LISTVIEW và RECYCLERVIER

LAB 02

PHÁT TRIỂN ỨNG DỤNG TRÊN THIẾT BỊ DI ĐỘNG

I. GIỚI THIỆU NỘI DUNG

- Sử dụng ListView control với mảng dữ liệu định sẵn.
- Sử dụng ListView với mảng dữ liệu được lưu trong Xml:
- Sử dụng ArrayList và Listview control
- Sử dụng ArrayList và ListView mà từng phần tử trong ArrayList là các Object bất kỳ.
- Sử dụng CustomAdapter cho Listview
- Sử dụng Gridview và Spinner
- Sử dụng RecylerView

II. GIỚI THIỆU CÁC BƯỚC XÂY DỰNG LISTVIEW

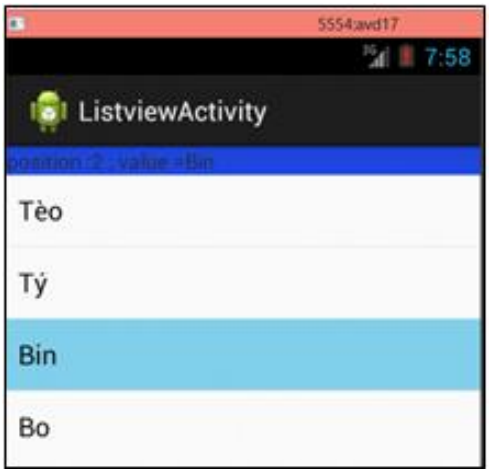
Bước	Nội dung	Ví dụ
1	Khởi tạo đối tượng listview: findViewById từ file XML hoặc tạo bằng code	ListView lvPerson = (ListView) findViewById(R.id.***lv_person***);
2	Load/Khởi tạo mảng chứa dữ liệu sẽ được hiển thị trong listview	**final** String arr[] = {**"Teo"**, **"Ty"**, **"Bin"**, **"Bo"**};
3	Xây dựng adapter	ArrayAdapter<String> adapter = **new** ArrayAdapter<String> (**this**, android.R.layout.***simple_list_item_1***, arr);
4	SetAdapter cho listview	lvPerson.setAdapter(adapter);
5	Xử lý các thao tác trên listview (click, longClick,…)	lvPerson.setOnItemClickListener(**new** AdapterView.OnItemClickListener() { **public void** onItemClick(AdapterView<?> arg0, View arg1, **int** arg2, **long** arg3) { *//đối số arg2 là vị trí phần tử trong Data Source (arr)* tvSelection.setText(**"position :"** + arg2 + **" ; value ="** + arr[arg2]); } });

Sinh viên thực hành các bước xây dựng listview theo hướng dẫn trực tiếp của giảng viên tại lớp.

III. YÊU CẦU THỰC HÀNH

1. Trường hợp 1: Sử dụng ListView control với mảng dữ liệu định sẵn

Sử dụng ListView control với mảng dữ liệu định sẵn. Xây dựng Listview như hình minh họa:



- Giao diện trên có 2 control:

+ **ListView**: dùng để hiển thị mảng dữ liệu

+ **TextView**: có màu xanh lục: Dùng để hiển thị vị trí và giá trị của phần tử được chọn trong ListView.

2. Trường hợp 2: Sử dụng ArrayList và Listview control

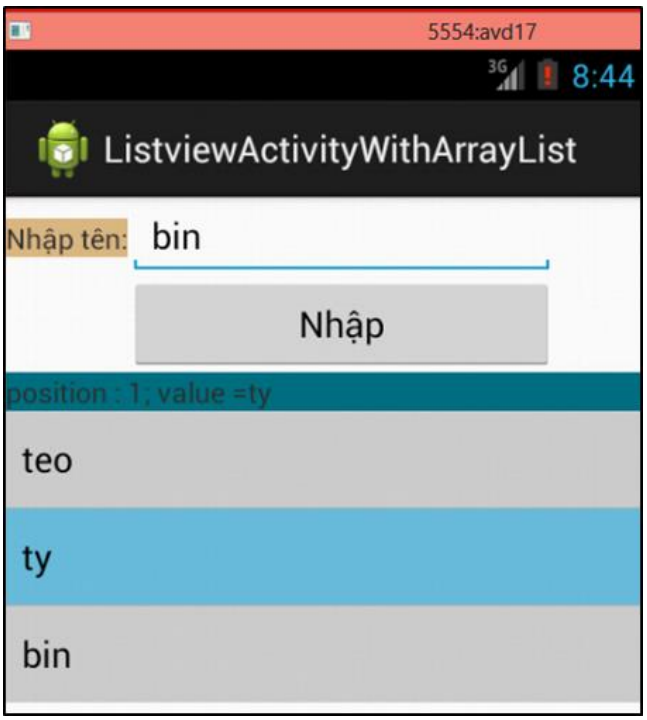
Xây dựng ứng dụng như sau:

- Mô tả:

+ Nhập dữ liệu và nhấn nút "Nhập" thì sẽ đưa vào ArrayList và hiển thị lên ListView.

+ Nhấn vào phần tử nào thì hiển thị vị trí và giá trị của phần tử đó lên TextView

+ Nhấn thật lâu (long click) vào phần tử nào đó trên ListView thì sẽ xóa phần tử đó.



- Hướng dẫn:

```
        //1. Tạo ArrayList object
        names = new ArrayList<String>();

        …
        //4. Xử lý sự kiện nhấn nút Nhập
        btnSubmit.setOnClickListener(new View.OnClickListener() {
```

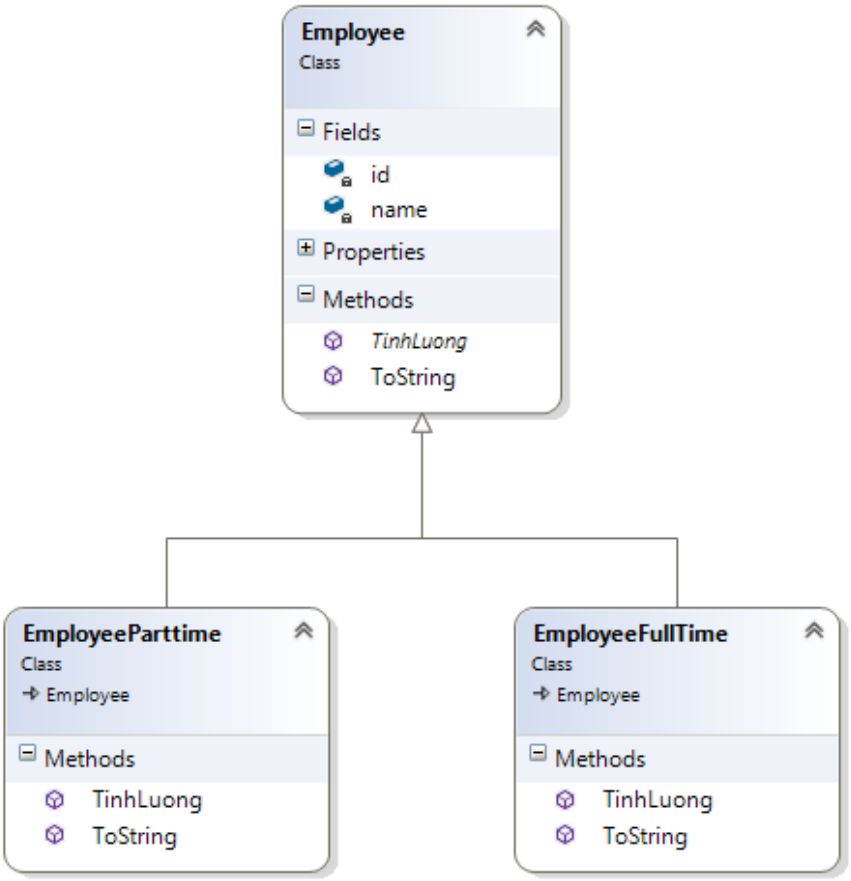
```
            public void onClick(View arg0) {
                //Thêm dữ liệu mới vào arraylist

                 …

                 //Cập nhật dữ liệu mới lên giao diên
                adapter.notifyDataSetChanged();
            }
        });
        //5. Xử lý sự kiện chọn một phần tử trong ListView
        lvPerson.setOnItemClickListener(…);
        //6. xử lý sự kiện Long click
        lvPerson.setOnItemLongClickListener(…);
    }
}
```

3. Trường hợp 3: Sử dụng ArrayList và ListView mà từng phần tử trong ArrayList là các Object bất kỳ:

Xây dựng ứng dụng theo mô tả sau: Có 2 loại nhân viên: Nhân viên chính thức (EmployeeFullTime) và nhân viên thời vụ (EmployeePartime). Mỗi loại nhân viên sẽ có cách tính lương khác nhau. Mỗi nhân viên có phương thức toString để xuất thông tin, Nội dung xuất khác nhau. Xem giao diện chương trình:

Hướng dẫn:

- Sơ đồ các class lưu thông tin nhân viên:



- Thông tin mã màu:

```
<resources>
   <color name="green">#008000</color>
   <color name="white">#FFFFFF</color>
</resources>
```

- Override hàm tinhLuong và toString trên các class EmployeeFullTime và EmployeePartTime để hiển thị thông tin tương ứng:

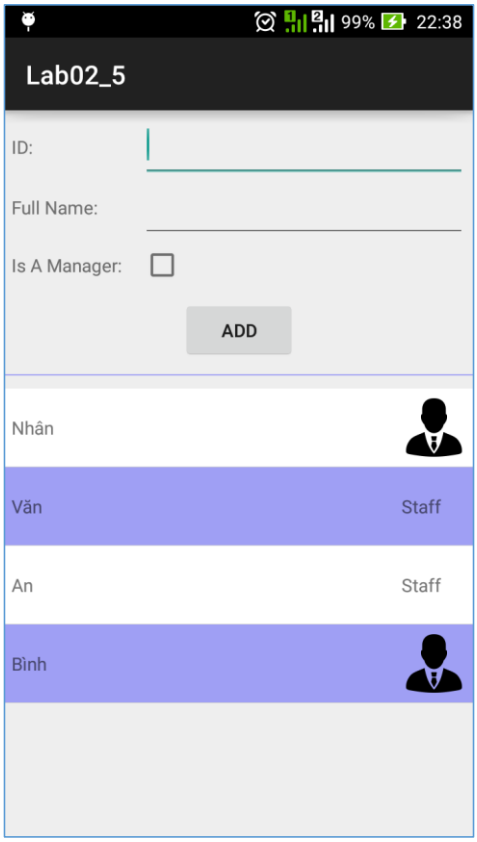
```
public class EmployeeFulltime extends Employee {
    @Override
    public double tinhLuong() {…}

    @Override
    public String toString() {return super.toString() + …;}

}
```

```
public class EmployeeParttime extends Employee {
    @Override
    public double tinhLuong() {…}

    @Override
    public String toString() {return super.toString() + …}

}
```

- Hàm xử lý thêm một nhân viên mới

```
public void addNewEmployee() {
        //Lấy ra đúng id của Radio Button được checked
        int radId = rgType.getCheckedRadioButtonId();
        String id = etId.getText().toString();
        String name = etName.getText().toString();
        if (radId == R.id.rd_chinhthuc) {
            //tạo instance là FullTime
            employee = new EmployeeFulltime();
        } else {
            //Tạo instance là Partime
            employee = new EmployeeParttime();
        }
        //FullTime hay Partime thì cũng là Employee nên có các hàm này là hiển nhiên
        employee.setId(id);
        employee.setName(name);
        //Đưa employee vào ArrayList
        employees.add(employee);
        //Cập nhập giao diện
        adapter.notifyDataSetChanged();
    }
}
```

4. Trường hợp 4: Sử dụng CustomAdapter cho Listview



Xây dựng ứng dụng với giao diện như sau:

Người dùng nhập thông tin ở trên, sau đó nhấn Add. Thông tin employee sẽ được hiện xuống listview. Nếu là Manager thì hiện thêm icon manager ở bên phải ngược lại hiện chữ Staff. Ngoài ra, giữa 2 employee liên tiếp trong listview sẽ được hiện background màu khác nhau cho dễ nhìn.

Hướng dẫn:

- Thông tin mã màu và các resource files:

```
<resources>
    <color name="white">#FFFFFF</color>
    <color name="light_blue">#550000FF</color>
</resources>
```

```
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="margin_base">5dp</dimen>
    <dimen name="margin_basex2">10dp</dimen>
</resources>
```

```
<resources>
    <string name="app_name">Lab02_5</string>
    <string name="id">ID:</string>
    <string name="full_name">Full Name:</string>
    <string name="is_manager">Is A Manager:</string>
    <string name="staff">Staff</string>
    <string name="add">Add</string>
</resources>
```

- Save ảnh sau thành ic_manager.png rồi kéo thả và thư mục drawable

- Để sử dụng CustomAdapter còn xây dựng 2 file: item_employee.xml và EmployeeAdapter.java. Trong đó, file item_employee.xml là file định nghĩa nội dung view hiển thị cho 1 dòng (1 nhân viên) trong listview. File EmployeeAdapter.java là file Custom Adapter để hiển thị nội dung lên listview.

  + File item_employee.xml cần chứa:

  - 1 textview để hiển thị tên nhân viên
  - 1 textview hiển thị Staff dành cho nhân viên
  - 1 imageview để hiển thị icon dành cho manager

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

        <TextView
            android:id="@+id/item_employee_tv_fullname"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:layout_weight="1"
            android:ellipsize="end"
            android:singleLine="true" />

        <TextView
            android:id="@+id/item_employee_tv_position"
            android:layout_width="50dp"
            android:layout_height="wrap_content"
            android:layout_gravity="center_vertical"
            android:singleLine="true" />

        <ImageView
            android:id="@+id/item_employee_iv_manager"
            android:layout_width="50dp"
            android:layout_height="50dp"
            android:scaleType="centerCrop"
            android:src="@drawable/ic_manager"
            android:visibility="gone"/>
    </LinearLayout>

…
```

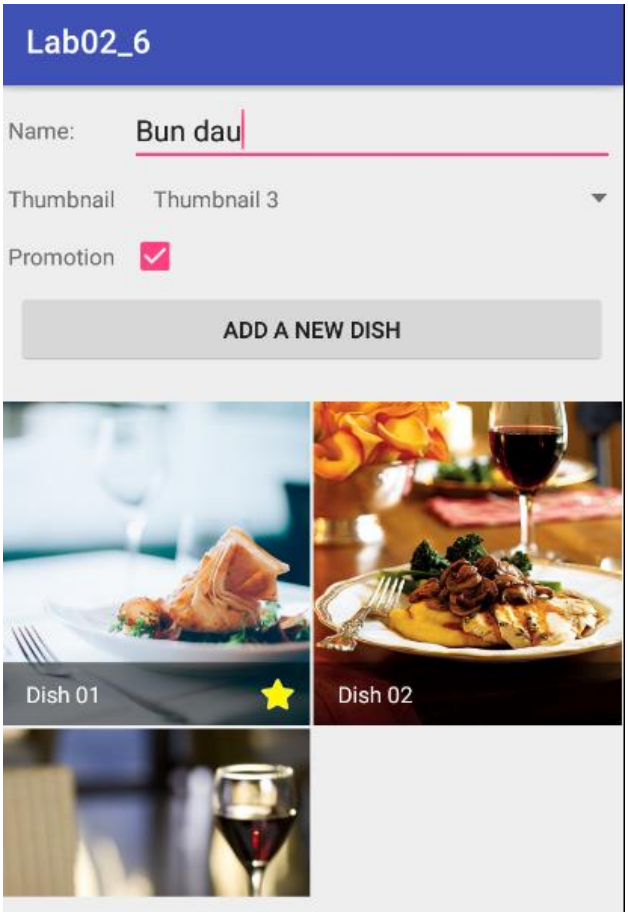
+ File EmployeeAdapter.java. Sinh viên tham khảo nội dung:

```
public class EmployeeAdapter extends ArrayAdapter<Employee> {
    private Activity context;

    public EmployeeAdapter(Activity context, int layoutID, List<Employee>
objects) {
        super(context, layoutID, objects);
        this.context = context;
    }

    @Override
    public View getView(final int position, View convertView, ViewGroup parent)
{
        if (convertView == null) {
            convertView =
LayoutInflater.from(context).inflate(R.layout.item_employee, null,
                    false);
        }

        // Get item
        Employee employee = getItem(position);

        // Get view
        TextView tvFullName = (TextView)
convertView.findViewById(R.id.item_employee_tv_fullname);
        TextView tvPosition = (TextView)
convertView.findViewById(R.id.item_employee_tv_position);
        ImageView ivManager = (ImageView)
convertView.findViewById(R.id.item_employee_iv_manager);
        LinearLayout llParent = (LinearLayout)
convertView.findViewById(R.id.item_employee_ll_parent);

        // Set fullname
        if (employee.getFullName()!=null) {
            tvFullName.setText(employee.getFullName());
        }
        else tvFullName.setText("");

        // If this is a manager -> show icon manager. Otherwise, show Staff in
tvPosition
        if (employee.isManager())
        {
            ivManager.setVisibility(View.VISIBLE);
            tvPosition.setVisibility(View.GONE);
        }
        else
        {
            ivManager.setVisibility(View.GONE);
            tvPosition.setVisibility(View.VISIBLE);
            tvPosition.setText(context.getString(R.string.staff));
        }

        // Show different color backgrounds for 2 continuous employees
        if (position%2==0)
        {
            llParent.setBackgroundResource(R.color.white);
        }
        else
        {
```

```
                llParent.setBackgroundResource(R.color.light_blue);
            }

            return convertView;
        }
}
```

5. Sử dụng GridView, Spinner

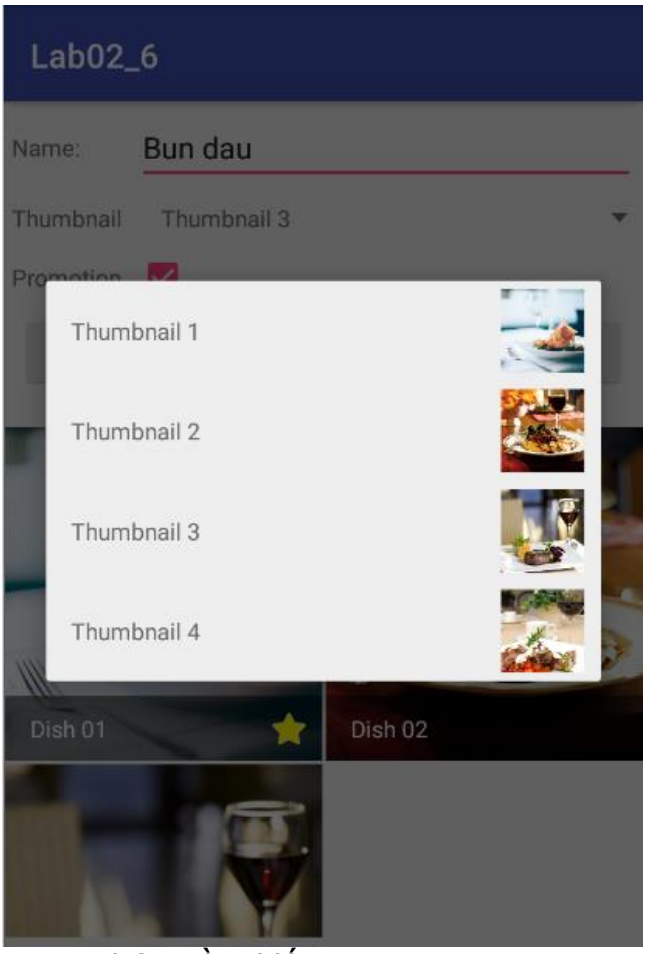
Các bước thiết đặt adapter để hiển thị nội dung lên gridview và spinner trong Android cũng tương tự trên listview. Sinh viên viết ứng dụng theo mô tả sau:



- Ứng dụng hỗ trợ chức năng thêm món ăn mới gồm các thông tin: tên món ăn, hình đại diện, và thông tin có khuyến mãi hay không. Sau khi thêm một món ăn, các trường name, thumbnail, promotion được reset về trạng thái ban đầu (name trống, thumbnail, promotion hiển thị giá trị mặc định). Thông báo "Added successfully" được hiển thị dưới dạng Toast.
- Danh sách các món ăn được hiển thị bằng gridview theo thiết kế như hình gồm 2 cột. Mỗi món ăn hiển thị hình đại diện, tên món ăn, nếu có khuyến mãi thì thêm icon star. Nếu tên món ăn quá dài, nội dung tên sẽ hiển thị trên gridview dạng chữ chạy.
- Hình đại diện được chọn từ spinner chứa 4 hình có sẵn. Danh sách này được hiển thị ở dạng dialog (không phải dạng dropdown sổ xuống thông thường) như hình minh họa gồm tên hình và hình. Khi một hình được chọn chỉ có tên được hiển thị lên spinner.

Hướng dẫn:

- Sinh viên download 4 hình đặt tên: first_thumnail.png, second_thumnail.png, third_thumnail.png, fourth_thumnail.png bỏ vào thư mục drawable.
- Định nghĩa enum chứa các loại thumbnails




```
public enum Thumbnail{
    Thumbnail1("Thumbnail 1", R.drawable.first_thumbnail),
    Thumbnail2("Thumbnail 2", R.drawable.second_thumbnail),
    Thumbnail3("Thumbnail 3",  R.drawable.third_thumbnail),
    Thumbnail4("Thumbnail 4",  R.drawable.fourth_thumbnail);

    private String name;
    private int img;

    Thumbnail(String name, int img) {
        this.name = name;
        this.img = img;
    }

    public String getName() { return name; }

    public int getImg() { return img; }
}
```

- Tạo class Dish có các thuộc tính và phương thức cần thiết.
- Tạo custom adapter cho gridview hiển thị các món ăn (item_dish.xml, DishAdapter.java .
- Tạo custom adapter cho spinner hiển thị danh sách thumbnails (item_thumbnail.xml, item_selected_thumbnail.xml, ThumbnailAdapter.java). Trong ThumbnailAdapter cần override các hàm:

  + getDropDownView() sử dụng view item_thumbnail.xml để hiện thị nội dung thumbnail trong dropdown (dạng dialog) chứa danh sách thumbnail cho người dùng chọn.

  + getView() sử dụng view item_selected_thumbnail.xml để hiển thị nội dung thumbnail được chọn lên spinner.

*** Lưu ý SV thêm một số nội dung:**

- Hướng dẫn sử dụng chức năng generate code tự động khi xây dựng class Dish.
- Các cách xử lý khi hiển thị tên dài trên textview: chữ chạy (marquee), ngắt chữ (end, start, middle), xuống dòng.
- Phân biệt chế độ VISIBLE, GONE, INVISIBLE của một view.
- Các chế độ co giãn hình ảnh trên imageview: centerCrop, fitXY, fitCenter…

6. Giới thiệu RecyclerView

GIỚI THIỆU

RecyclerView được cho là sự kế thừa của ListView và GridView. RecyclerView là một framework có thể mở rộng, và đặc biệt nó cung cấp khả năng triển khai cả bố cục Hozizontal và Vertical. Sử dụng RecyclerView khi mà data có các thành phần thay đổi trong quá trình chạy dựa trên hành động của người dùng hoặc các sự kiện mạng.

Sử dụng RecyclerView, sẽ làm việc với các thành phần sau:

- **RecyclerView.Adapter** : đây là nơi xử lý dữ liệu và gán dữ liệu lên các item của RecylerView.

  Khi tạo custom Adapter phải override lại 2 phương thức chính là:

  - **onCreateViewHolder**(): dùng để tạo View mới cho RecyclerView, nếu RecyclerView đã cached lại View thì phương thức này sẽ không được gọi.
  - **onBindViewHolder**(): dùng gắn dữ liệu vào View.

- **Layoutmanager**: xác định ra vị trí của các item trong RecyclerView. ListView chỉ hỗ trợ danh sách dạng cuộn dọc, RecylerView cung cấp RecyclerView.LayoutManager cho phép layout hiển thị item trong ListView theo các kiểu khác nhau (ngang, dọc, dạng lưới, dạng staggered grid- lưới so le).

  Các lớp con của LayoutManager:

  - **LinearLayoutManager**: hỗ trợ cuộn các item theo chiều ngang hay chiều dọc.
  - **GridLayoutManager**: layout các item trong RecyclerView dưới dạng Grid giống như khi sử dụng GridView.

- **StaggerdGridLayoutManager**: layout các item trong ListView dưới dạng lưới so le.

- **ItemAnimator**: Tạo hiệu ứng cho các hành động thêm, sửa, xóa các item ra khỏi Recycler 1 cách dễ dàng. Mặc định RecyclerView, sử dụng DefaultItemAnimator.
- **ViewHolder**: Để tái sử dụng View, nhằm tránh việc tạo View mới và findViewById quá nhiều.

SỬ DỤNG:

Bao gồm các bước:

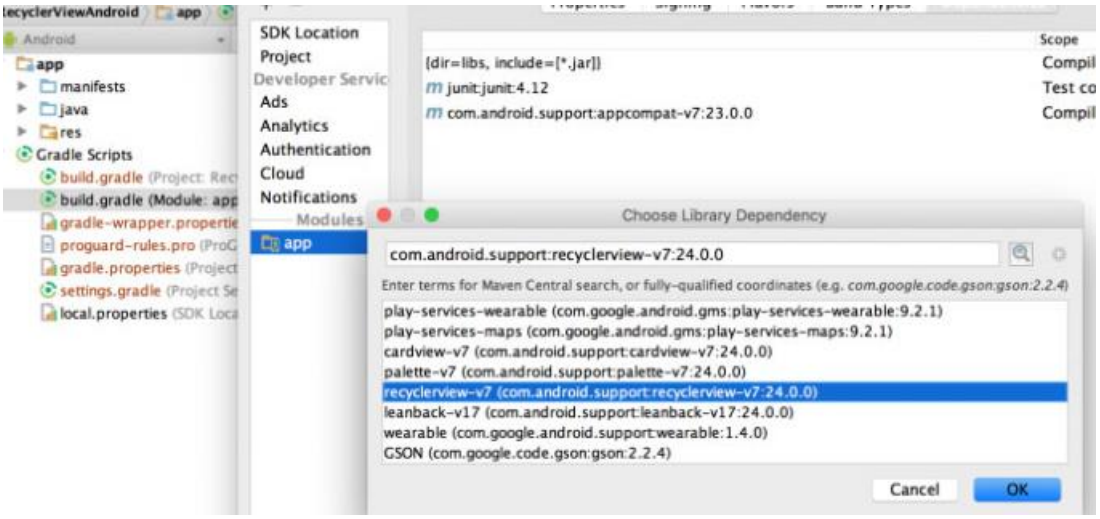
1. *Thêm thư viện*
2. *Tạo model lớp để chứa dữ liệu.*
3. *Thêm RecyclerView vào main_activity.xml.*
4. *Tạo giao diện cho 1 dòng.*
5. *Tạo custom Adapter và gán dữ liệu cho từng dòng trong Adapter.*
6. *Cài đặt RecyclerView trong MainActivity.java.*

Bước 1: Import thư viện

Mặc định RecyclerView không có sẵn trong Android SDK mà phải import vào thư viện. Chỉ cần dán dòng dưới đây vào *build.gradle* của *module app*, sau đó nhấn *Sync Now* để Android Studio tải và nạp thư viện tự động.

```
compile 'com.android.support:recyclerview-v7:23.0.0'
```

Cách khác là vào *File → Project Structure*, chọn *module app* và chuyển sang tab *Dependencies*, sau đó nhấn vào *dấu* + và chọn *Library Dependencies*. Tìm thư viện RecyclerView, sau đó nhấn OK để import thư viện.



Bước 2: Tạo model để chứa dữ liệu

Tạo class Hero có tên và hình đại diện

```
public class Hero {
    private String mName;
    private int mImage;

    public Hero(String mName, int mImage) {
        this.mName = mName;
        this.mImage = mImage;
    }

    public String getName() {
        return mName;
    }

    public void setName(String mName) {
        this.mName = mName;
    }

    public int getImage() {
        return mImage;
    }

    public void setImage(int mImage) {
        this.mImage = mImage;
    }
}
```

Bước 3: Thêm RecyclerView vào trong layout (main_activity.xml)

```
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recyclerHero"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

Bước 4: Tạo giao diện cho 1 dòng.

Định nghĩa ra XML layout file sử dụng cho mỗi row của danh sách. item layout file dưới đây chỉ chứa một ImageView hiển thị ảnh của Hero và một TextView hiển thị tên Hero.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <ImageView
        android:id="@+id/image_hero"
        android:layout_width="0dp"
        android:layout_height="@dimen/dp_180"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:scaleType="center"
```

```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/text_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="TextView"
        android:textColor="#FFF"
        app:layout_constraintBottom_toBottomOf="@+id/image_hero"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.112"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/image_hero"
        app:layout_constraintVertical_bias="1.0" />
</android.support.constraint.ConstraintLayout>
```

Bước 5: Tạo custom Adapter và gán dữ liệu cho từng dòng trong Adapter.

Vai trò của Adapter sẽ chuyển đổi một object tại một vị trí trở thành 1 hàng của danh sách sẽ được gắn vào RecyclerView.

```
public class HeroAdapter extends
RecyclerView.Adapter<HeroAdapter.ViewHolder> {
    private Context mContext;
    private ArrayList<Hero> mHeros;
// tạo ra các biến cho danh sách các Hero và truyền chúng qua hàm khởi tạo:
    public HeroAdapter(Context mContext, ArrayList<Hero> mHeros) {
        this.mContext = mContext;
        this.mHeros = mHeros;
    }
// Mọi Adapter sẽ có 3 phương thức quan trọng:
    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        LayoutInflater inflater = LayoutInflater.from(mContext);
```

```
        View heroView = inflater.inflate(R.layout.item_hero, parent,
false);
        ViewHolder viewHolder = new ViewHolder(heroView);
        return viewHolder;
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position)
    {
        Hero hero = mHeros.get(position);
        Glide.with(mContext)
                .load(hero.getImage())
                .into(holder.mImageHero);
        holder.mTextName.setText(hero.getName());
    }

    @Override
    public int getItemCount() {
        return mHeros.size();
    }
// Tạo class viewholder
    public class ViewHolder extends RecyclerView.ViewHolder {
        private ImageView mImageHero;
        private TextView mTextName;

        public ViewHolder(@NonNull View itemView) {
            super(itemView);
            mImageHero = itemView.findViewById(R.id.image_hero);
            mTextName = itemView.findViewById(R.id.text_name);
        }
    }

}
```

Bước 6: Cài đặt RecyclerView trong MainActivity.java

```
public class HeroActivity extends AppCompatActivity {
    private ArrayList<Hero> mHeros ;
    private RecyclerView mRecyclerHero;
    private HeroAdapter mHeroAdapter ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_hero);
        mRecyclerHero = findViewById(R.id.recyclerHero);
```

```
        mHeros = new ArrayList<>();
        createHeroList();
        mHeroAdapter = new HeroAdapter(this,mHeros);
        mRecyclerHero.setAdapter(mHeroAdapter);
        mRecyclerHero.setLayoutManager(new LinearLayoutManager(this));
    }

    private void createHeroList() {
        mHeros.add(new Hero("Thor",R.drawable.image_thor));
        mHeros.add(new Hero("IronMan",R.drawable.image_ironman));
        mHeros.add(new Hero("Hulk",R.drawable.image_hulk));
        mHeros.add(new Hero("SpiderMan",R.drawable.image_spiderman));
        mHeros.add(new Hero("Thor",R.drawable.image_thor));
        mHeros.add(new Hero("IronMan",R.drawable.image_ironman));
        mHeros.add(new Hero("Hulk",R.drawable.image_hulk));
        mHeros.add(new Hero("SpiderMan",R.drawable.image_spiderman));
        mHeros.add(new Hero("Thor",R.drawable.image_thor));
        mHeros.add(new Hero("IronMan",R.drawable.image_ironman));
    }
}
```

Kết quả chương trình:

7. Bài tập RecyclerView

Làm lại bài ở mục **Trường hợp 4: Sử dụng CustomAdapter cho Listview)** bằng **RecyclerView**

IV. THAM KHẢO THÊM (Tự xem thêm)

1. QuickAdapter

Tham khảo tại: https://github.com/JoanZapata/base-adapter-helper

2. Glide

Tham khảo tại: https://github.com/bumptech/glide

3. Gson

Tham khảo tại: https://github.com/google/gson