**DialogFragment и создание своих диалоговых окон**

Для создания своих диалоговых окон используется компонент **AlertDialog** в связке с классом фрагмента **DialogFragment**. Рассмотрим их применение.

Вначале добавим в проект в папку *layout* новый файл разметки, который будет представлять интерфейс создаваемого диалогового окна. Назовем файл *loginform.xml*:

|  |
| --- |
| <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      android:orientation="vertical"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent">      <LinearLayout          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:orientation="horizontal">          <TextView              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:text="Логин"/>          <EditText              android:id="@+id/login"              android:layout\_width="match\_parent"              android:layout\_height="wrap\_content" />      </LinearLayout>      <LinearLayout          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:orientation="horizontal">          <TextView              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:text="Пароль"/>          <EditText              android:id="@+id/password"              android:layout\_width="match\_parent"              android:layout\_height="wrap\_content"              android:inputType="textPassword"/>      </LinearLayout>  </LinearLayout> |

Данный интерфейс будет представлять форму ввода логина и пароля.

И также изменим разметку MainActivity:

<LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"

    xmlns:tools="<http://schemas.android.com/tools>" android:layout\_width="match\_parent"

    android:layout\_height="match\_parent" android:orientation="vertical">

    <TextView android:id="@+id/loginText"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content" />

    <TextView android:id="@+id/passwordText"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content" />

    <Button

        android:layout\_width="wrap\_content"

        android:layout\_height="wrap\_content"

        android:text="Установить логин и пароль"

        android:onClick="setLoginPass"/>

</LinearLayout>

Здесь определена кнопка, нажатие на которую будет вызывать диалоговое окно. Для отображения введенных в диалоговом окне данных также определены два текстовых поля.

Теперь добавим в приложение новый класс фрагмента:

import android.app.AlertDialog;

import android.app.Dialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.support.v4.app.DialogFragment;

import android.util.Log;

import android.view.View;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class CustomDialogFragment extends DialogFragment implements

        DialogInterface.OnClickListener {

    private View form=null;

    @Override

    public Dialog onCreateDialog(Bundle savedInstanceState) {

        form= getActivity().getLayoutInflater()

                        .inflate(R.layout.loginform, null);

        AlertDialog.Builder builder=new AlertDialog.Builder(getActivity());

        return(builder.setTitle("Форма авторизации").setView(form)

                .setPositiveButton(android.R.string.ok, this)

                .setNegativeButton(android.R.string.cancel, null).create());

    }

    @Override

    public void onClick(DialogInterface dialog, int which) {

        EditText loginBox=(EditText)form.findViewById(R.id.login);

        EditText passwordBox=(EditText)form.findViewById(R.id.password);

        String login = loginBox.getText().toString();

        String password = passwordBox.getText().toString();

        TextView loginText = (TextView)getActivity().findViewById(R.id.loginText);

        TextView passwordText = (TextView)getActivity().findViewById(R.id.passwordText);

        loginText.setText(login);

        passwordText.setText(password);

    }

    @Override

    public void onDismiss(DialogInterface unused) {

        super.onDismiss(unused);

    }

    @Override

    public void onCancel(DialogInterface unused) {

        super.onCancel(unused);

    }

}

Класс фрагмента содержит всю стандартную функциональность фрагмента с его жизненным циклом, но при этом наследуется от класса **DialogFragment**, который добавляет ряд дополнительных функций. И для его создания мы можем использвоать два способа:

* Переопределение метода onCreateDialog(), который возвращает объект Dialog.
* Использование стандартного метода onCreateView().

Для создания диалогового окна в методе onCreateDialog() применяется класс AlertDialog.Builder. С помощью своих методов он позволяет настроить отображение диалогового окна:

* setTitle: устанавливает заголовок окна
* setView: устанавливает разметку интерфейса окна
* setPositiveButton: устанавливает кнопку ввода данных
* setNegativeButton: устанавливает кнопку отмены
* setMessage: устанавливает текст диалогового окна, но при использовании setView данный метод необязателен или может рассматриваться в качестве альтернативы, если нам надо просто вывести сообщение.
* create: создает окно

При нажатии на кнопку OK (создаваемую методом setPositiveButton()) будет срабатывать обработчик onClick(), который будет получать данные из полей диалогового окна и устанавливать их в текстовых блоках MainActivity.

В коде MainActivity определим обработчик нажатия кнопки, который будет запускать диалоговое окно:

|  |
| --- |
| public void setLoginPass(View v) {        new CustomDialogFragment().show(getSupportFragmentManager(),                  "login");  } |

И поле нажатия на кнопку мы сможем ввести данные в диалоговое окно:

