3.7. Диалоговые окна

Сайт: Samsung Innovation Campus
Курс: Мобильная разработка на Kotlin
Книга: 3.7. Диалоговые окна

Напечатано:: Murad Rezvan
Дата: понедельник, 3 июня 2024, 17:50

Оглавление

Простые диалоговые окна

Очень часто в графическом интерфейсе появляются диалговые окна, как информационные, с единственной кнопкой "ОК", так и такие, где пользователь что-то выбирает и его выбор учитывается в ходе программы. В Android это можно реализовать при помощи подклассов `DialogFragment`.

Если в `onCreate()` активности написать

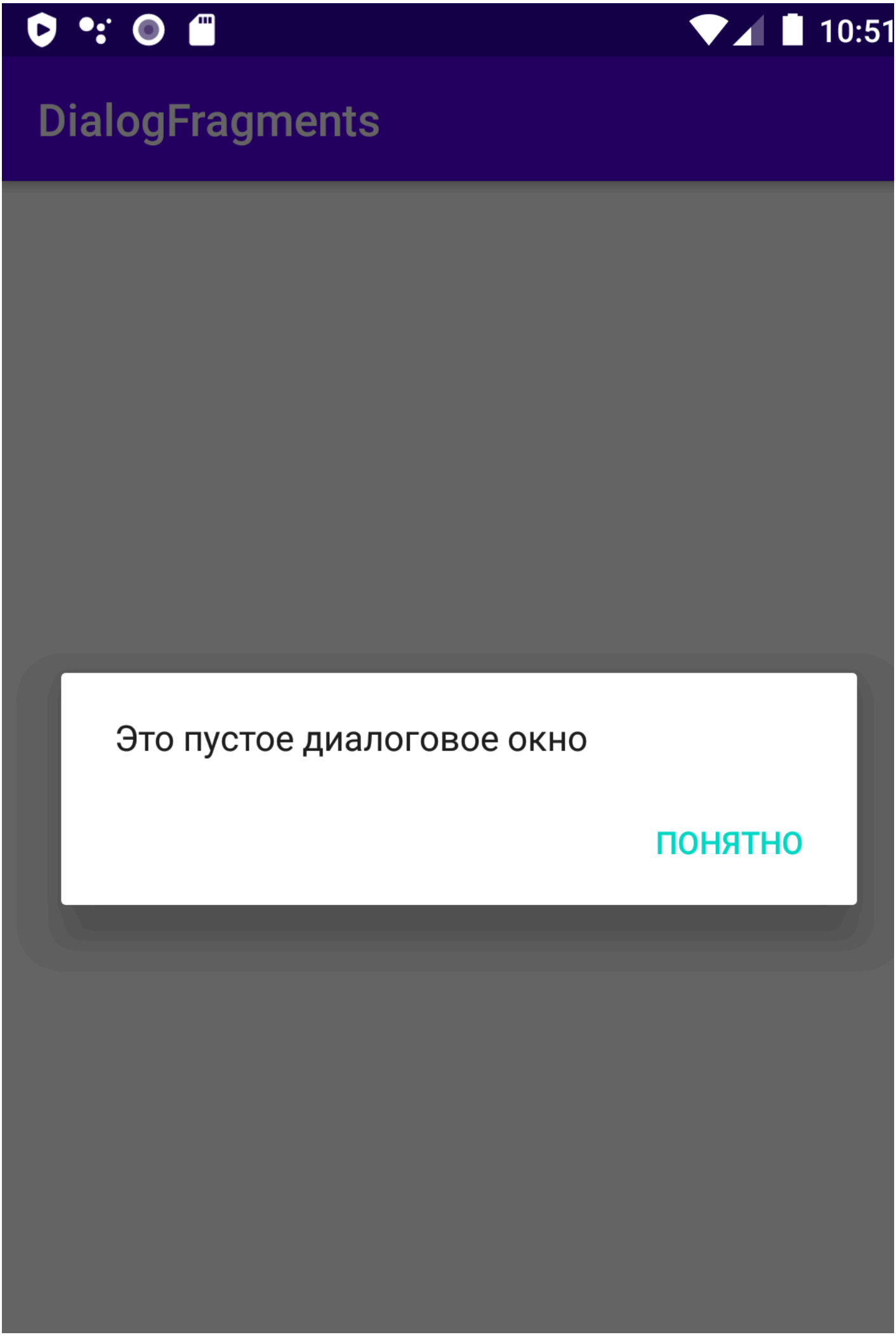
```
DialogFragment().show(supportFragmentManager, "EmptyDialog")
```

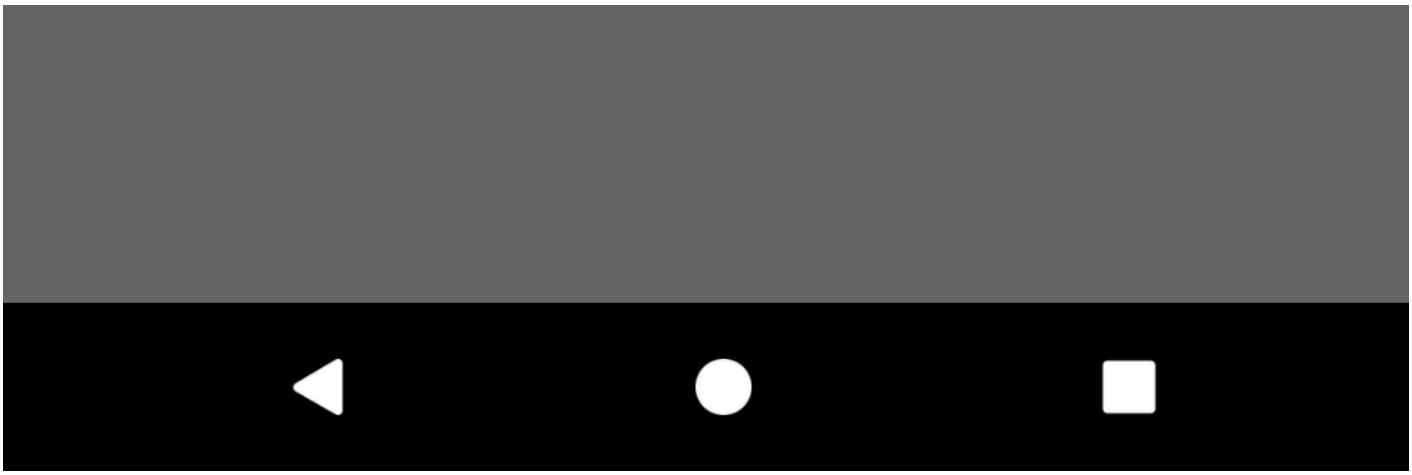
Экран просто потемнеет. Диалог будет реально пустым.

Для того, чтобы отобразить окно, нужно пронаследоваться от `DialogFragment` и в нем переопределить функцию `onCreateDialog()`.

```
class MyAlertDialog: DialogFragment(){
    override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        return activity?.let {
          AlertDialog.Builder(it)
                .setMessage("Это пустое диалоговое окно")
                .setPositiveButton("Понятно", null)
                .create()
        } ?: throw IllegalStateException("Activity cannot be null")
    }
}
```

Здесь для построения диалога используется используется объект класса `Builder`, строитель Alert-диалогов. Можно настраивать много различных свойств)(см. https://developer.android.com/reference/kotlin/android/app/AlertDialog.Builder. В коде выше мы используем только два, устанавливаем текст сообщения и действие при нажатии на кнопку - закрыть диалог.


10:51
DialogFragments
Это пустое диалоговое окно
ПОНЯТНО



Кнопки "Yes", "No", "Cancel"

Например, можно добавить стандартные кнопки "No" и "Cancel" при помощи функций строителя диалогов `setNegativeButton()`, `.setNeutralButton()`. Все они буду закрывать окно. Как различить, что выбрал пользователь?

Вторым параметром в эти функции передается слушатель. Например, можно сделать слушателем саму активность.

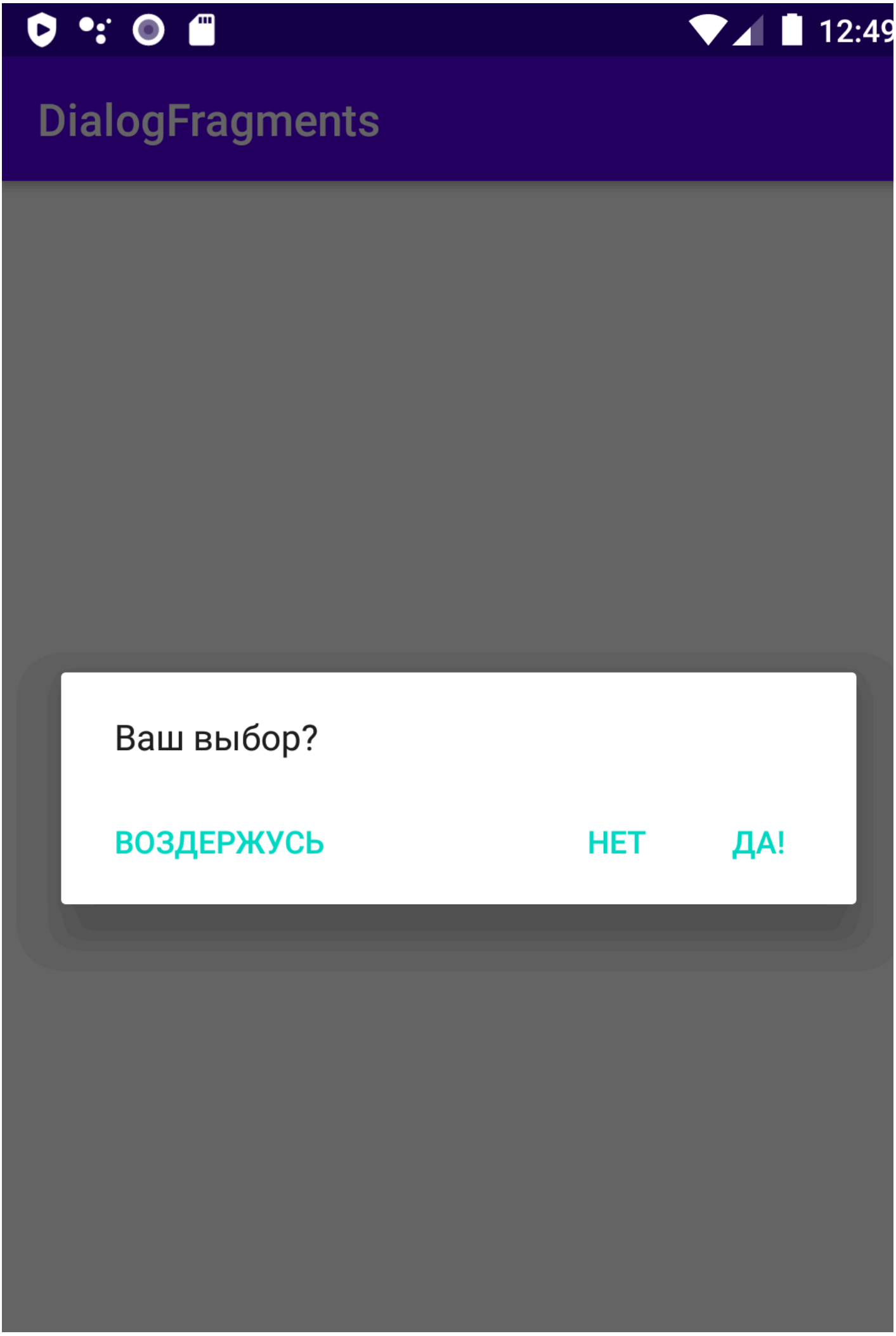
Добавим вызов диалогового окна по единственному `TextView` стандартного приложения и пропишем реакцию на выбор пользователя.

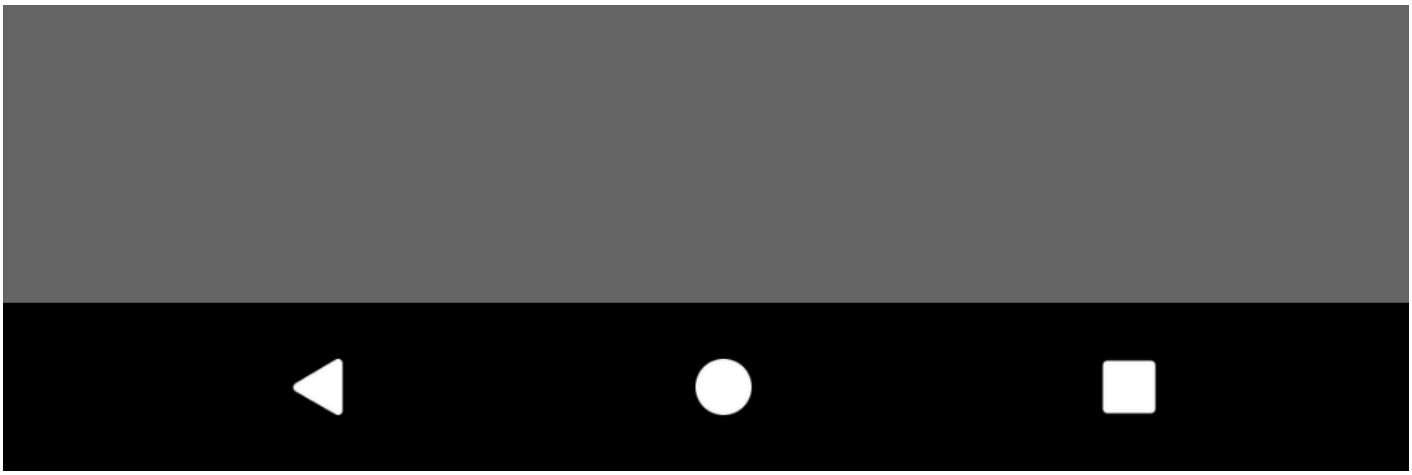
В активности (допустим мы добавили в XML к текстовому полю id значение `textView`):

```
class MainActivity : AppCompatActivity(), DialogInterface.OnClickListener{

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        textView.setOnClickListener {
  MyAlertDialog().show(supportFragmentManager, "YesNoCancelDialog")
        }
  }

    override fun onClick(dialog: DialogInterface?, which: Int) {
        when (which){
            DialogInterface.BUTTON_POSITIVE -> textView.setText("Вы согласились!");
            DialogInterface.BUTTON_NEGATIVE -> textView.setText("Вы отказались!");
            DialogInterface.BUTTON_NEUTRAL -> textView.setText("Выбор не сделан");
        }
    }
}
```

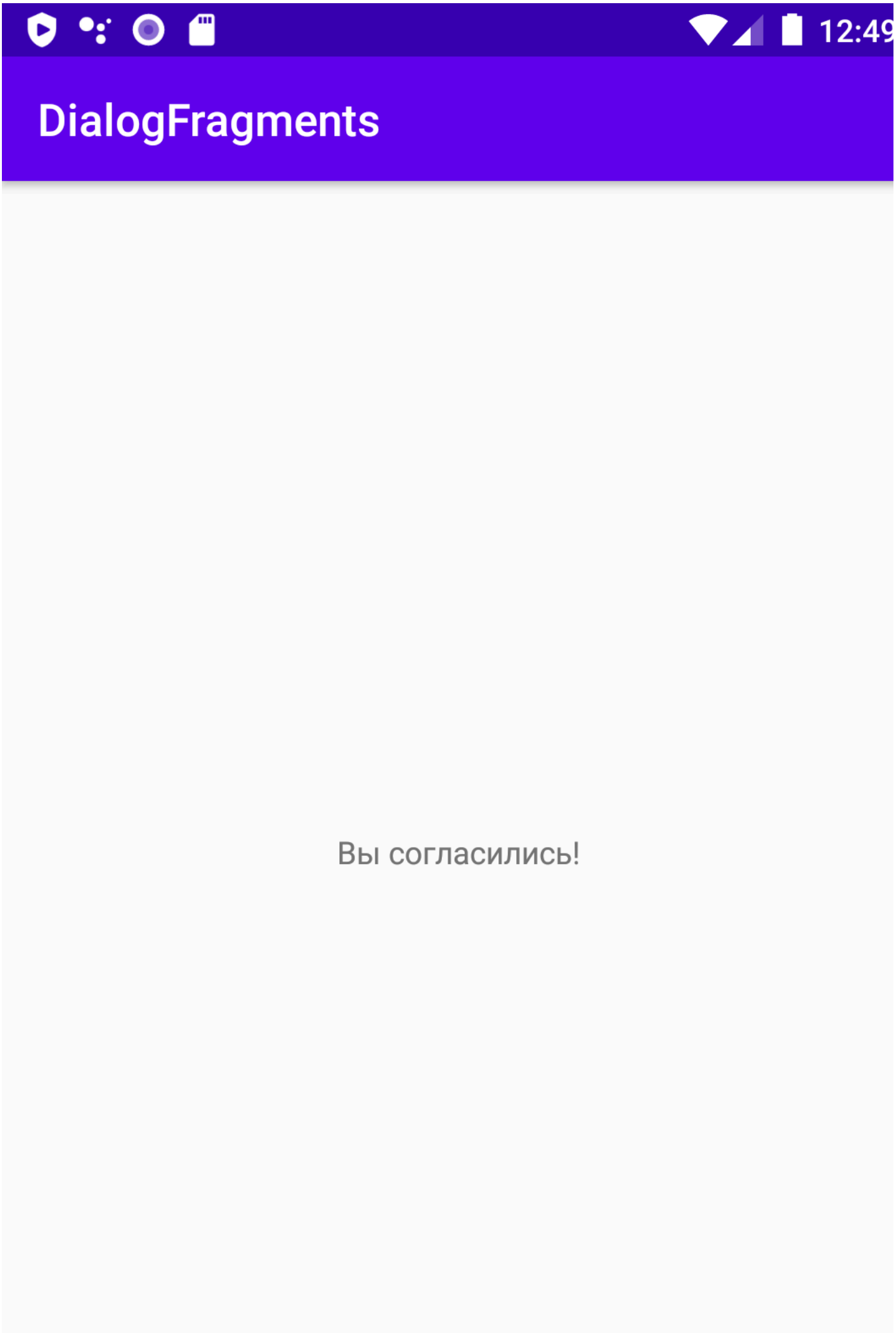
В классе диалога

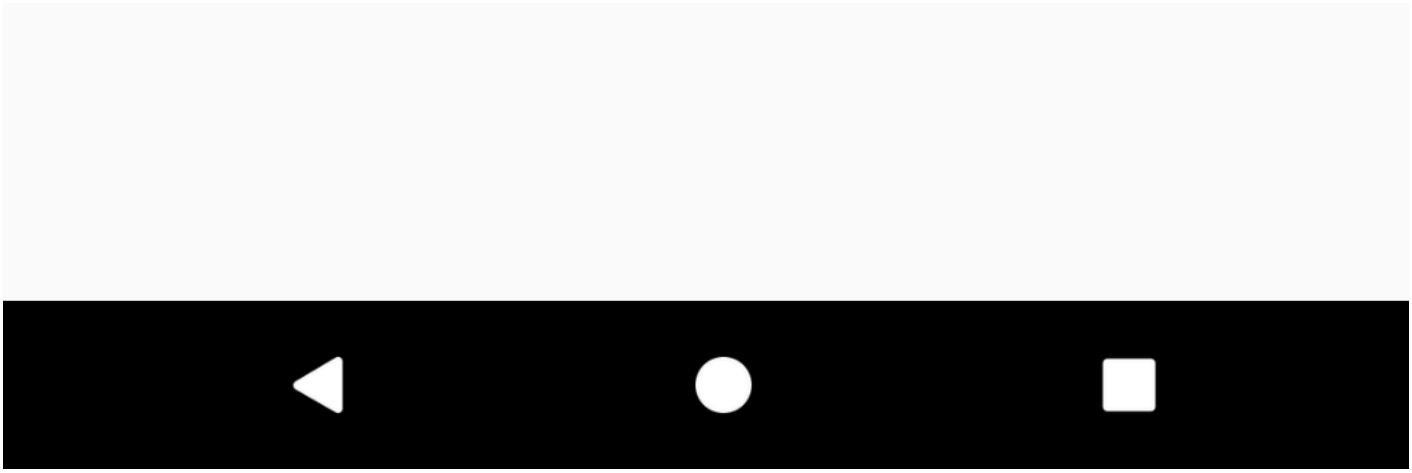
```
 class MyAlertDialog: DialogFragment(){
    override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        return activity?.let {
  AlertDialog.Builder(it)
                .setMessage("Ваш выбор?")
                .setPositiveButton("Да!", activity as DialogInterface.OnClickListener)
                .setNegativeButton("Нет", activity as DialogInterface.OnClickListener)
                .setNeutralButton("Воздержусь", activity as DialogInterface.OnClickListener)
                .create()
        } ?: throw IllegalStateException("Activity cannot be null")
    }
}
```


12:49
DialogFragments
Ваш выбор?
ВОЗДЕРЖУСЬ
НЕТ
ДА!



При нажатии в `TextView` отображается наш выбор


12:49
DialogFragments
Вы согласились!



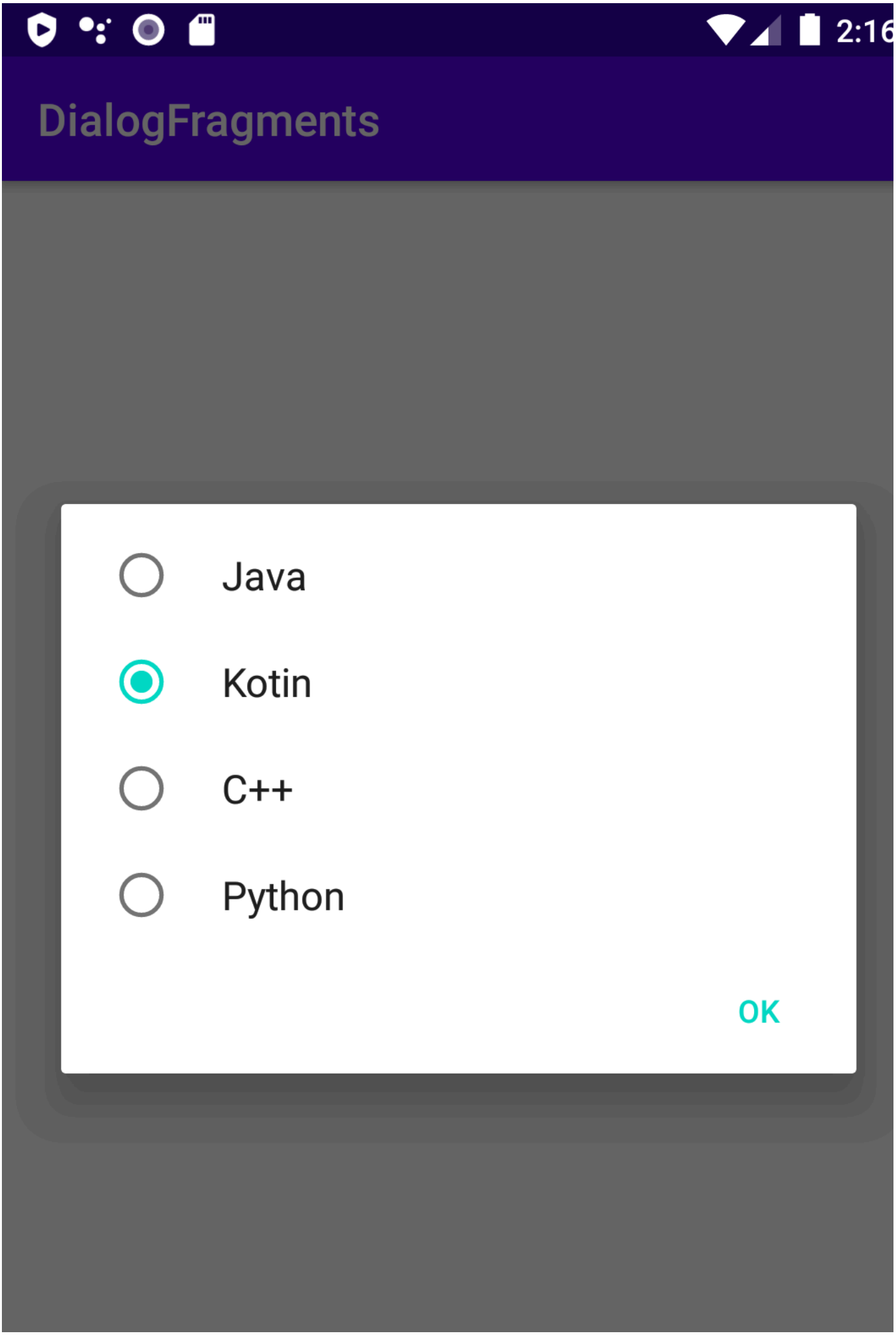
Диалоговые окна с переключателями и флажками

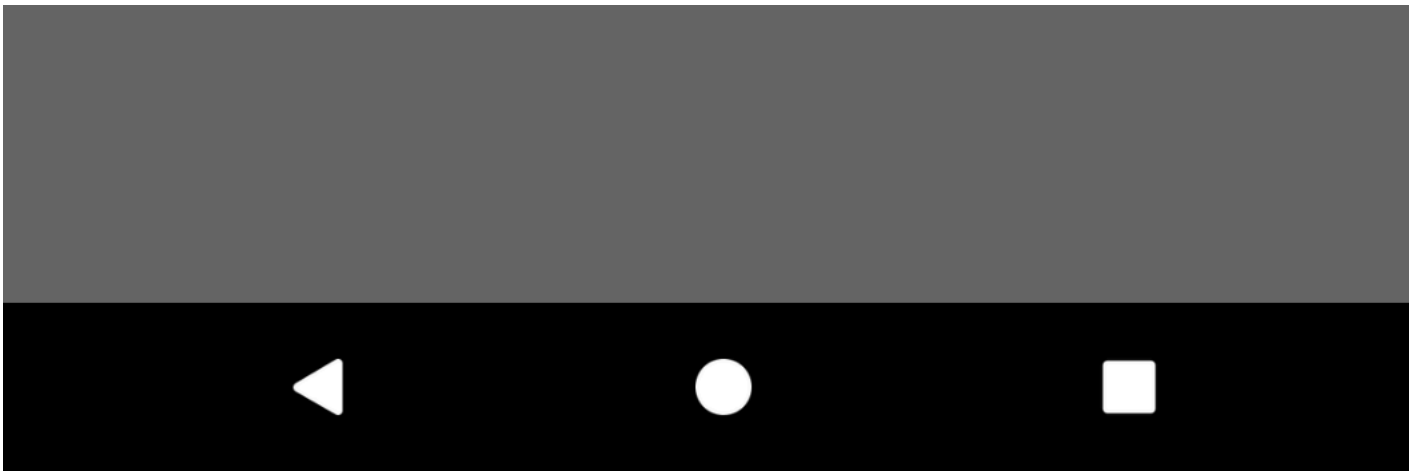
Радиокнопки

Строитель диалогов позволяет добавить в диалог стандартные переключатели-радиокнопки помощи функции `setSingleChoiceItems()`, которой можно передать список надписей на кнопках, индекс изначально выбранной кнопки (-1 для того, чтобы ничего не было выбрано в начале) и слушателя.

В примере используем анонимного слушателя, который будет выводить `Toast` с выбором *после закрытия окна*. Выбор пользователя будет сохраняться в переменную, а потом при закрытии выводится тост.

```
class MyAlertDialog: DialogFragment(){
    val langs = arrayOf("Java", "Kotin", "C++", "Python")
    var choice = 0
  override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        return activity?.let {
           AlertDialog.Builder(it)
                .setSingleChoiceItems(langs, 1, {dialog, which -> choice = which})
                .setPositiveButton("Ok")
                    {dialog, which ->
                        Toast.makeText(activity, langs[choice] + " отличный выбор!",
                                Toast.LENGTH_LONG).show()}
            .create()
        } ?: throw IllegalStateException("Activity cannot be null")
    }
}
```


2:16
DialogFragments
Java
Kotin
C++
Python
OK



После выбора диалоговое окно не закрывается, поэтому обычно добавляют кнопку "Ok", хотя можно обойтись и без этого, закрывая окно в функции слушателя при помощи функции `dismiss()` диалога, который слушатель получает в параметрах.

Диалог, созданный таким образом

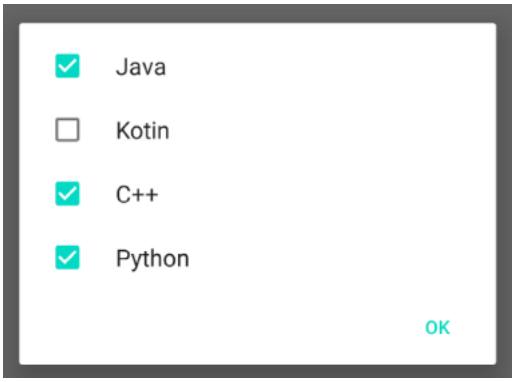
```
  AlertDialog.Builder(it)
    .setSingleChoiceItems(langs, 1,
        {dialog, which ->
  Toast.makeText(activity, langs[which] + " отличный выбор!",
            Toast.LENGTH_LONG).show()
            dialog.dismiss()
        }
  )
    .create()
```

Закрывается сразу после выбора.

Флажки

Аналогично можно использовать функцию `setMultiChoiceItem()`, которая принимает еще и boolean-массив установленных флажков .

```
class MyAlertDialog: DialogFragment(){
    val langs = arrayOf("Java", "Kotin", "C++", "Python")
    val checked= booleanArrayOf(true, true, false, false)
  override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        return activity?.let {
  AlertDialog.Builder(it)
                .setMultiChoiceItems(langs, checked){
  dialog, which, isChecked ->
  checked[which] = isChecked
                    }
  .setPositiveButton("Ok",
                    {dialog, which ->
  Toast.makeText(activity, "Выбрано языков: "+ checked.count({it}) ,
                                  Toast.LENGTH_LONG).show()}
  )
                .create()
        } ?: throw IllegalStateException("Activity cannot be null")
    }
}
```



Лямбда функция, которая передается функции `setMultiChoiceItem()`, принимает три параметра: диалог, индекс нажатой кнопки и установлена ли она, потому что в отличие от радиокнопок флажки нажатием можно не только устанавливать, но и снимать.

Выбор времени

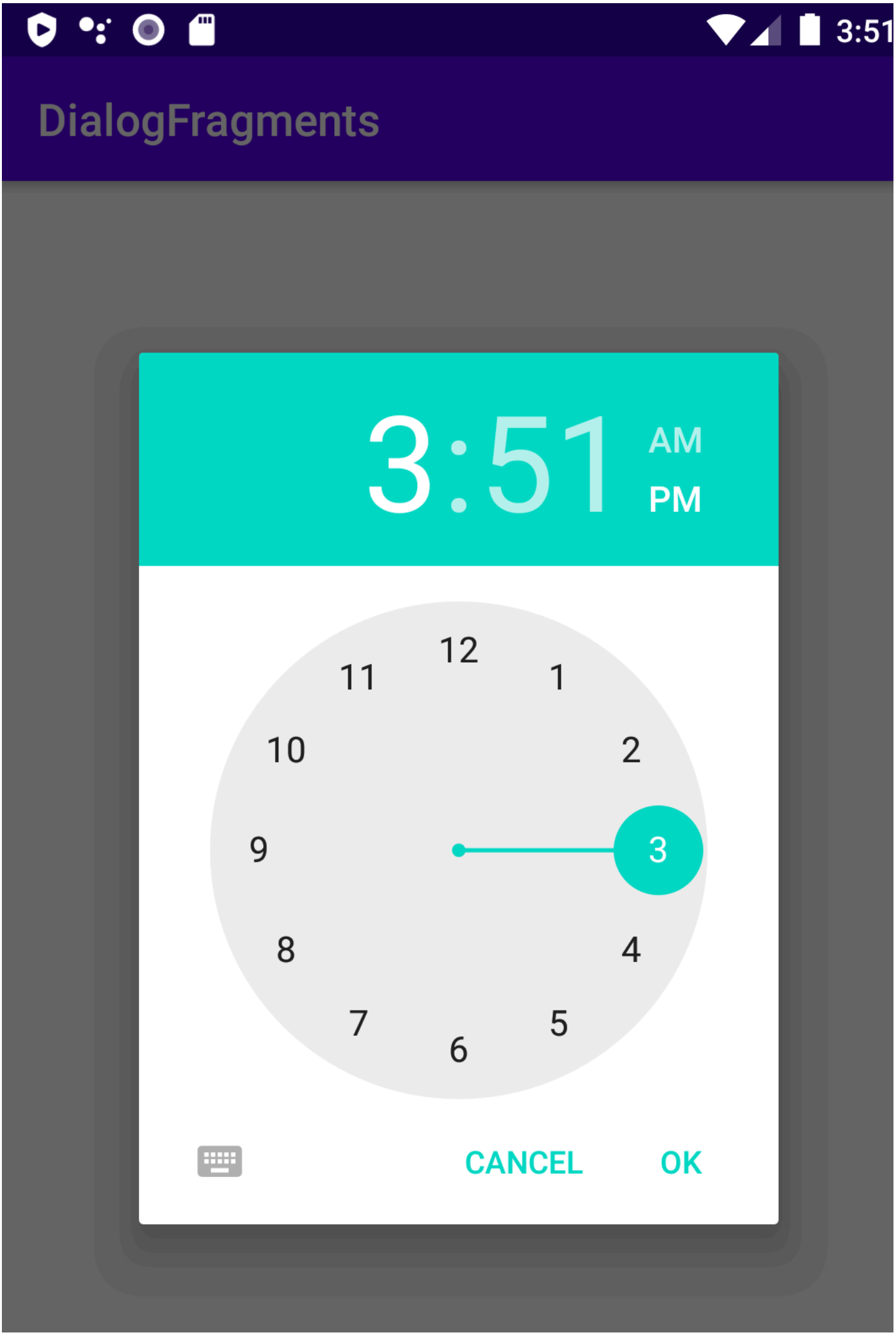
Очень часто в приложении пользователь должен установить время.

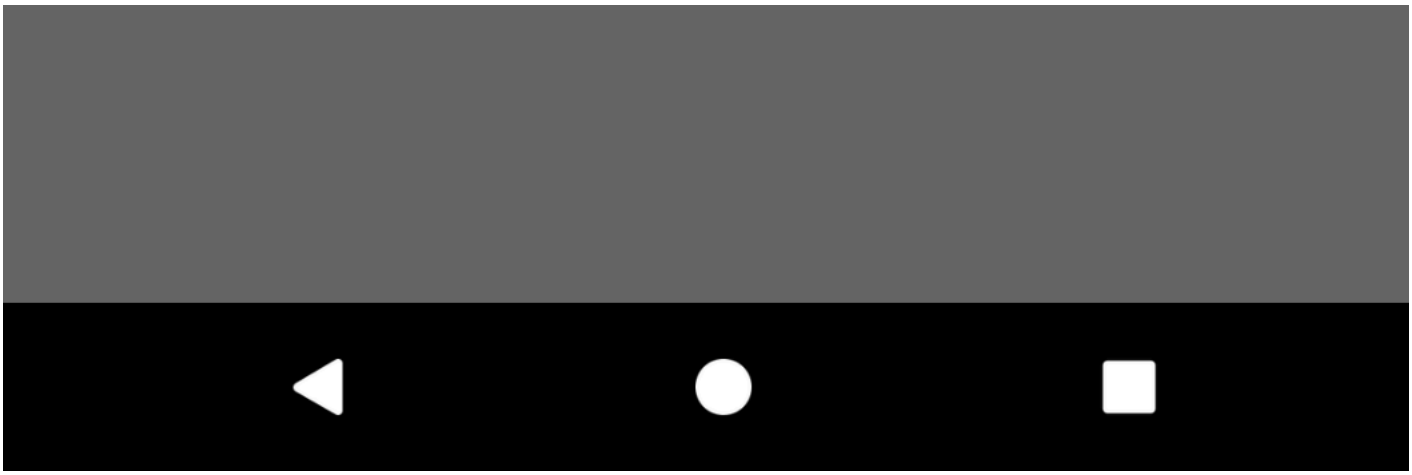
Для этого есть специальный диалоговый класс `TimePickerDialog`. Достаточно передать ему при создании начальное время формат (24 и 12 часов) и слушателя.

Создадим демонстрационный пример, который будет устанавливать на нем текущее время и после выбора выводить установленное время в тост.

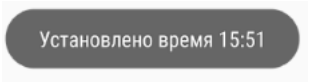
Текст получается очень простым:

```
class MyTimePickerDialog: DialogFragment(){
    override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {

        val c = Calendar.getInstance();
        val curHour = c.get(Calendar.HOUR_OF_DAY);
        val curMinute = c.get(Calendar.MINUTE);
        return TimePickerDialog(activity, {
             _, hour, minute ->
                  Toast.makeText(activity,
                        "Установлено время " + hour + ":" + minute, Toast.LENGTH_LONG).show()
        }, curHour, curMinute, false)
    }
}
```


3:51
DialogFragments
3:51
AM
PM
12
1
2
3
4
5
6
7
8
9
10
11
CANCEL
OK



После вывода времени на экране появляется тост



В практической работе вам предстоит разработать более содержательный пример.

Начать тур для пользователя на этой странице