Picasso

Easy Image Loading on Android

Norman Peitek & Marcus Pöhls

Picasso: Easy Image Loading on Android

The Fast Approach to Build Image-Rich Apps

Norman Peitek

This book is for sale at http://leanpub.com/picasso-image-loading-on-android

This version was published on 2019-02-17

Tweet This Book!

Please help Norman Peitek by spreading the word about this book on Twitter!

The suggested hashtag for this book is #PicassoBook.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#PicassoBook

Contents

About the Book

Due to the popularity of the Picasso blog post series published on the Future Studio blog, and the positive feedback on our Retrofit book, we've decided to publish a book on Picasso. If your Android app uses images, this book will save you a ton of time researching and avoid stressful evenings of bug fixes. If you value your time, this might be something for you.

We cover all topics from the blog post series and additionally add more explanations to each topic and the example code snippets. Besides a more coherent introduction to Picasso, you'll also benefit from more new, book-exclusive advanced topics.

This book is for beginners and advanced readers as well. We'll walk you through each topic with direct reference to code examples. Once you've worked through this book, you'll have an extensive knowledge of image loading on Android with Picasso.

What Topics are Covered in this Book?

The list below provides a comprehensive overview of covered topics within the book.

- Introduction to Picasso
- Loading and Displaying Images
- Placeholders, Resizing and Optimized Request Management
- Image Rotation and Transformations
- Caches are Critical
- Customizing Picasso
- Practical Code Examples
- App Release Preparations

Who Is This Book For?

This book is for Android developers who want to get a substantial overview and reference book of Picasso. You'll benefit from the clearly recognizable code examples in regard to your daily work with Picasso.

Rookie

If you're just starting out with Picasso (or coming from any other image loading library like Glide) this book will show you all important parts on how to work with images. The provided code snippets let you jumpstart and create an image-rich app within minutes.

Expert

You already worked with Picasso before? You’ll profit from our extensive code snippets and can improve your existing code base. Additionally, the book illustrates various optimizations for an even better user experience.

Chapter 1 — Image Loading

We had a lot of success and feedback on our retrofit[1] series, so we decided to do another series on one of our favorite Square library[2]: Picasso.

What is Picasso?

Picasso[3] can load and display images from many sources, while also taking care of caching and keeping a low memory impact when doing image manipulations. Picasso's advantages are an extremely slick API approach, which makes the use for developers very easy.

Why Use Picasso?

Experienced Android developers can skip this section, but for the starters: you might ask yourself why you want to use Picasso<sup>*</sup> instead of your own implementation.

Android is quite the diva when working with images, since it'll load images into the memory pixel by pixel[4]. A single photo of an average phone camera with the dimensions of 2592x1936 pixels (5 megapixels) will allocate around 19 MB of memory. If you add the complexity of network requests on spotty wireless connections, caching and image manipulations, you will safe yourself a lot of time & headache, if you use a well-tested and developed library like Picasso.

In this series, we'll look at many of the features of Picasso. Just take a peek at the table of contents and think for a minute if you really want to develop all of these features yourself.

<sup>*</sup> = *or any other image loading library, like Glide, ION, etc.*

Adding Picasso to Your Setup

Hopefully by now we've convinced you to use a library to handle your image loading requests. If you want to take a learn in-depth about Picasso, this is the guide for you!

First things first, add Picasso to your dependencies. At the time of writing, the last version of Picasso is `2.5.2`.

Gradle

As with most dependencies, pulling it in a Gradle project is a single line in your `build.gradle`:

[1] http://futurestud.io/blog/retrofit-series-round-up/
[2] http://square.github.io/
[3] http://square.github.io/picasso/
[4] http://developer.android.com/training/displaying-bitmaps/index.html

```
compile 'com.squareup.picasso:picasso:2.5.2'
```

Maven

While we moved all our projects to Gradle, Picasso also supports Maven projects:

```
<dependency>
    <groupId>com.squareup.picasso</groupId>
    <artifactId>picasso</artifactId>
    <version>2.5.2</version>
</dependency>
```

Example Project

Bundled with this book comes an example project. The example project includes all functionalities shown in this book. If you want to see the theoretical knowledge in the book in action, go ahead and download the project. The code and the compiled app should give you an additional insight.

Basic Usage: Loading Image from a URL

The Picasso library is using a fluent interface[5], specifically implemented with the Picasso class[6]. The Picasso class requires at least three parameters for a fully functional request:

- `with(Context context)` - Context[7] is necessary for many Android API calls. Picasso is no difference here.
- `load(String imageUrl)` - here you specify which image should be loaded. Mostly it'll be a String representing a URL to an Internet image.
- `into(ImageView targetImageView)` - the target ImageView your image is supposed to get displayed in.

Theoretical explanations are always harder to grasp, so let's look at a hands-on example:

[5]http://en.wikipedia.org/wiki/Fluent_interface
[6]http://square.github.io/picasso/javadoc/com/squareup/picasso/Picasso.html
[7]http://developer.android.com/reference/android/content/Context.html

```java
ImageView targetImageView = (ImageView) findViewById(R.id.imageView);
String internetUrl = "http://i.imgur.com/DvpvklR.png";

Picasso
    .with(context)
    .load(internetUrl)
    .into(targetImageView);
```

That's it! If the image at the URL exists and your `ImageView` is visible, you'll see the image in a few seconds. In case the image doesn't exist, Picasso will return to the error callbacks, which we'll look at in a later chapter. You might already be convinced with this three-line example that Picasso is useful to you, but this is just the tip of the feature iceberg.

A very helpful feature is the integration into Android's lifecycle. For example, when the Android activity is paused, Picasso will automatically pause the image request. When the app returns from the background, it'll resume the request and load the image. Awesome!

Advanced Loading: Loading from Various Sources

We've just seen how to load an image from an Internet source. But this isn't the only possible image source for Picasso. Picasso can also load images from the Android resources, files and Uri[8]s. In this section, we'll cover all three options.

Loading from Resources

First up is loading from Android resources. Instead of giving a `String` pointing to an Internet URL, you give a resource `int`.

```java
int resourceId = R.mipmap.ic_launcher;

Picasso
    .with(context)
    .load(resourceId)
    .into(targetImageView);
```

If you're confused by the *R.mipmap.*, it's Android's new way[9] of handling icons.

Loading from File

Second up is loading from a file. This can be useful when you let the user select a photo to display an image (similar to a gallery). The parameter is just a `File` object.

[8]http://developer.android.com/reference/android/net/Uri.html
[9]http://android-developers.blogspot.de/2014/10/getting-your-apps-ready-for-nexus-6-and.html

```java
// this file probably does not exist on your device.
// However, you can use any file path, which points to an image file

File file = new File(
    Environment.getExternalStoragePublicDirectory(
        Environment.DIRECTORY_PICTURES
    ),
    "Running.jpg"
);

Picasso
    .with(context)
    .load(file)
    .into(targetImageView);
```

Loading from Uri

Lastly, you can also load images defined by an `Uri`. The request is no different from the previous options:

```java
// this could be any Uri. for demonstration purposes we're just creating
// an Uri pointing to a launcher icon

Uri uri = resourceIdToUri(context, R.mipmap.future_studio_launcher);

Picasso
    .with(context)
    .load(uri)
    .into(targetImageView);
```

The small helper function you see above is a simple conversion from the `resourceId` to an `Uri`.

```java
public static final String ANDROID_RESOURCE = "android.resource://";
public static final String FOREWARD_SLASH = "/";

private static Uri resourceIdToUri(Context context, int resourceId) {
    return Uri.parse(ANDROID_RESOURCE + context.getPackageName()
    + FOREWARD_SLASH + resourceId);
}
```

However, the `Uri` does not have to be generated from a resourceId. It can be any `Uri`.

Chapter Summary

The basic loading principles are done. In this chapter, you should have learned:

- [x] Why to use Picasso
- [x] How to integrate Picasso into your project
- [x] How to load images from various resources

Next, we can finally look at more interesting stuff. In the next chapter, we'll cover how you can change the display of images with Picasso. We'll show you how to manipulate the images before displaying them. We also explain how to load images into other destinations, besides `ImageViews`.

Additional Chapter Resources

- Picasso Project Homepage[10]
- Picasso Java docs[11]
- Picasso on Github[12]
- Android Studio IDE[13]

[10]http://square.github.io/picasso/
[11]http://square.github.io/picasso/javadoc/index.html
[12]https://github.com/square/picasso
[13]https://developer.android.com/sdk/index.html

Outro

Our goal is to truly help you getting started and improve your knowledge around Picasso. We hope you learned many new things throughout this book. We want you to save time while learning the basics and in-depth details about Picasso. We think the existing Picasso documentation lacks various information and this book should help you to gain extensive in-depth knowledge without loosing time searching StackOverflow for correct answers.

Did we miss something in this book? Do you've any feedback for us on your mind? Please take a minute to write us an email[14] to give us a chance to improve the book. Thank you!

This book based on Picasso `2.5.2`. The development team usually isn't afraid to make big changes, if necessary. Currently, it looks like the version is stable. However, if the team releases significant updates for Picasso, we'll update the book accordingly. Please keep in mind though, that it might take some time to integrate all the changes. Of course, everyone who bought this book will get the book update **for free**.

In the meantime, feel free to visit our homepage[15] and blog[16]. We're publishing new valuable posts every Monday and Thursday about Android, Node.js and various open source tools.

Thanks a lot for reading this book! We highly appreciate your interest and hope you learned a lot from this book! <3

[14] mailto:info@futurestud.io
[15] https://futurestud.io
[16] https://futurestud.io/blog