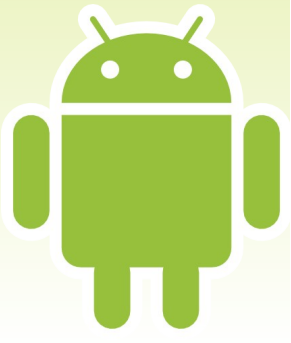
# Android Developer Training

# Image Loaders





# What You Are Looking For

- Flexible Sources
  - Network
  - Local Files
  - MediaStore
  - Other Provider Uri Values
    - e.g., contact photos
  - Pluggable Sources



# What You Are Looking For

- Flexible Targets
  - ImageView
  - Plain Bitmap
    - E.g., Notifications
  - Pluggable Targets
    - E.g., Leanback ImageCardView



# What You Are Looking For

- Intelligent Bitmap Handling
  - Pluggable and/or Resizeable Cache
  - inBitmap Support
  - Recycling Support
    - e.g., ListView rows



# What You Are Looking For

- Other Features
  - SSL Support
    - Including custom TrustManager[]
  - Configurable/Shareable Thread Pool
  - Two-Level Caching
    - RAM
    - Disk
  - Clean API



# Image Loading Libraries

- Popular
  - Picasso
  - Universal Image Loader
- Controversial
  - Fresco
- Others
  - https://android-arsenal.com/tag/46



```
{
  "items": [
    {
      "tags": [
        "android",
        "parse.com"
      ],
      "owner": {
        "reputation": 8,
        "user_id": 3834761,
        "user_type": "registered",
        "accept_rate": 100,
        "profile_image": "https://www.gravatar.com/avatar/ffb9e28e492c35033337788e3fb0d583?s=1
        "display_name": "user3834761",
        "link": "http://stackoverflow.com/users/3834761/user3834761"
      },
      "is_answered": false,
      "view_count": 15,
      "answer_count": 1,
      "score": 0,
      "last_activity_date": 1436208937,
      "creation_date": 1435981054,
      "question_id": 31216642,
      "link": "http://stackoverflow.com/questions/31216642/parsequery-for-profile-picture-whil
      "title": "ParseQuery for profile picture while displaying post"
    },
```

```
dependencies {
    compile 'de.greenrobot:eventbus:2.4.0'
    compile 'com.squareup.picasso:picasso:2.5.2'
    compile 'com.squareup.retrofit:retrofit:1.9.0'
}
```

```
public class SOQuestions {
    List<Item> items;
}
```

```java
public class Item {
   String title;
   Owner owner;
   String link;

   @Override
   public String toString() { return(title); }
}
```

```java
import com.google.gson.annotations.SerializedName;

public class Owner {
   @SerializedName("profile_image") String profileImage;
}
```

```java
public interface StackOverflowInterface {
  @GET("/2.1/questions?order=desc&sort=creation&site=stackoverflow")
  void questions(@Query("tagged") String tags, Callback<SOQuestions> cb);
}
```

```java
@Override
public View onCreateView(LayoutInflater inflater,
                         ViewGroup container,
                         Bundle savedInstanceState) {
  View result=
      super.onCreateView(inflater, container, savedInstanceState);

  setRetainInstance(true);

  RestAdapter restAdapter=
      new RestAdapter.Builder().setEndpoint("https://api.stackexchange.com")
                               .build();
  StackOverflowInterface so=
      restAdapter.create(StackOverflowInterface.class);

  so.questions("android", this);

  return(result);
}
```

```java
@Override
public void failure(RetrofitError exception) {
  Toast.makeText(getActivity(), exception.getMessage(),
                 Toast.LENGTH_LONG).show();
  Log.e(getClass().getSimpleName(),
        "Exception from Retrofit request to StackOverflow", exception);
}

@Override
public void success(SOQuestions questions, Response response) {
  setListAdapter(new ItemsAdapter(questions.items));
}
```

```
class ItemsAdapter extends ArrayAdapter<Item> {
  ItemsAdapter(List<Item> items) {
    super(getActivity(), R.layout.row, R.id.title, items);
  }

  @Override
  public View getView(int position, View convertView, ViewGroup parent) {
    View row=super.getView(position, convertView, parent);
    Item item=getItem(position);
    ImageView icon=(ImageView)row.findViewById(R.id.icon);

    Picasso.with(getActivity()).load(item.owner.profileImage)
           .fit().centerCrop()
           .placeholder(R.drawable.owner_placeholder)
           .error(R.drawable.owner_error).into(icon);

    TextView title=(TextView)row.findViewById(R.id.title);

    title.setText(Html.fromHtml(getItem(position).title));

    return(row);
  }
}
```

```
dependencies {
    compile 'com.nostra13.universalimageloader:universal-image-loader:1.9.3'
}
```

```
public class VideosFragment extends
    ContractListFragment<VideosFragment.Contract> implements
    LoaderManager.LoaderCallbacks<Cursor>,
    SimpleCursorAdapter.ViewBinder {
  private ImageLoader imageLoader;

  @Override
  public void onAttach(Activity host) {
    super.onAttach(host);

    ImageLoaderConfiguration ilConfig=
        new ImageLoaderConfiguration.Builder(getActivity()).build();

    imageLoader=ImageLoader.getInstance();
    imageLoader.init(ilConfig);
  }
```

```java
@Override
public void onActivityCreated(Bundle state) {
  super.onActivityCreated(state);

  String[] from=
      { MediaStore.Video.Media.TITLE, MediaStore.Video.Media._ID };
  int[] to= { android.R.id.text1, R.id.thumbnail };
  SimpleCursorAdapter adapter=
      new SimpleCursorAdapter(getActivity(), R.layout.row, null,
                              from, to, 0);

  adapter.setViewBinder(this);
  setListAdapter(adapter);

  getLoaderManager().initLoader(0, null, this);
}
```

```
@Override
public Loader<Cursor> onCreateLoader(int arg0, Bundle arg1) {
  return(new CursorLoader(
                          getActivity(),
                          MediaStore.Video.Media.EXTERNAL_CONTENT_URI,
                          null, null, null,
                          MediaStore.Video.Media.TITLE));
}

@Override
public void onLoadFinished(Loader<Cursor> loader, Cursor c) {
  ((CursorAdapter)getListAdapter()).swapCursor(c);
}

@Override
public void onLoaderReset(Loader<Cursor> loader) {
  ((CursorAdapter)getListAdapter()).swapCursor(null);
}
```

```
@Override
public boolean setViewValue(View v, Cursor c, int column) {
  if (column == c.getColumnIndex(MediaStore.Video.Media._ID)) {
    Uri video=
        ContentUris.withAppendedId(MediaStore.Video.Media.EXTERNAL_CONTENT_URI,
            c.getInt(column));
    DisplayImageOptions opts=new DisplayImageOptions.Builder()
        .showImageOnLoading(R.drawable.ic_media_video_poster)
        .build();

    imageLoader.displayImage(video.toString(), (ImageView)v, opts);

    return(true);
  }

  return(false);
}
```