**LISTVIEW**

**Che cosa è un ListView? Che cosa è un Adapter? In che modo ListView e Adapter interagiscono?**

Un ***ListView*** è un Widget specifico per le liste, raggruppa diversi elementi e li visualizza in un elenco scorrevole verticale. Gli elementi vengono automaticamente inseriti nell’elenco utilizzando un ***Adapter*** che estrae il contenuto da un’origine come un array o un DB.

Un Adapter è un oggetto, che può essere agganciato a qualsiasi ***AdapterView*** (ovvero ListView), che gestisce la visualizzazione degli elementi in un ***AdapterView***: fornisce accesso a tali elementi ed è responsabile della creazione di una ***View*** per ogni elemento nel data set.

L’***Adapter*** viene assegnata a ***ListView*** tramite il metodo ***setAdapter*** sull’oggetto ***ListView***: *listView.setAdapter(arrayAdapter)*

**Descrivere schematicamente tutto ciò che serve per far comparire sullo schermo un ListView (semplice) con una lista di nomi (definiti in un array). Si specifichi per ogni passo l’istruzione, il widget o il file XML, necessari per il ListView.**

* Si definisce nel file ***xml*** (***activity\_main***) un ***widget ListView***, specificando l’id (mylistview) e i vari parametri:

|  |
| --- |
| *<ListView*  *android:id=”@+id/mylistview”*  *android:layout\_width=”wrap\_content”*  *android:layout\_width=”wrap\_content”*  *</ListView>* |

* Si crea un ulteriore file ***xml*** per rappresentare il ***singolo elemento della lista***, coi relativi parametri:

|  |
| --- |
| *<TextView*  *android:id=”@+id/textViewList”*  *android:layout\_width=”wrap\_content”*  *android:layout\_height=”wrap\_content”*  *android:text=”” android:padding=”10dp”*  *android:textSize=”22dp”*  */>* |

* All’interno del file Java ***MainActivity*** si definisce un ***Array di stringhe*** contente nomi di persone:

|  |
| --- |
| *String [] array = {“Pasquale”,”Maria”,”Michele”, …}* |

* Si definisce un ***ArrayAdapter***, dove ***context*** rappresenta il contesto dell’applicazione, ***R.layout.list\_element***rappresenta il file di layout dove viene definito il singolo elemento nella lista, ***R.id.textViewList*** che rappresenta l’identificativo del TextView nel file di layout e poi c’è ***array*** che rappresenta le informazioni da visualizzare nel ListView:

|  |
| --- |
| *ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(context, R.layout.list\_element, R.id.textViewList, array);* |

* Viene individuato il ***widget ListView***:

|  |
| --- |
| *listView = (ListView)findViewById(R.id.mylistview);* |

* Infine si associa ***l’adapter al widget***:

|  |
| --- |
| *listView.setAdapter(arrayAdapter);* |

**Che cosa è un Adapter customizzato per un ListView? Si fornisca una spiegazione possibilmente con frammenti di codice.**

Un ***Adapter customizzato*** nasce dall’esigenza di creare una visualizzazione personalizzata di un ***ListView***.

Un ***Adapter*** customizzato per un ***ListView*** è un ***Adapter*** che consente di adattare, ad un ***ListView***, un elemento che contiene dei sottoelementi.

L’elemento da adattare dev’essere descritto in un file ***XML***, in cui si inserisce il layout dell’elemento (compreso, pertanto, dei sottoelementi).

Per creare un adapter customizzato, occorre creare una classe che estende ***ArrayAdapter***, al cui interno va aggiunto il costruttore (avente come parametri il ***context***, la ***resource*** id dell’elemento da adattare, ed una ***lista di elementi*** da adattare) ed il metodo ***getView():*** quest’ultimo viene invocato ogniqualvolta è necessario mostrare un elemento del ***ListView*** (ad esempio, al lancio dell’applicazione o a seguito di uno scroll), ed è utilizzato anche per eseguire ***l’inflate*** del file di layout dell’elemento nel ***ListView***.

Per adattare un ***ListView***, nell’activity in cui tale lista è presente occorre seguire i prossimi step:

1. Creare l’adapter customizzato ed aggiungerlo al ***ListView***:

|  |
| --- |
| *CustomAdapter adapter = new CustomAdapter(getApplicationContext(), R.layout.?, new ArrayList<Object>);*  *ListView listView = (ListView) findViewById(R.id.listView);*  *listView.setAdapter(adapter);* |

1. Aggiungere i vari oggetti all’adapter:

|  |
| --- |
| *for (…) { object = …; adapter.add(object); }* |

1. Aggiungere, se richiesto, un listener sul ***ListView***:

|  |
| --- |
| *listView.setOnItemClickListener(new OnItemClickListener() {*  *@Override*  *public void onItemClick (AdapterView<?> parent, View view, int position, long id) {*  *…*  *}*  *});* |

**Un listview prevede OnItemClickListener che gestisce i click sugli elementi della lista chiamando il metodo onItemClick al quale viene passato un riferimento dell’elemento selezionato. Se usiamo un listview customizzato, in cui ogni elemento della lista è composto da vari sottoelementi (es. una foto, un nome, un numero), il riferimento passato al metodo onItemClick non distingue quale dei sottoelementi è stato selezionato. Come si può fare per reagire in maniera diversa in funzione di quale dei sottoelementi è stato selezionato con il click?**

Se usiamo un ***ListView*** customizzato, per distinguere quale sottoelemento è stato selezionato è opportuno definire dapprima un ***Adapter*** ***customizzato***. Supponendo che esso sia stato già creato ed “agganciato” al ***ListView***, allora tale problema deriva dal fatto che è stato passato un ***OnItemClickListener*** sull’intero elemento della lista, “creato” dal ***CustomAdapter***. Per risolvere, è opportuno agganciare un onclick listener su ogni sottoelemento del ***ListView***; inoltre, per far sì che si possa risalire alla posizione dell’elemento dell’adapter cliccato, è possibile utilizzare il metodo ***setTag()*** su ogni sottoelemento passando la posizione dell’elemento adattato (questo va fatto nel metodo ***getView()*** del custom adapter): questa potrà essere ricavata in un altro punto con il metodo ***getTag()*** sullo stesso sottoelemento.

**Si consideri il seguente frammento di codice che definisce un OnItemClickListener per un ListView.**

*listView.setOnItemClickListener(new OnItemClickListener() {*

*@Override*

*public void onItemClick(AdapterView<?> parent, View view, int position, long id) {*

*String str = listView.getItemAtPosition(position).toString();*

*// Fai qualcosa con l’elemento …*

*}*

*});*

**Si spieghi il ruolo del parametro position.**

Il metodo ***onItemClick()*** viene invocato quando si effettua un click su un elemento del listview. Si assume che sul ***listView*** è stato agganciato un semplice ***ArrayAdapter<String>.*** Il parametro ***position***, al click, tiene traccia dell’indice dell’array che è stato adattato al list view dall’adapter: in questo caso, viene ottenuta, con la prima linea di codice del metodo ***onItemClick(),*** una stringa dell’elemento in posizione position del ***listView***.

**Il seguente frammento di codice serve a gestire un listView con gli elementi di un array. Completare il codice dell’onItemClickListener (riquadro vuoto) in modo tale che quando l’utente clicca un elemento visualizzato nel listView venga mostrato un Toast con il nome cliccato.**

|  |
| --- |
| **@Override**  **protected void onCreate(Bundle savedInstanceState) {**  **super.onCreate(savedInstanceState);**  **setContentView(R.layout.activity\_main);**  **String [] nazioni = {”Italia",”Francia",”Spagna",”Germania", ”Portogallo", ”Svizzera", ”Belgio", ”Polonia", ”Grecia", ”Svezia"};**  **listView = (ListView)findViewById(R.id.mylistview);**  **ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(this, R.layout.list\_element, R.id.textViewList, array);**  **listView.setAdapter(arrayAdapter);**  **listView.setOnItemClickListener(new OnItemClickListener() {**  **@Override**  **public void onItemClick(AdapterView<?> parent, View view, int n, long label) {**  *String str = listView.getItemAtPosition(position).toString();*  *// Show Toast*  *Toast.makeText(getApplicationContext(), “Nazione :" +str, Toast.LENGTH\_LONG).show();*  **}**  **});**  **}** |

**Il seguente codice incompleto è un CustomAdapter per una lista customizzata di oggetti Object. Ogni oggetto Object possiede i getter getString() e getInt(). Il file di layout “list\_element” contiene 2 TextView con i seguenti identificativi: “stringa” e “intero”. Completare il CustomAdapter per creare la view di ogni singolo elemento.**

|  |
| --- |
| **public class CustomAdapter extends ArrayAdapter<Object> {**  **private int resource;**  **private LayoutInflater inflater;**  **public CustomAdapter(Context context, int resourceId, List<Objects> objects) {**  **super(context, resourceId, objects);**  **resource = resourceId;**  **inflater = LayoutInflater.from(context);**  **}**  **@Override**  **public View getView(int position, View v, ViewGroup parent) {**  **if (v == null) {**  **v = inflater.inflate(R.layout.list\_element, null);**  **}**  *TextView tv1 = v.findViewById(R.id.stringa);*  *TextView tv2 = v.findViewById(R.id.intero);*  *Object o = getItem(position);*  *tv1.setText(o.getString());*  *tv2.setText(o.getInt());*  **return v;**  **}**  **}** |