Диалоговые окна

В Android 3.0 (API 11) появилась новинка - класс **android.app.DialogFragment** и его аналог **android.support.v4.app.DialogFragment**, а чуть позже и **android.support.v7.app.AppCompatDialogFragment** из библиотеки совместимости, позволяющие выводить диалоговое окно поверх своей активности. Это рекомендуемый стандарт для вывода диалоговых окон в новых проектах. Раньше использовался класс **Dialog** и его производные, например, **AlertDialog**. Они никуда не делись, только теперь их нужно встраивать в фрагмент, который выступает в качестве контейнера. Поэтому условно

окнами.

Использование фрагментов для диалоговых окон в силу своей архитектуры является удобным вариантом в приложениях, который лучше справляется с поворотами устройства, нажатием кнопки Назад, лучше масштабируется под разные экраны и т.д.

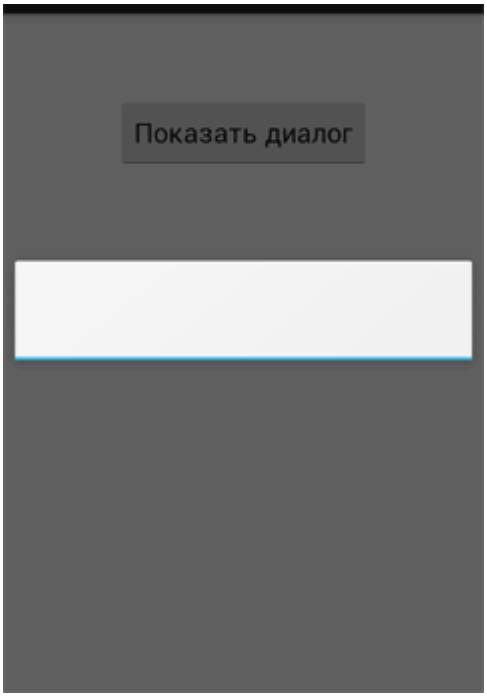
Если вам будут попадаться старые примеры, то займитесь их переделкой. Это не сложно. Тем более, что среда разработки будет всячески ругаться на использование устаревших классов и методов.

Для создания диалога следует наследоваться от класса **AppCompatDialogFragment**. Создадим новый класс **MyDialogFragment**:

```
package ru.alexanderklimov.dialog;

import android.support.v7.app.AppCompatDialogFragment;

public class MyDialogFragment extends AppCompatDialogFragment {
}
```

Допустим у нас есть кнопка на экране активности. Вызвать диалоговое окно можно через метод **show()**.

```
public void onClick(View view) {
    FragmentManager manager = getSupportFragmentManager();
    MyDialogFragment myDialogFragment = new MyDialogFragment();
    myDialogFragment.show(manager, "dialog");
}
```



Скорее всего вы увидите пустой прямоугольник или квадрат. А возможно у вас просто потемнеет экран активности.

Так как это обычный фрагмент, то нам нужно позвать менеджера фрагментов и попросить его показать фрагмент.

Для вызова диалога мы создаём экземпляр класса **MyDialogFragment** и вызываем метод **show()**. Метод принимает два параметра: объект класса **FragmentManager**, получаемый через метод **getFragmentManager()**, и тег - идентификатор диалога в виде строковой константы, по которому можно идентифицировать диалоговое окно, если их будет много в нашем проекте.

Существует и альтернативный вариант показа окна через транзакцию.

```
public void onClick(View view) {
    MyDialogFragment myDialogFragment = new MyDialogFragment();
    FragmentManager manager = getSupportFragmentManager();
    //myDialogFragment.show(manager, "dialog");

    FragmentTransaction transaction = manager.beginTransaction();
    myDialogFragment.show(transaction, "dialog");
}
```

Мы получили пустой бесполезный фрагмент. Следует заняться его конструированием. В созданном классе нужно переопределить метод **onCreateDialog()**. Если используется разметка, то также используется метод **onCreateView()**, как и у обычных фрагментов.

AlertDialog

Самый распространённый вариант диалогового окна - это **AlertDialog**. С него и начнём.

Диалоговое окно **AlertDialog** является расширением класса **Dialog**, и это наиболее используемое диалоговое окно в практике программиста. Очень часто требуется показать диалог с кнопками **Да** и **Нет**. В создаваемых диалоговых окнах можно задавать следующие элементы:

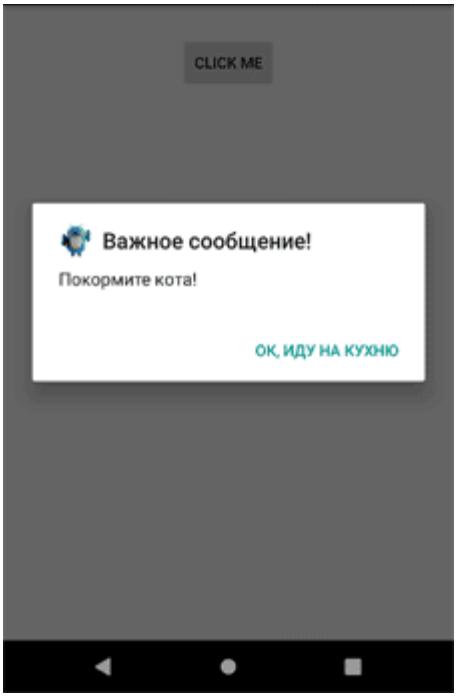
- заголовок
- текстовое сообщение
- кнопки: от одной до трёх
- список
- флажки
- переключатели

Используйте **android.support.v7.app.AlertDialog** при работе с стандартными проектами студии, хотя есть и другие варианты.

Диалог с одной кнопкой

Начнём с простого примера - покажем на экране диалоговое окно с одной кнопкой.

```
@NonNull
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {
    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    builder.setTitle("Важное сообщение!")
            .setMessage("Покормите кота!")
            .setIcon(R.drawable.ic_launcher_cat)
            .setPositiveButton("OK, иду на кухню", new
DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    // Закрываем окно
                    dialog.cancel();
                }
            });
    return builder.create();
}
```



Сначала мы создаём объект класса **AlertDialog.Builder**, передав в качестве параметра ссылку на активность. Затем, используя методы класса **Builder**, задаём для создаваемого диалога заголовок (метод **setTitle()**), текстовое сообщение в теле диалога (метод **setMessage()**), значок (метод **setIcon()**), а также кнопку через метод под странным названием **setPositiveButton()**.

Сама обработка нажатия кнопки внутри диалогового окна задаётся внутри метода **setPositiveButton()**. В нашём случае мы просто закрываем окно диалога через метод **cancel()**.

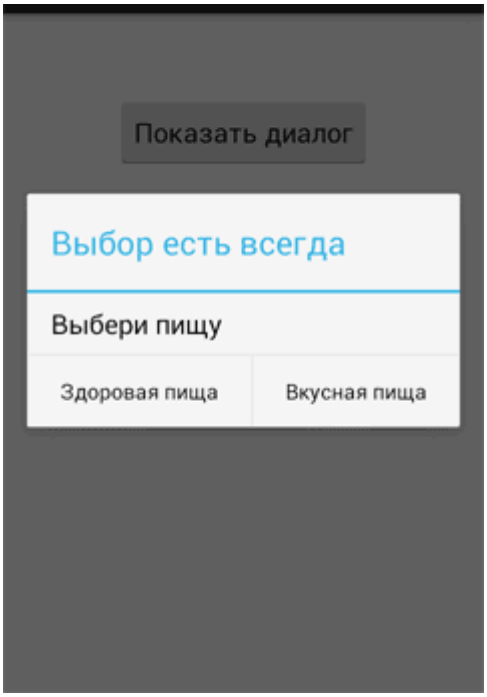
Обратите внимание на не совсем обычный способ вызова череды методов цепочкой через точку **.setMessage("Покормите кота!").setIcon(R.drawable.ic_android_cat)** и т.д. Такой синтаксис можно часто увидеть в jQuery. При таком способе не нужно использовать точку с запятой в конце каждого метода, вы просто склеиваете все вызовы. Но можете использовать и обычный синтаксис.

Если вы используете Java 8, то студия предложит использовать лямбда-выражение вместо анонимного класса. Решайте сами, в каком стиле писать код.

Нелёгкий выбор - пример с двумя кнопками

Теперь рассмотрим пример создания диалогового окна с двумя кнопками на основе иллюстрации.

```
import android.app.Dialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.DialogFragment;
import android.support.v7.app.AlertDialog;
import android.widget.Toast;

public class MyDialogFragment extends DialogFragment {
    @NonNull
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        String title = "Выбор есть всегда";
        String message = "Выбери пищу";
        String button1String = "Вкусная пища";
        String button2String = "Здоровая пища";

        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        builder.setTitle(title);  // заголовок
        builder.setMessage(message); // сообщение
        builder.setPositiveButton(button1String, new DialogInterface.OnClickListener()
{
            public void onClick(DialogInterface dialog, int id) {
                Toast.makeText(getActivity(), "Вы сделали правильный выбор",
                        Toast.LENGTH_LONG).show();
            }
        });
        builder.setNegativeButton(button2String, new DialogInterface.OnClickListener()
{
            public void onClick(DialogInterface dialog, int id) {
                Toast.makeText(getActivity(), "Возможно вы правы", Toast.LENGTH_LONG)
                        .show();
            }
        });
        builder.setCancelable(true);

        return builder.create();
    }
}
```

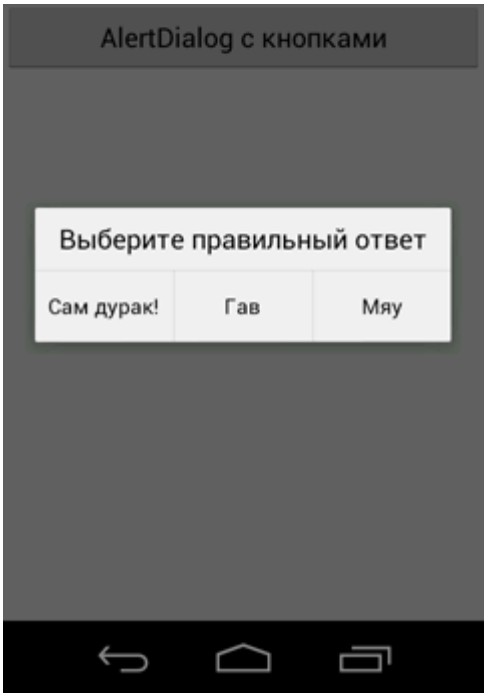


Общая часть кода осталась прежней - объект класса **AlertDialog.Builder**, методы для настройки окна, а также кнопки диалога и обработку событий на них. В **AlertDialog** можно добавить только по одной кнопке каждого типа: **Positive**, **Neutral** и **Negative**, т. е. максимально возможное количество кнопок в диалоге — три. На названия кнопок не обращайте внимания, они не несут смысловой нагрузки, а только определяют порядок вывода. Причём в разных версиях Android порядок менялся. Поэтому на старых устройствах кнопка "Да" может быть первой, а на новых - последней. Для каждой кнопки используется один из методов с префиксом **set...Button**, которые принимают в качестве параметров надпись для кнопки и интерфейс **DialogInterface.OnClickListener**, определяющий действие при нажатии. Чтобы пользователь не мог закрыть диалог нажатием в любой точке экрана, вызывается метод **setCancelable()** с значением *true*.

Три кнопки

Рассмотрим пример с тремя кнопками. Разницы практически нет. Повторяем все предыдущие шаги, для отображения диалогового окна вызывается метод **builder.create()**. Например, для создания диалога с кнопками *Мяу, Гав, Сам дурак!* код будет выглядеть приблизительно так:

```
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {

    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    builder.setMessage("Выберите правильный ответ")
            .setCancelable(true)
            .setPositiveButton("Мяу",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                            int id) {
                            dialog.cancel();
                        }
                    })
            .setNeutralButton("Гав",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                            int id) {
                            dialog.cancel();
                        }
                    })
            .setNegativeButton("Сам дурак!",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog,
                                            int id) {
                            dialog.cancel();
                        }
                    });

    return builder.create();
}
```

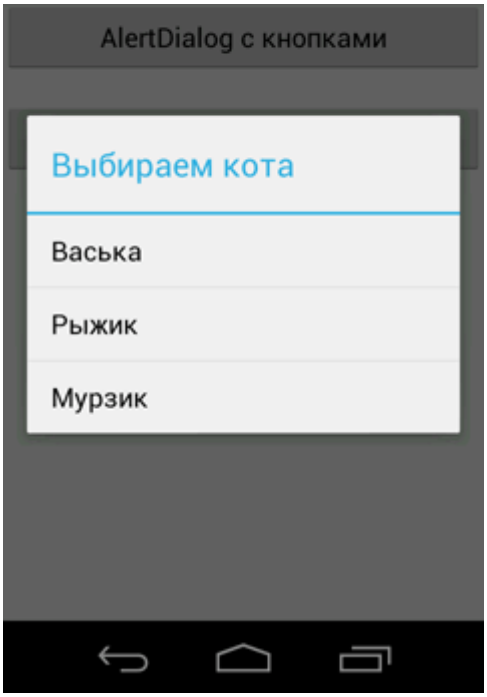


AlertDialog со списком

Если вам нужно диалоговое окно со списком выбираемых пунктов вместо кнопок, то используйте метод **setItems()**, где нужно указать массив данных для отображения в списке диалога. Данный метод нельзя использовать вместе с методом **setMessage()**, так они выводят содержимое в основной части окна.

```
@NonNull
@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {

    final String[] catNamesArray = {"Васька", "Рыжик", "Мурзик"};

    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    builder.setTitle("Выберите кота")
            .setItems(catNamesArray, new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int which) {
                    Toast.makeText(getActivity(),
                            "Выбранный кот: " + catNamesArray[which],
                            Toast.LENGTH_SHORT).show();
                }
            });

    return builder.create();
}
```

Выбранный элемент содержится в параметре **which**. При выборе одного из пунктов меню появится всплывающее уведомление, показывающее выбранного кота.

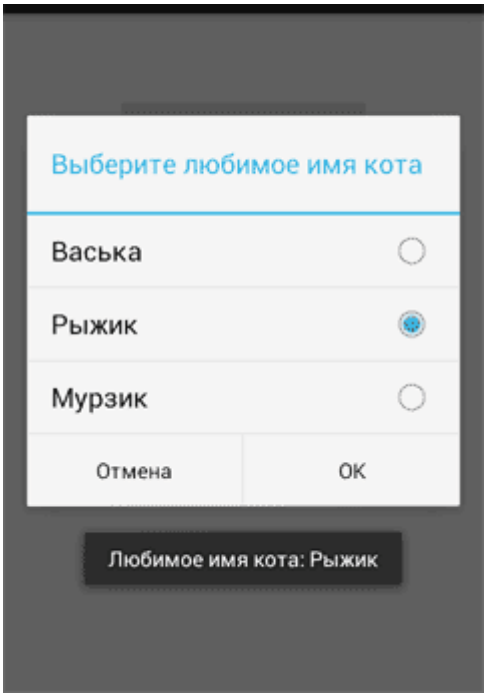


AlertDialog с переключателями

Для создания диалогового окна с переключателями применяется метод **setSingleChoiceitems()** вместо метода **setItems()**. Если диалоговое окно создается внутри **onCreateDialog()**, система Android управляет состоянием списка с переключателями. Пока текущая деятельность активна, диалоговое окно при последующих вызовах запоминает ранее выбранные пункты.

```
public Dialog onCreateDialog(Bundle savedInstanceState) {

    final String[] catNamesArray = {"Васька", "Рыжик", "Мурзик"};

    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    builder.setTitle("Выберите любимое имя кота")
            // добавляем переключатели
            .setSingleChoiceItems(catNamesArray, -1,
                    new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog,
                                            int item) {
                            Toast.makeText(
                                    getActivity(),
                                    "Любимое имя кота: "
                                            + catNamesArray[item],
                                    Toast.LENGTH_SHORT).show();
                        }
                    })
            .setPositiveButton("OK", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int id) {
                    // User clicked OK, so save the mSelectedItems results somewhere
                    // or return them to the component that opened the dialog

                }
            })
            .setNegativeButton("Отмена", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int id) {

                }
            });

    return builder.create();
}
```

Обратите внимание на следующие детали. При выборе переключателя диалоговое окно закрываться не будет. Поэтому необходимо предусмотреть механизм закрытия окна, например, добавить кнопку. Второй момент - в методе **setSingleChoiceitems** для первого параметра используется массив значений для переключателей, а для второго параметра используется целочисленное значение индекса переключателя, который будет включен по умолчанию при вызове диалогового окна. Если вы хотите, чтобы все переключатели при запуске были в выключенном состоянии, то используйте значение -1.



AlertDialog с флажками

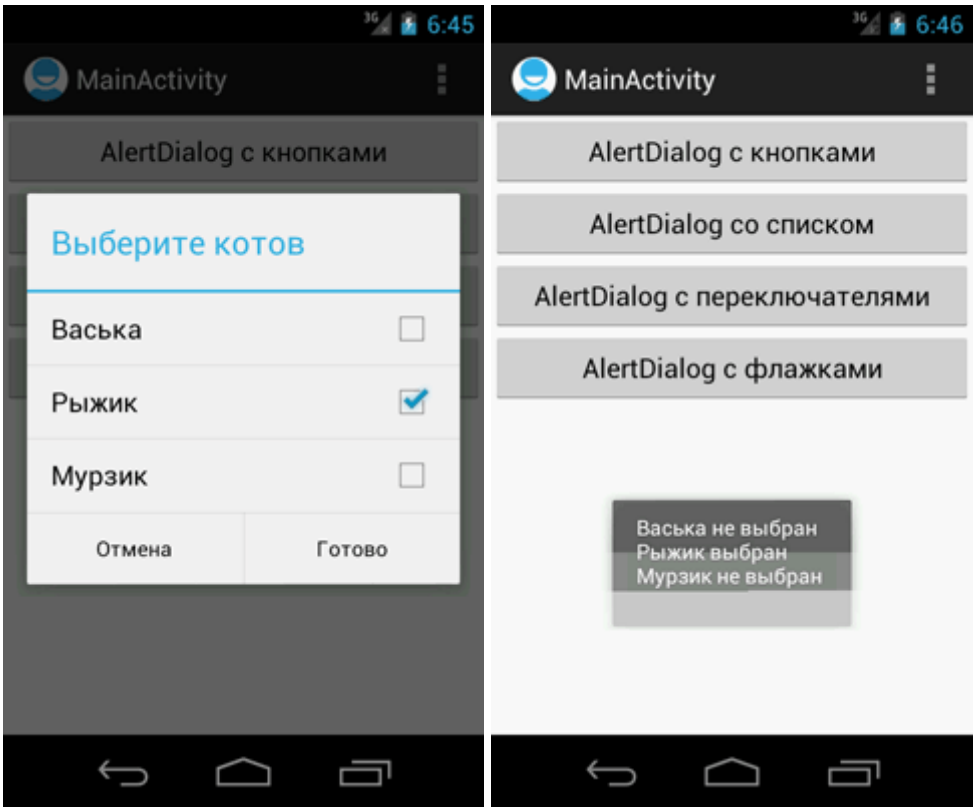
Если вы хотите использовать вместо переключателей флажки (CheckBox) для множественного выбора, то вам нужен метод **setMultiChoiceItems()**. Код практически идентичен предыдущему примеру:

```
public Dialog onCreateDialog(Bundle savedInstanceState) {

    final String[] catNamesArray = {"Васька", "Рыжик", "Мурзик"};
    final boolean[] checkedItemsArray = {false, true, false};

    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    builder.setTitle("Выберите котов")
            .setMultiChoiceItems(catNamesArray, checkedItemsArray,
                    new DialogInterface.OnMultiChoiceClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog,
                                            int which, boolean isChecked) {
                            checkedItemsArray[which] = isChecked;
                        }
                    })
            .setPositiveButton("Готово",
                    new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog,
                                            int id) {
                            StringBuilder state = new StringBuilder();
                            for (int i = 0; i < catNamesArray.length; i++) {
                                state.append(catNamesArray[i]);
                                if (checkedItemsArray[i])
                                    state.append(" выбран\n");
                                else
                                    state.append(" не выбран\n");
                            }
                            Toast.makeText(getActivity(),
                                    state.toString(), Toast.LENGTH_LONG)
                                    .show();
                        }
                    })

            .setNegativeButton("Отмена",
                    new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog,
                                            int id) {
                            dialog.cancel();
                        }
                    });

    return builder.create();
}
```

при вызове диалога. Например, мы хотим, чтобы второй элемент списка был отмечен флажком, а остальные элементы нужно оставить неотмеченными. В этом случае используем массив из булевых значений:

```
final boolean[] checkedItemsArray = {false, true, false};
```

Как и в предыдущем случае с переключателями, для диалогового окна с флажками необходимо использовать кнопки для закрытия окна.



Передать данные в активность

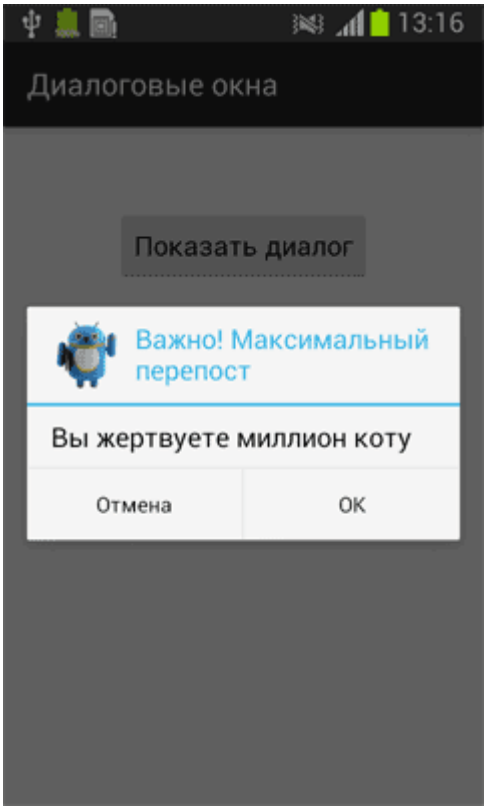
Для обработки щелчков кнопок в диалоговом окне вы пишете код, в котором указываете родительскую активность.

```
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.support.v4.app.DialogFragment;

public class MyDialogFragment extends DialogFragment {
    @NonNull
    @Override
    public Dialog onCreateDialog(Bundle savedInstanceState) {
        AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
        builder.setMessage("Вы жертвуете миллион коту")
                .setIcon(R.drawable.ic_launcher_cat)
                .setTitle("Важно! Максимальный перепост")
                .setPositiveButton("OK", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        ((MainActivity) getActivity()).okClicked();
                    }
                })
                .setNegativeButton("Отмена", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        ((MainActivity) getActivity()).cancelClicked();
                    }
                });

        return builder.create();
    }
}
```

В коде каких-то сложностей нет - устанавливаем заголовок, значок, кнопки. При построении диалогового окна указываем родительскую активность и название методов в ней, которые будут отвечать за обработку нажатий кнопок диалога - в нашем случае это методы **okClicked()** и **cancelClicked()**. Кстати, имена методов будут подчёркнуты красной линией и среда разработки предложит создать данные методы в классе активности (используйте комбинацию клавиш Alt+Enter).

Возвращаемся в код главной активности и пропишем код для нажатий кнопок диалогового окна:

```
            Toast.LENGTH_LONG).show();
}

public void cancelClicked() {
    Toast.makeText(getApplicationContext(), "Вы выбрали кнопку отмены!",
            Toast.LENGTH_LONG).show();
}
```



Пример был написан по принципу - "работает и ладно". На самом деле пример не совсем грамотный, хотя даже в документации он ещё встречается.

Правильный вариант рассматривается во второй части о диалоговых окнах DialogFragment.

AlertDialog с собственной разметкой

Если стандартный вид **AlertDialog** вас не устраивает, то можете придумать свою разметку и подключить её через метод **setView()**

```
public Dialog onCreateDialog(Bundle savedInstanceState) {
    AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
    LayoutInflater inflater = getActivity().getLayoutInflater();
    View view = inflater.inflate(R.layout.fragment_custom, null);
    builder.setView(view);
    // Остальной код
    return builder.create();
}
```

Вы познакомились с базовыми принципами использования диалоговых окон. Вы можете посмотреть устаревшие примеры с AlertDialog и попытаться переделать их под новые способы с фрагментами.

С появлением нового стиля Material Design в библиотеке совместимости появился класс **android.support.v7.app.AlertDialog**, который позволит вам сделать диалоговое окно стильным и красивым.

Дополнительное чтение

Обсуждение статьи на форуме.