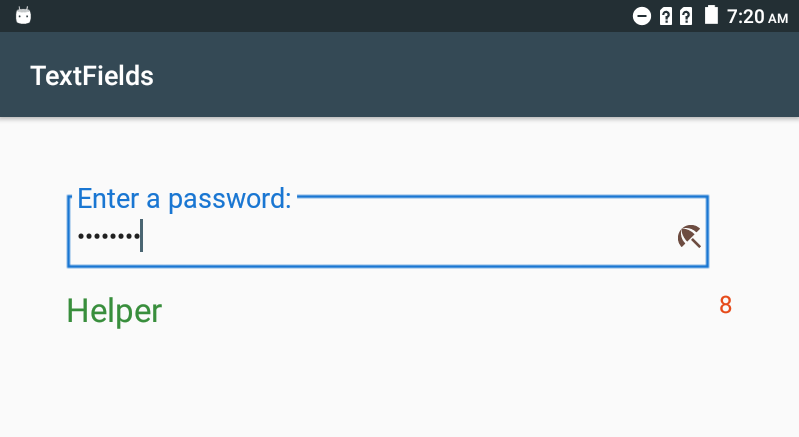
………

android:inputType="textPassword"/>

</android.support.design.widget.TextInputLayout>



Custom Design:



<style name="designTextField" parent="Widget.Design.TextInputLayout">

<item name="boxBackgroundMode">outline</item>

<item name="errorTextAppearance">@style/error</item>

<item name="counterOverflowTextAppearance">@style/counterOverflow</item>

<item name="helperTextTextAppearance">@style/helper</item>

<item name="counterTextAppearance">@style/counter</item>

<item name="passwordToggleContentDescription">@string/password</item>

<item name="hintTextAppearance">@style/hint</item>

<item name="passwordToggleDrawable">@drawable/ic\_beach\_access\_black\_24dp</item>

<item name="passwordToggleTint"> #6D4C41</item>

</style>

<style name="hint" parent="TextAppearance.Design.Hint">

<item name="android:textSize">20sp</item>

<item name="android:textColor">#1976D2</item>

</style>

<style name="counterOverflow" parent="TextAppearance.Design.Counter.Overflow">

<item name="android:textColor"> #FFFF00</item>

<item name="android:textSize">30sp</item>

</style>

<style name="counter" parent="TextAppearance.Design.Counter">

<item name="android:textSize">18sp</item>

<item name="android:textColor"> #E64A19</item>

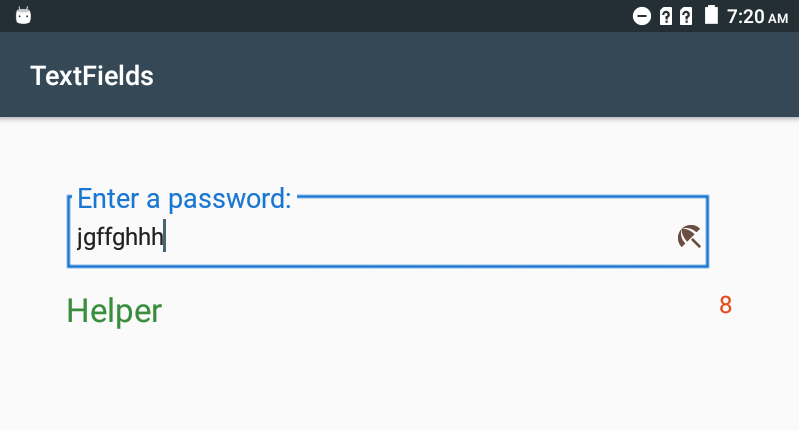
</style>

<style name="error" parent="TextAppearance.Design.Error">

<item name="android:textColor"> #D50000</item>

<item name="android:textSize">15sp</item>

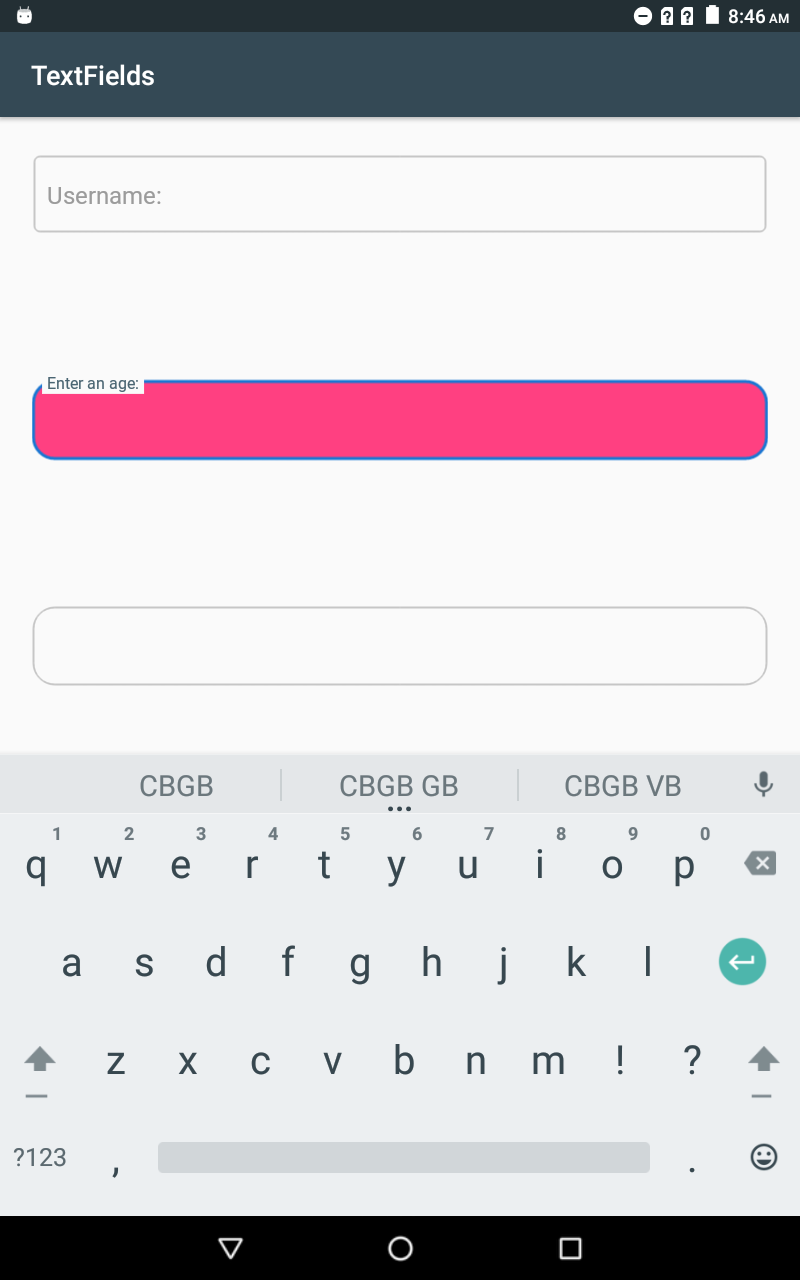
</style>



**Base.Widget.MaterialComponents.TextInputLayout**

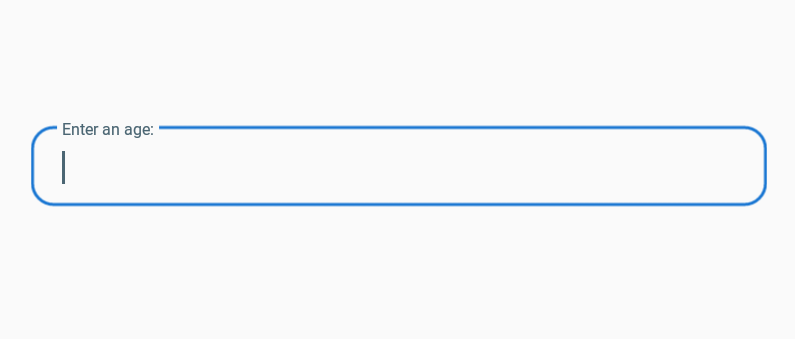
Its parent is Widget.Design.TextInputLayout. Its special attributes are:

* ***boxBackgroundMode***: the options are filled, outline or none (which are the types that are explained above) the default is outline.
* ***boxBackgroundColor***: change the color of container. The default is null because of the outline type.
* ***boxCornerRadiusTopLeft***: set the roundness of the field. Default: 4dp
* ***boxCornerRadiusTopRight***: set the roundness of the field. Default: 4dp
* ***boxCornerRadiusBottomRight***: set the roundness of the field. Default: 4dp
* ***boxCornerRadiusBottomLeft***: set the roundness of the field. Default: 4dp
* ***boxStrokeColor***: change the color of stroke.
* ***boxPaddingLeft***: moves the input +hint to the left. Default: 12dp
* ***boxPaddingRight***: moves the input + hint to the right. Default: 12dp
* ***boxCollapsedPaddingTop***: it moves the input to the upwards. Default: 0dp for outline, 12dp for filled
* ***boxCollapsedPaddingBottom***: it moves the input to the downwards. Default: 20dp for outline, 12dp for filled
* ***boxExpandedPaddingTop***: moves the hint to the upward. Default: 16dp for outline, 20dp for filled
* ***boxExpandedPaddingBottom***: moves the hint to the downward. Default: 16dp for outline, 20dp for filled



<item name=”boxBackgroundColor”> #FF4081</item>

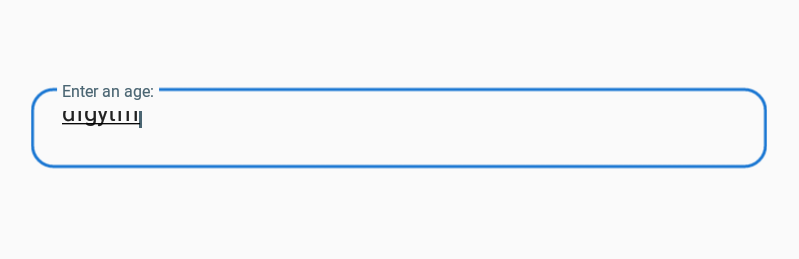
When you change the background color of the outline text field, the input is not visible. This is useful for filled text field.



<item name=”boxStrokeColor”> #1976D2</item>

<item name=”boxPaddingLeft”> 24dp</item>

<item name=”boxPaddingRİght”>24dp</item>



<item name=”boxCollapsedPaddingBottom”> 60dp</item>



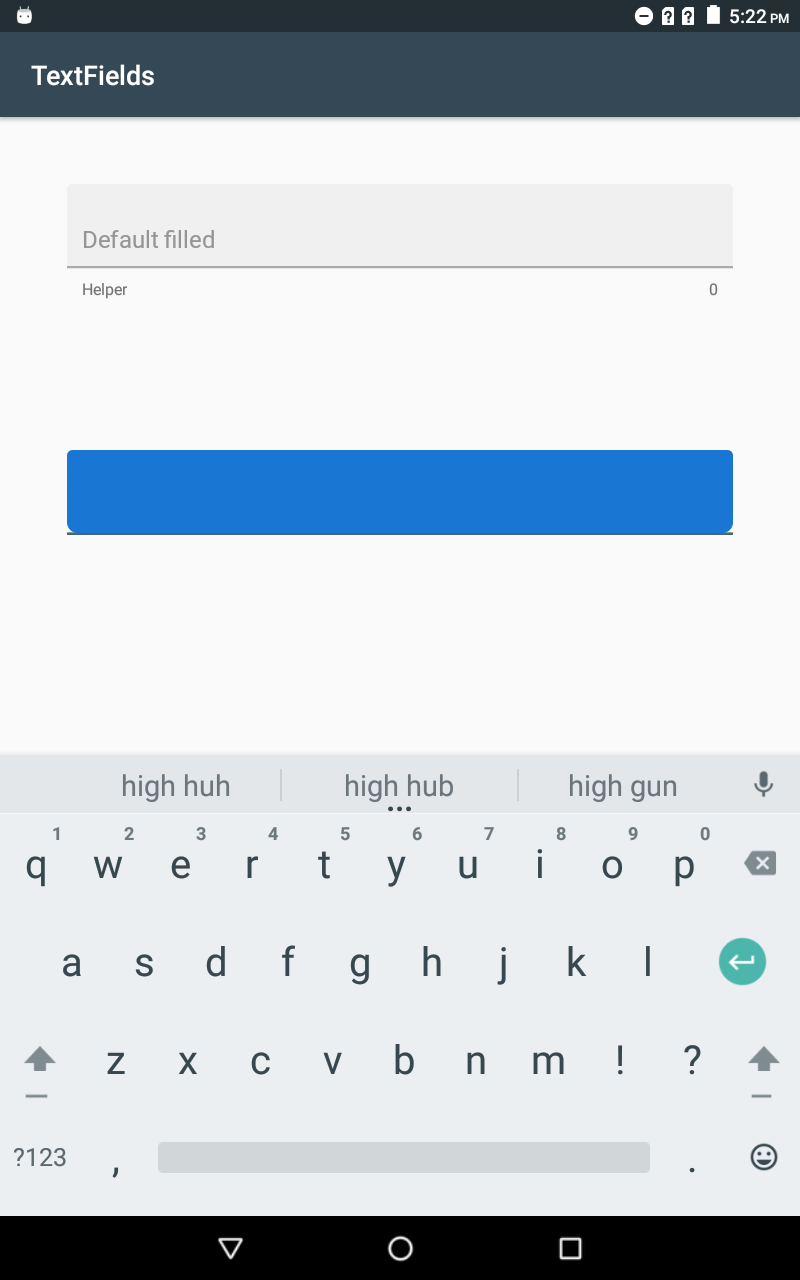
<item name=”boxExpandedPaddingBottom”> 32dp</item>

**Widget.MaterialComponents.TextInputLayout.FilledBox**

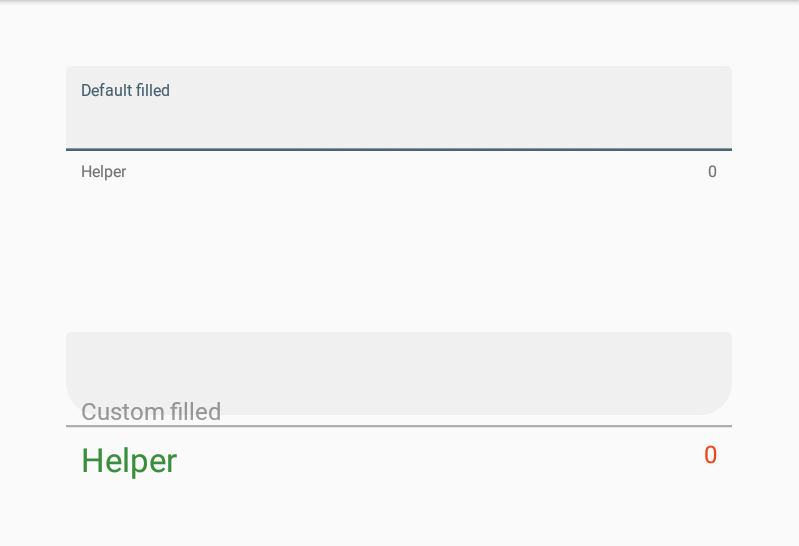
Its parent is Base.Widget.MaterialComponents.TextInputLayout. It customizes: (These all are the same with above attributes.)

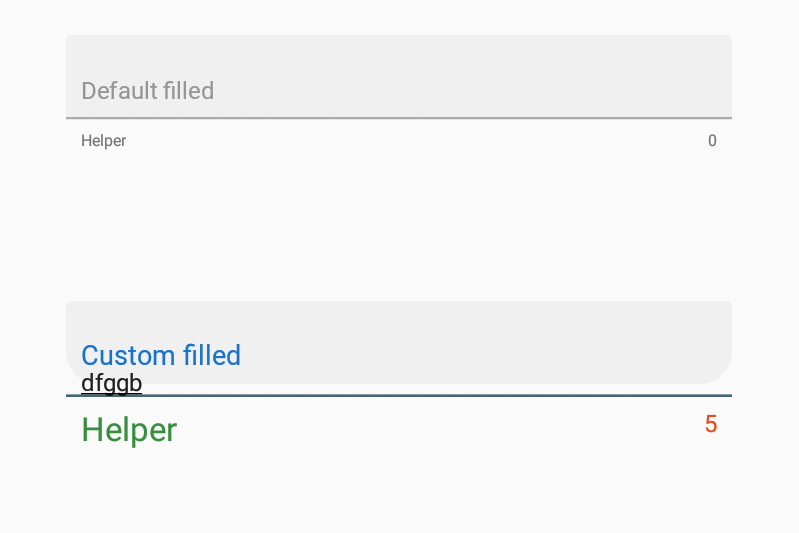
* ***boxBackgroundMode***: filled is chosen.
* ***boxBackgroundColor***: #0A000000.
* ***boxCornerRadiusBottomLeft***: 0dp
* ***boxCornerRadiusBottomRight***: 0dp
* ***boxCollapsedPaddingTop***: 12dp
* ***boxCollapsedPaddingBottom***: 20dp
* ***boxExpandedPaddingTop***: 20dp
* ***boxExpandedPaddingBottom***: 16dp

The appearance of helper, hint, error and counter text can be customized.



When the background color is changed, the writings cannot be visible.





<style name="FilledBoxTextField" parent="Widget.MaterialComponents.TextInputLayout.FilledBox">

<item name="boxCornerRadiusBottomLeft">24dp</item>

<item name="boxCornerRadiusBottomRight">24dp</item>

<item name="boxCollapsedPaddingTop">30dp</item>

<item name="boxCollapsedPaddingBottom">25dp</item>

<item name="boxExpandedPaddingTop">28dp</item>

<item name="boxExpandedPaddingBottom">5dp</item>

<item name="helperTextTextAppearance">@style/helper</item>

<item name="hintTextAppearance">@style/hint</item>

<item name="counterTextAppearance">@style/counter</item>

</style>

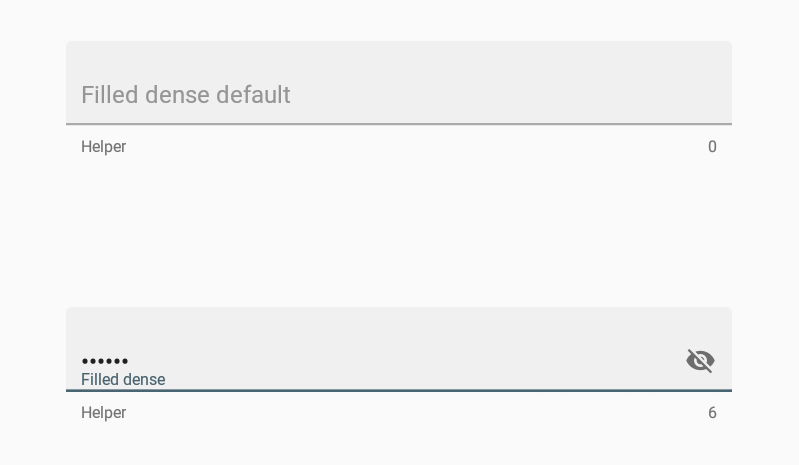
**Widget.MaterialComponents.TextInputLayout.FilledBox.Dense**

It customizes ***boxCollapsedTop***: 16dp, ***boxCollapsedBottom***: 16dp, ***boxExpandedPaddingTop***: 16dp and ***boxExpandedPaddingBottom***: 16dp.

\*\*\*The other material text fields attributes cannot be used in this style.



Default filled box dense

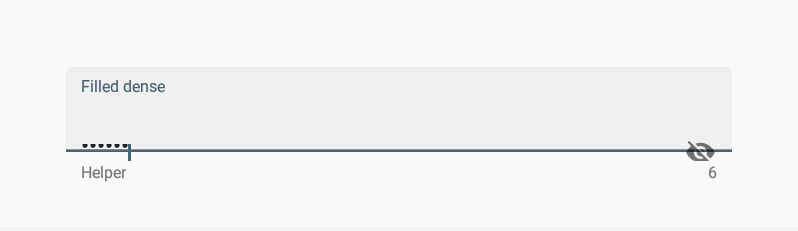


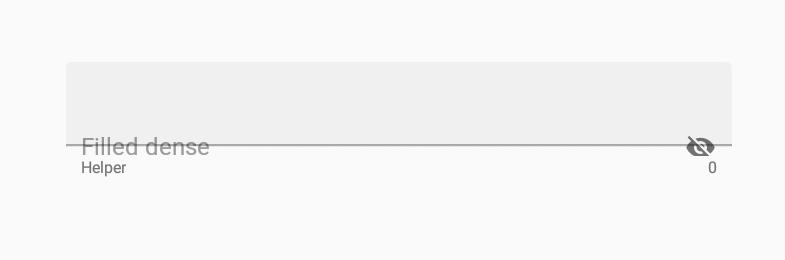
<style name="FilledBoxDenseTextField" parent="Widget.MaterialComponents.TextInputLayout.FilledBox.Dense">

<item name="boxCollapsedPaddingTop">48dp</item>

<item name="boxCollapsedPaddingBottom">0dp</item>

</style>





<style name="FilledDenseText" parent="Widget.MaterialComponents.TextInputLayout.FilledBox.Dense">

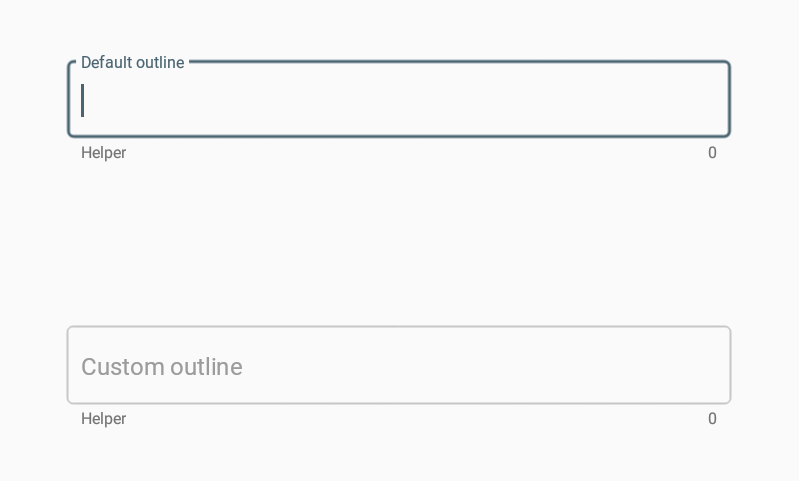
<item name="boxExpandedPaddingTop">48dp</item>

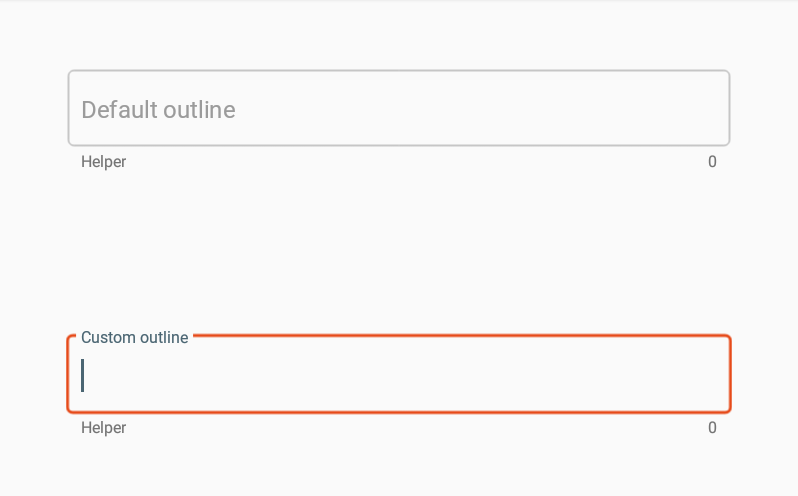
<item name="boxExpandedPaddingBottom">0dp</item>

</style>

**Widget.MaterialComponents.TextInputLayout.OutlineBox**

Its parent is *Base.Widget.MaterialComponents.TextInputLayout*. It does not customize any attributes, but all material text field attributes can be customized.





Custom outline:

<style name="OutlineBoxTextField" parent="Widget.MaterialComponents.TextInputLayout.OutlineBox">

<item name="boxStrokeColor">#E64A19</item>

</style>

**Widget.MaterialComponents.TextInputLayout.OutlineBox.Dense**

It customizes:

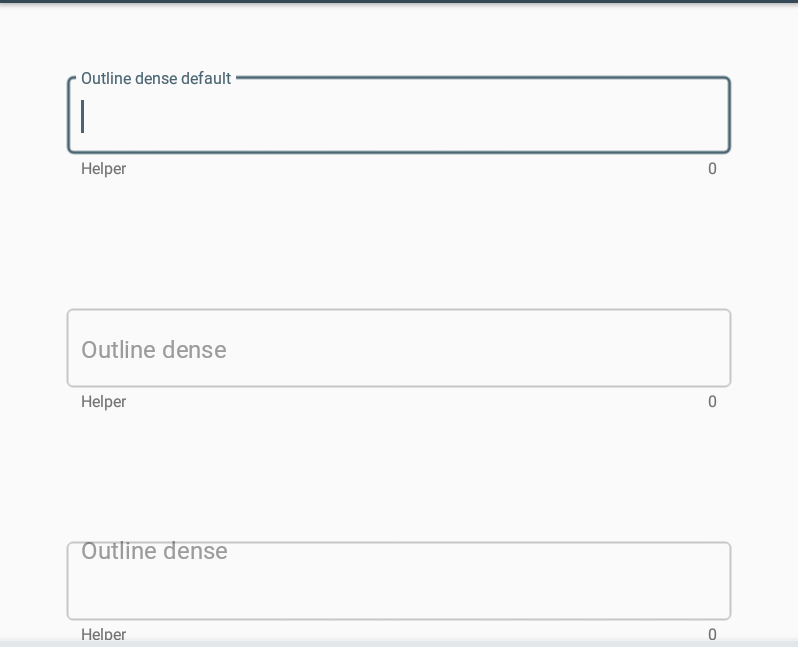
***boxCollapsedTop***: 16dp,

***boxCollapsedBottom***: 16dp,

***boxExpandedPaddingTop***: 16dp and

***boxExpandedPaddingBottom***: 16dp.

\*\*\*The other material text fields attributes cannot be used in this style.



The topmost field is the default outline dense.

Second box is one of the custom outline dense which customizes:

<style name="OutlineBoxDenseTextField" parent="Widget.MaterialComponents.TextInputLayout.OutlineBox.Dense">

<item name="boxCollapsedPaddingBottom">0dp</item>

<item name="boxCollapsedPaddingTop">75dp</item>

</style>

But there is no changing because of the type of the background mode.

The third one is also a custom outline dense box which customizes:

<style name="OutlineBoxDenseTextField2" parent="Widget.MaterialComponents.TextInputLayout.OutlineBox.Dense">

<item name="boxExpandedPaddingBottom">48dp</item>

<item name="boxExpandedPaddingTop">0dp</item>

</style>



**References**

Homepage. (n.d.). Retrieved June 27, 2018, from <https://material.io/>

Android Developers. (n.d.). Retrieved June 27, 2018, from https://developer.android.com/