



# Android Programing

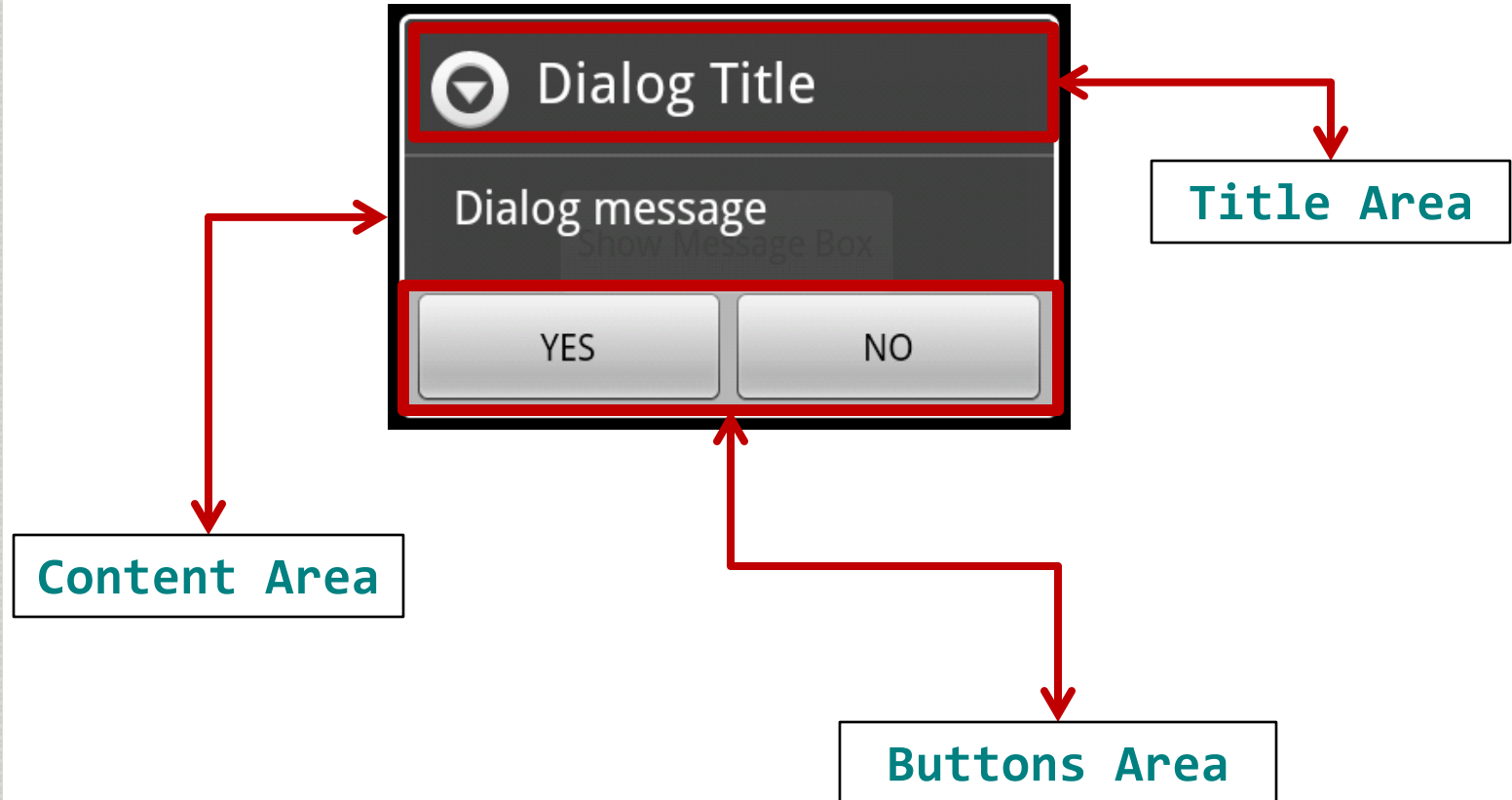
Gavrilut Dragos



# AlertDialog

- This is the equivalent of a MessageBox in Desktop based systems
- Besides the normal behavior (show a popup message with some buttons), an AlertDialog can be customized to look and behave like a modal window.
- Every AlertDialog can be canceled by pressing the Back key (this is the default behavior)

# AlertDialog - layout



# AlertDialog – builder functions

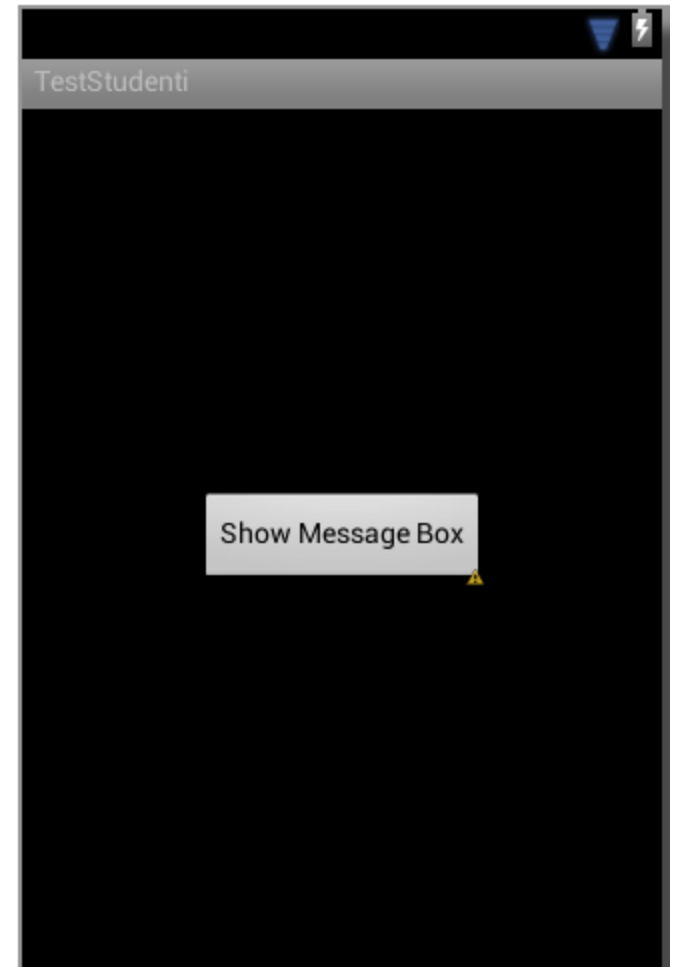
Function	Description
setTitle	Sets the title of the dialog
setMessage	Sets the message that is display in the dialog
setIcon	Sets the icon of the dialog
setPositiveButton	Sets the positive (YES / OK / AGREE) button of the dialog
setNegativeButton	Sets the negative (CANCEL / NO) button of the dialog
setNeutralButton	Sets the 3 <sup>rd</sup> button of the dialog (usually called the neutral button) for the cases when a 3 options dialog is required
setCustomTitle	Sets the title using a custom view
setItems	Sets a list of items to be display in the dialog.
setMultiChoiceItems	Sets a list of items from where multiple items can be selected
setSingleChoiceItems	Sets a list of items from only one item can be selected
setView	Sets a custom view for the content of the dialog

# AlertDialog - Example

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:gravity="center" >

    <Button
        android:id="@+id/MyButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=" Show Message Box "
        android:onClick="ShowMessageBox" />

</LinearLayout>
```

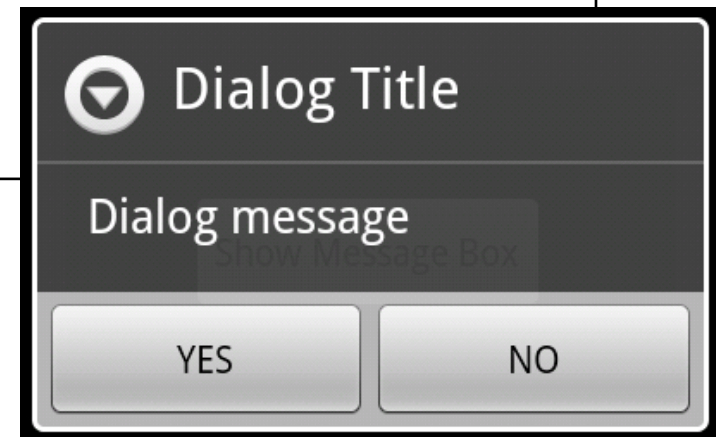


# AlertDialog - Example

```
public void ShowMessageBox(View btn)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Dialog Title");
    builder.setMessage("Dialog message");
    builder.setPositiveButton("YES", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {

        }
    });
    builder.setNegativeButton("NO", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {

        }
    });
    AlertDialog alertDialog = builder.create();
    alertDialog.show();
}
```



# AlertDialog - Example

```
public void ShowMessageBox(View btn)
{
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Picture");
    builder.setMessage("Do you like this picture ?");
    builder.setIcon(getResources().getDrawable(R.drawable.ic_launcher));

    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) { ... }
    });
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) { ... }
    });
    AlertDialog alertDialog = builder.create();
    alertDialog.show();
}
```



Picture

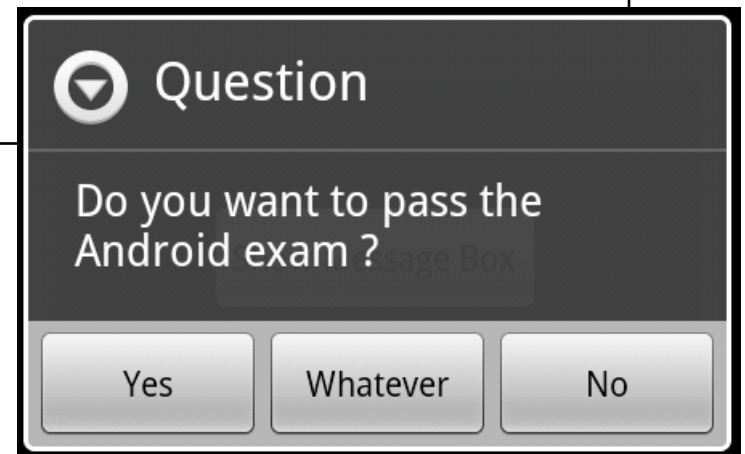
Do you like this picture ?

Ok

No

# AlertDialog - Example

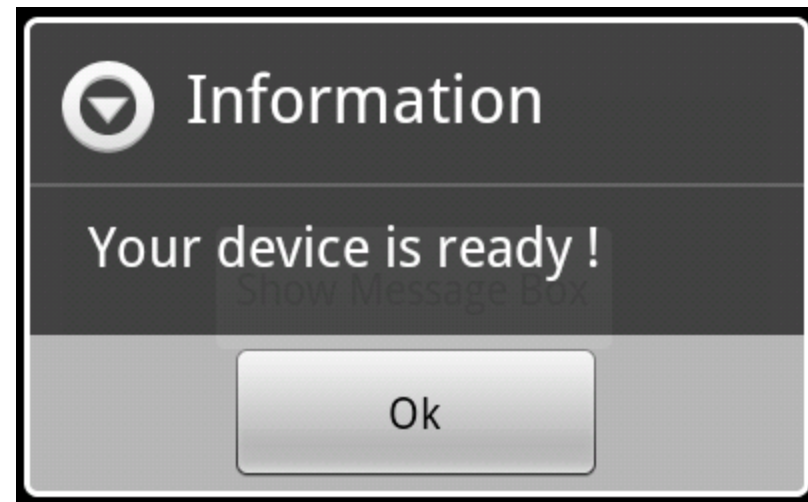
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Question");  
    builder.setMessage("Do you want to pass the Android exam ?");  
    builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNeutralButton("Whatever", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```





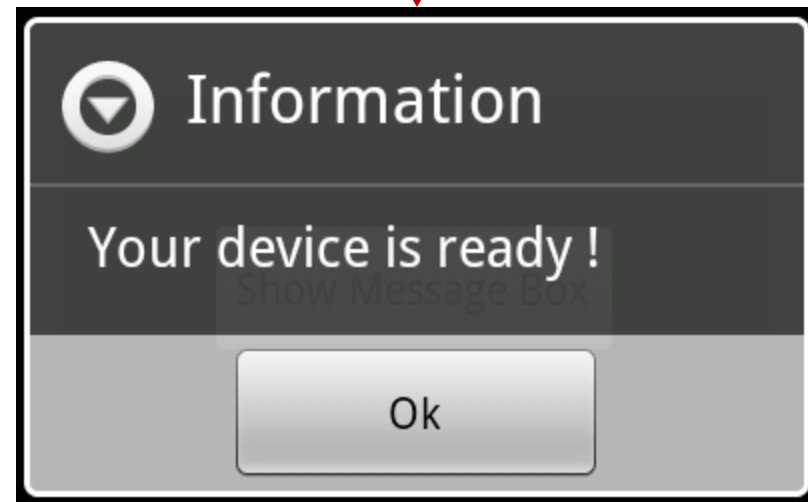
# AlertDialog - Example

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setMessage("Your device is ready !");  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```



# AlertDialog - Example

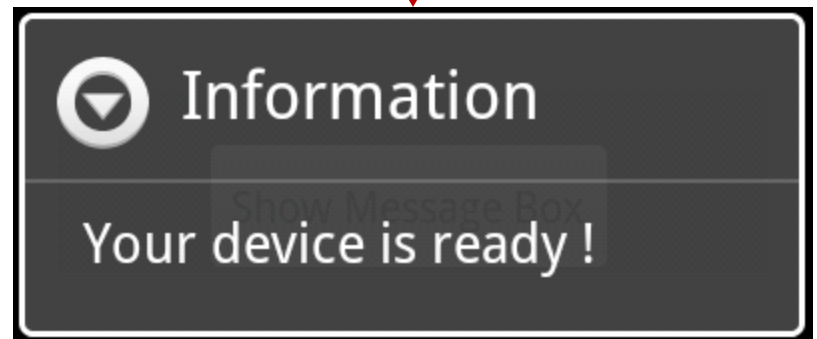
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setMessage("Your device is ready !");  
    builder.setNegativeButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```



# AlertDialog - Example

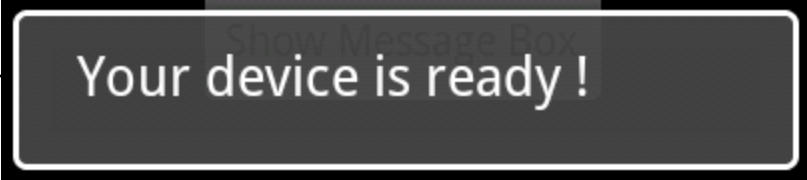
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setMessage("Your device is ready !");  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

Hit "**BACK**" key to exit this dialog




# AlertDialog - Example

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setMessage("Your device is ready !");  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

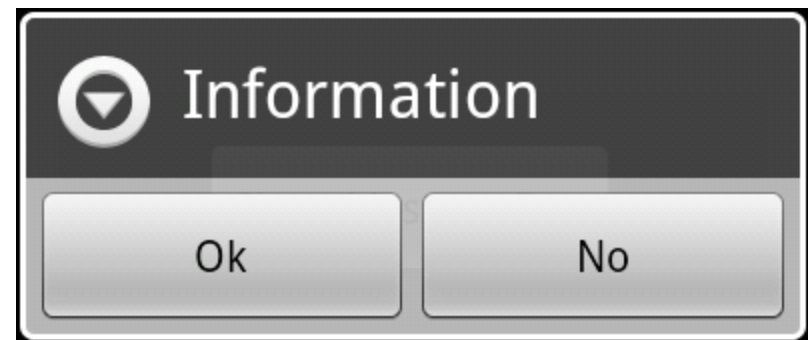
A dark gray rectangular dialog box with a white border. It contains the text "Your device is ready !" in a white, sans-serif font. The dialog is centered on the screen.

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

A dark gray rectangular dialog box with a white border. It features a white circular icon with a downward-pointing triangle on the left. To the right of the icon, the word "Information" is written in a white, sans-serif font. The dialog is centered on the screen.

# AlertDialog - Example

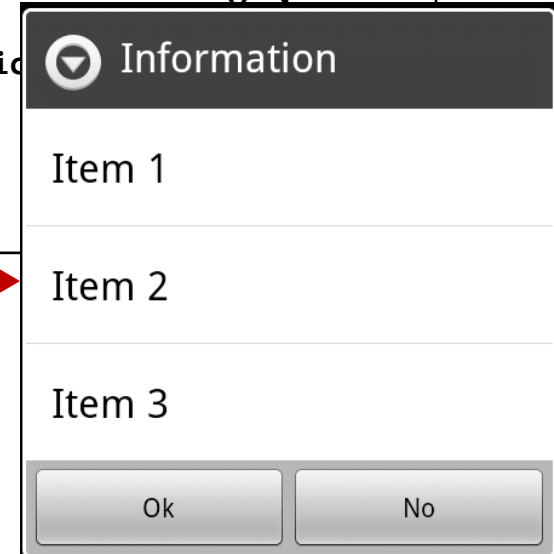
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```



# AlertDialog – Example - Items

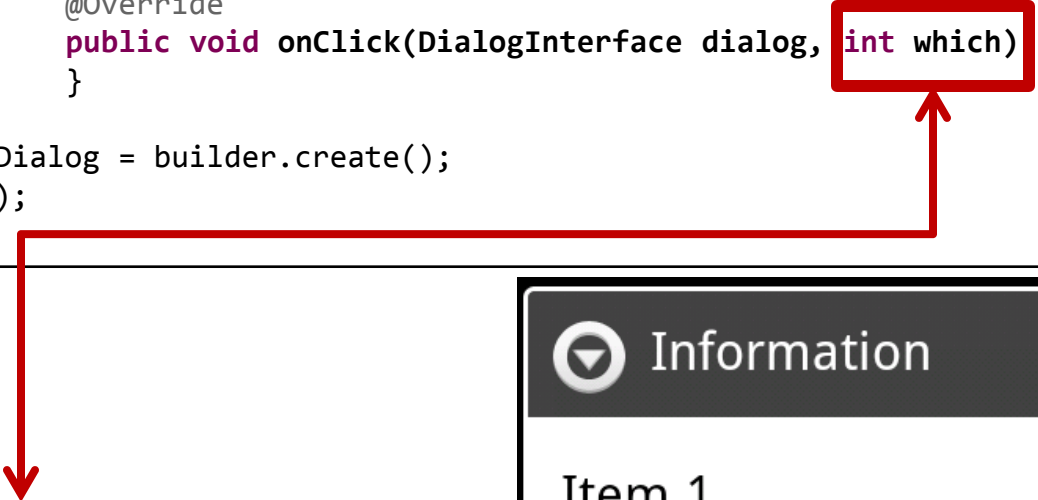
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setItems(new CharSequence[] {"Item 1", "Item 2", "Item 3"},  
        new DialogInterface.OnClickListener() {  
            @Override  
            public void onClick(DialogInterface dialog, int which) {  
            }  
        });  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

Clicking one item will close the alert dialog  
IT IS NOT OK TO HAVE BUTTONS AND ITEMS AT THE  
SAME TIME



# AlertDialog – Example - Items

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setItems(new CharSequence[] {"Item 1", "Item 2", "Item 3"},  
        new DialogInterface.OnClickListener() {  
            @Override  
            public void onClick(DialogInterface dialog, int which) {  
            }  
        });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```



The diagram illustrates the relationship between the code and the UI. A red arrow points from the `int which` parameter in the `onClick` method to a red box. Another red arrow points from this box to a text box explaining the parameter. A third red arrow points from the text box to the 'Item 2' in the UI list.

The “which” parameters refers to the index of the item that was clicked. For example if you click on the second item (“Item 2”) the “which” parameter will be 1 (as the list is 0-indexed).

 Information

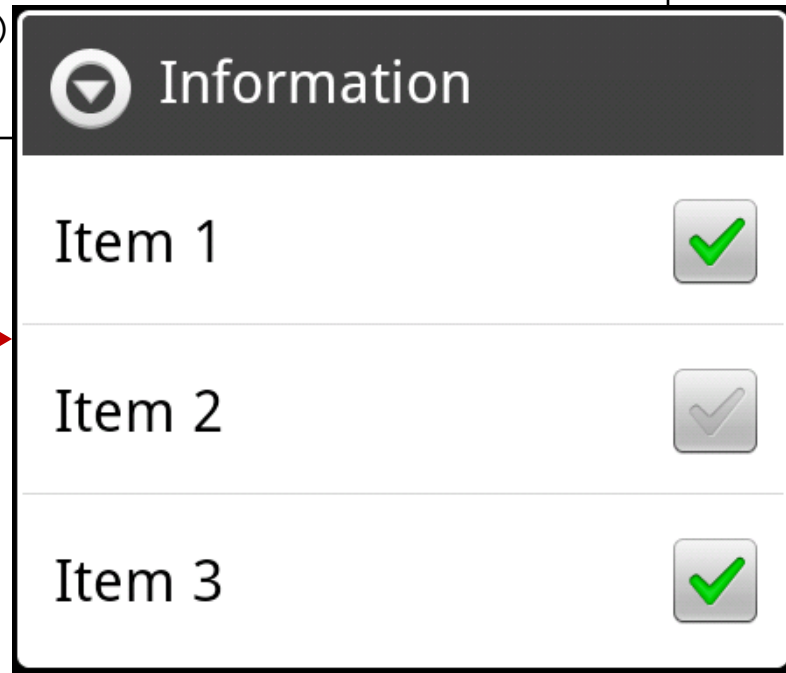
Item 1

Item 2

Item 3

# AlertDialog – Example - Items

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setMultiChoiceItems(new CharSequence[] {"Item 1", "Item 2", "Item 3"},  
                                new boolean[] {true, false, true},  
                                new DialogInterface.OnMultiChoiceClickListener() {  
                                    @Override  
                                    public void onClick(DialogInterface dialog,  
                                                            int which,  
                                                            boolean isChecked) { ... }  
                                })  
    );  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

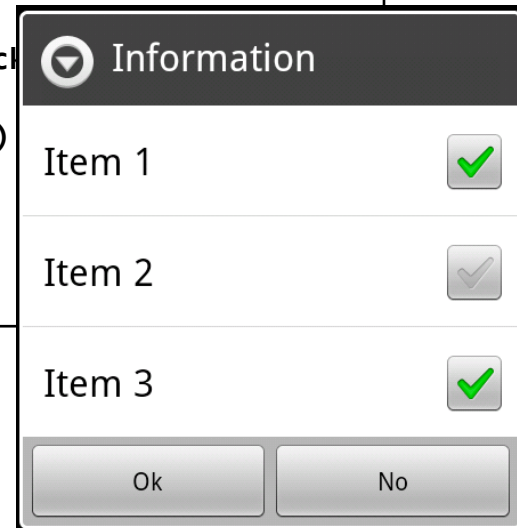


**IT IS NOT OK TO NOT HAVE  
BUTTONS FOR CHECKD ITEMS !**



# AlertDialog – Example - Items

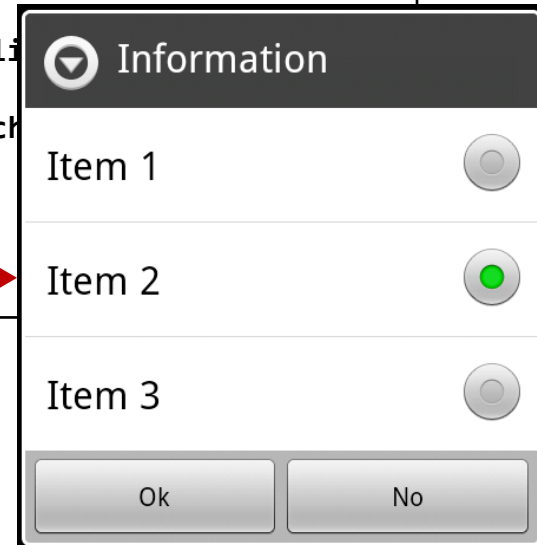
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setMultiChoiceItems(new CharSequence[] {"Item 1", "Item 2", "Item 3"},  
                                new boolean[] {true, false, true},  
                                new DialogInterface.OnMultiChoiceClickListener() {  
                                    @Override  
                                    public void onClick(DialogInterface dialog,  
                                                            int which,  
                                                            boolean isChecked) { ... }  
                                })  
    );  
    builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which)  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```



# AlertDialog – Example - Items

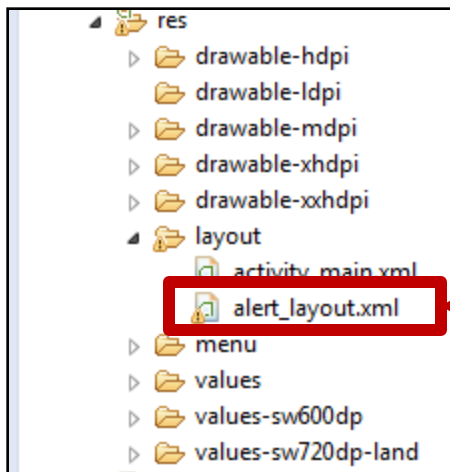
```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Information");  
    builder.setSingleChoiceItems(new CharSequence[] {"Item 1", "Item 2", "Item 3"},  
                                1,  
                                new DialogInterface.OnClickListener() {  
                                    @Override  
                                    public void onClick(DialogInterface dialog,  
                                                            int which) { ... }  
                                }  
    );  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

The second parameter refers to the initial item that should be checked in the a 0-based list. A value of "-1" means that no item should be checked by default.



# Custom Alert Dialog

- Create a new layout with the content that will be used on the custom alert dialog



```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Back"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Next"/>

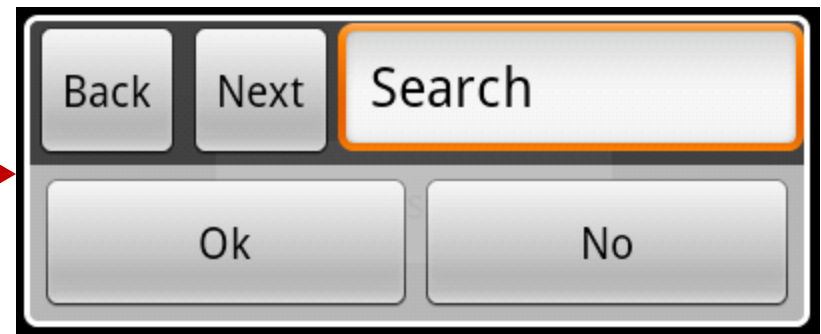
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Search" />

</LinearLayout>
```

# Custom Alert Dialog

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
  
    LayoutInflater inflater = this.getLayoutInflater();  
    builder.setCustomTitle(inflater.inflate(R.layout.alert_layout, null));  
  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

The **LayoutInflater** object can help you convert any layout into a **View** object and use it in a Alert Dialog



# Custom Alert Dialog

```
<TableLayout
    android:layout_width="fill_parent"
    android:layout_height="match_parent" >

    <TableRow android:layout_width="fill_parent" android:layout_height="match_parent" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="User" />
        <EditText
            android:layout_width="200dp"
            android:layout_height="wrap_content" />
    </TableRow>

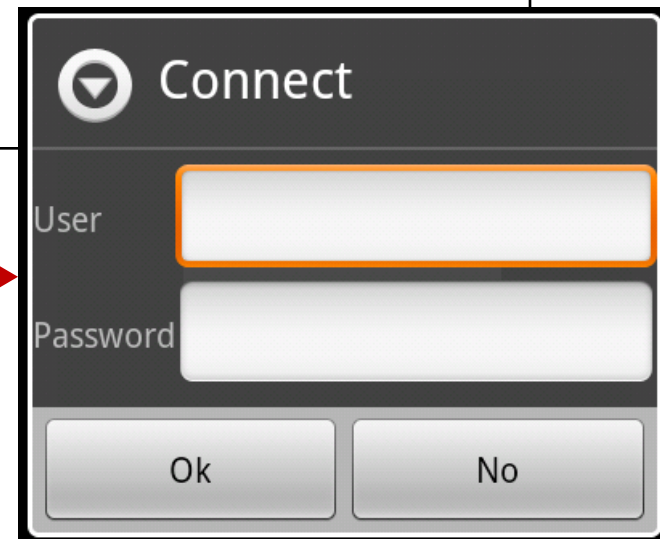
    <TableRow android:layout_width="fill_parent" android:layout_height="match_parent" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Password" />
        <EditText
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:inputType="textPassword" />
    </TableRow>

</TableLayout>
```

# Custom Alert Dialog

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Connect");  
  
    LayoutInflater inflater = this.getLayoutInflater();  
    builder.setView(inflater.inflate(R.layout.alert_layout, null));  
  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

Use [builder.setView](#) to set a new custom view to the content area of an Alert Dialog



# Custom Alert Dialog

```
public void ShowMessageBox(View btn) {  
    AlertDialog.Builder builder = new AlertDialog.Builder(this);  
    builder.setTitle("Connect");  
    builder.setCancelable(false);  
  
    LayoutInflater inflater = this.getLayoutInflater();  
    builder.setView(inflater.inflate(R.layout.alert_layout, null));  
  
    builder.setPositiveButton("Ok", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    builder.setNegativeButton("No", new DialogInterface.OnClickListener() {  
        @Override  
        public void onClick(DialogInterface dialog, int which) { ... }  
    });  
    AlertDialog alertDialog = builder.create();  
    alertDialog.show();  
}
```

By default, every Alert Dialog is cancelable. This means that hitting the "Back" key will close that dialog. Use [builder.setCancelable](#) to overwrite this behavior.

# Toast

- A “Toast” is a short message that is shown to the user (an informative message).
- Create:

**Toast toast = Toast.makeText(context, text, duration);**

Where:

1. “context” is the context of the current Activity
2. “text” is the text that should be displays
3. “duration” is one of the following
  - Toast.LENGTH\_LONG
  - Toast.LENGTH\_SHORT



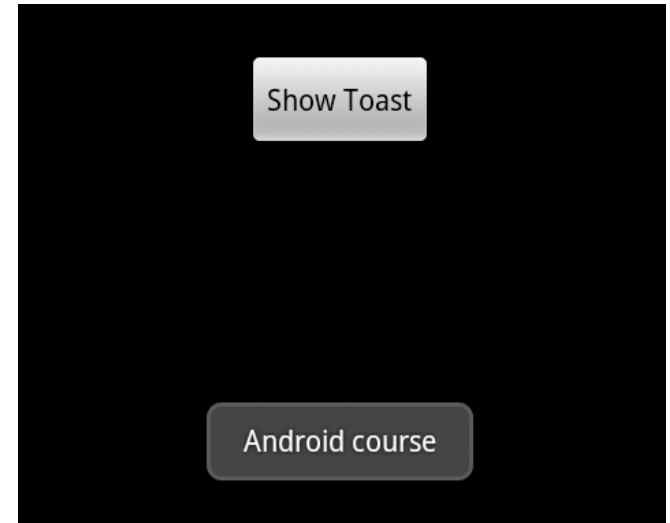
# Toast - Example

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:gravity="center" >

    <Button
        android:id="@+id/MyButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=" Show Toast"
        android:onClick="ShowToast" />

</LinearLayout>
```

```
public void ShowToast(View btn)
{
    Toast toast = Toast.makeText(this, "Android course", Toast.LENGTH_SHORT );
    toast.show();
}
```

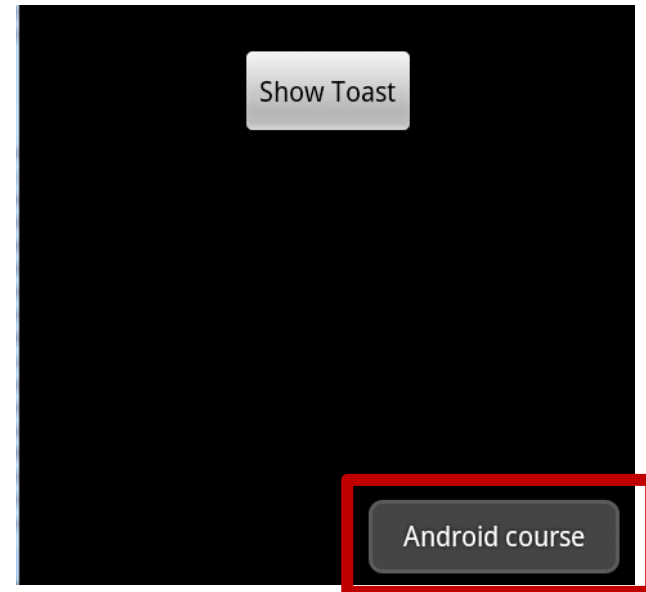


# Toast – Example (position)

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:gravity="center" >

    <Button
        android:id="@+id/MyButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text=" Show Toast"
        android:onClick="ShowToast" />

</LinearLayout>
```



```
public void ShowToast(View btn)
{
    Toast toast = Toast.makeText(this, "Android course", Toast.LENGTH_SHORT );
    toast.setGravity(Gravity.BOTTOM/Gravity.RIGHT, 0, 0);
    toast.show();
}
```

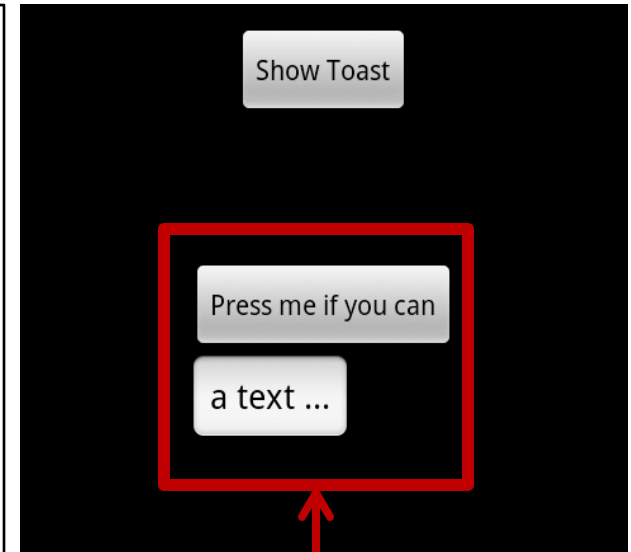
# Custom Toast – Example

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Press me if you can"/>

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="a text ..."/>

</LinearLayout>
```



```
public void ShowToast(View btn)
{
    Toast toast = Toast.makeText(this, "Android course", Toast.LENGTH_SHORT );

    LayoutInflater inflater = this.getLayoutInflater();
    toast.setView(inflater.inflate(R.layout.toast_layout, null));

    toast.show();
}
```

# Intents

- An “Intent” is a way to communicate between applications, activities, services, receivers, and so on
- An “Intent” is an object that contains the following informations:
  - Component name → the component that should receive this intent and process it. This field (if set) is a package name of the application that should process the intent
  - Action → a string that represents the action that needs to be performed. There are some predefined actions (such as phone calls, getting a picture, ...). Every application can create its own actions
  - Data → URI for the data needed by the Intent and the MIME type of the data.
  - Category → the category of the intent. This can be set to specify what kind of components can process this intent (browsers, launchers, ...)
  - Extra → pairs (key,value) of extra informations send to the component that processes the Intent
  - Flags → different flags for the intent

# Intents

- Create

- `Intent intent = new Intent("<u><action></u>")`
- `Intent intent = new Intent("<u><action></u>", Uri)`
- `Intent intent = new Intent(Context, Class)`
- `Intent intent = new Intent("<u><action></u>", Uri, Context, Class)`

- Starting an intent

- `Context.startActivity (Intent intent)`
- `Context.startActivity (Intent intent, Bundle options)`
- `Context.startActivityForResult (Intent intent, int requestCode)`
- `Context.startActivityForResult (Intent intent, int requestCode, Bundle options)`
- `Context.startService (Intent intent)`
- `Context.sendBroadcast (Intent intent)`

# Intents

- Start a new activity from another

```
public class MainActivity extends Activity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        ...  
    }  
  
    public void OnButtonPressed() {  
        Intent i = new Intent(this, SecondaryActivity.class);  
        startActivity(i);  
    }  
}
```

# Intents

- Share data between activities

```
public class MainActivity extends Activity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {...}  
  
    public void OnButtonPressed() {  
        Intent i = new Intent(this, SecondaryActivity.class);  
        i.putExtra("Param1", "some_text");  
        startActivityForResult(i, 123);  
    }  
  
    @Override  
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
        if ((resultCode == RESULT_OK) && (requestCode == 123)) {  
            if (data.hasExtra("Return1")) {  
                ...  
            }  
        }  
    }  
}
```

# Intents

- Share data between activities

```
public class SecondaryActivity extends Activity

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_secondary);

        Bundle extraInfo = getIntent().getExtras();
        String someInfo = extraInfo.getString("Param1");
    }

    @Override
    public void finish() {
        Intent data = new Intent();
        data.putExtra("Return1", "Return value");
        setResult(RESULT_OK, data);
        super.finish();
    }
}
```



# Intents

- Using the camera to take a picture

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent imageIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        imageIntent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(new File("...")));
        startActivityForResult(imageIntent,1278);
    }
}
```

- Or a video

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent imageIntent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
        imageIntent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(new File("...")));
        startActivityForResult(imageIntent,1278);
    }
}
```

# Intents

- Open the market to a specific application

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        String market="market://details?id="+ "com.fii.app";
        Intent marketIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(market));
        marketIntent.addFlags(    Intent.FLAG_ACTIVITY_NO_HISTORY |
                                Intent.FLAG_ACTIVITY_CLEAR_WHEN_TASK_RESET);
        startActivity(marketIntent);
    }
}
```

- Open the browser to a webpage

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent browser = new Intent(Intent.ACTION_VIEW, Uri.parse("www.google.com"));
        startActivity(browser);
    }
}
```

# Intents

- Send a SMS

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Uri uri = Uri.parse("smsto:0740123456");
        Intent it = new Intent(Intent.ACTION_SENDTO, uri);
        it.putExtra("sms_body", "text to send");
        startActivity(it);
    }
}
```

- Send an email

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent emailIntent = new Intent(Intent.ACTION_SEND);
        emailIntent.putExtra(Intent.EXTRA_TEXT, "Email text");
        emailIntent.putExtra(Intent.EXTRA_SUBJECT, "Subject");
        emailIntent.setType("application/image");
        Uri attachament = Uri.parse("file://" + filePath);
        emailIntent.putExtra(Intent.EXTRA_STREAM, attachament);
        startActivity(emailIntent);
    }
}
```

# Intents

- Make a phone call

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent intent = new Intent(Intent.ACTION_CALL);
        //needs: android.permission.CALL_PHONE
        intent.setData(Uri.parse("tel:0740123456"));
        startActivity(intent);
    }
}
```

- Start the phone application

```
public class MainActivity extends Activity

    public void OnButtonPressed() {
        Intent intent = new Intent(Intent.ACTION_DIAL);
        intent.setData(Uri.parse("tel:0740123456"));
        startActivity(intent);
    }
}
```