*PICASSO*

Images add much-needed context and visual flair to Android applications. Picasso allows for hassle-free image loading in your application—often in one line of code!

Picasso.with(context).load("http://i.imgur.com/DvpvklR.png").into(imageView);

Many common pitfalls of image loading on Android are handled automatically by Picasso:

* Handling ImageView recycling and download cancelation in an adapter.
* Complex image transformations with minimal memory use.
* Automatic memory and disk caching.



*Features*

ADAPTER DOWNLOADS

Adapter re-use is automatically detected and the previous download canceled.

@Override public void getView(int position, View convertView, ViewGroup parent) {

SquaredImageView view = (SquaredImageView) convertView;

if (view == null) {

view = new SquaredImageView(context);

}

String url = getItem(position);

Picasso.with(context).load(url).into(view);

}

IMAGE TRANSFORMATIONS

Transform images to better fit into layouts and to reduce memory size.

Picasso.with(context)

.load(url)

.resize(50, 50)

.centerCrop()

.into(imageView)

You can also specify custom transformations for more advanced effects.

public class CropSquareTransformation implements Transformation {

@Override public Bitmap transform(Bitmap source) {

int size = Math.min(source.getWidth(), source.getHeight());

int x = (source.getWidth() - size) / 2;

int y = (source.getHeight() - size) / 2;

Bitmap result = Bitmap.createBitmap(source, x, y, size, size);

if (result != source) {

source.recycle();

}

return result;

}

@Override public String key() { return "square()"; }

}

Pass an instance of this class to the transform method.

PLACE HOLDERS

Picasso supports both download and error placeholders as optional features.

Picasso.with(context)

.load(url)

.placeholder(R.drawable.user\_placeholder)

.error(R.drawable.user\_placeholder\_error)

.into(imageView);

A request will be retried three times before the error placeholder is shown.

RESOURCE LOADING

Resources, assets, files, content providers are all supported as image sources.

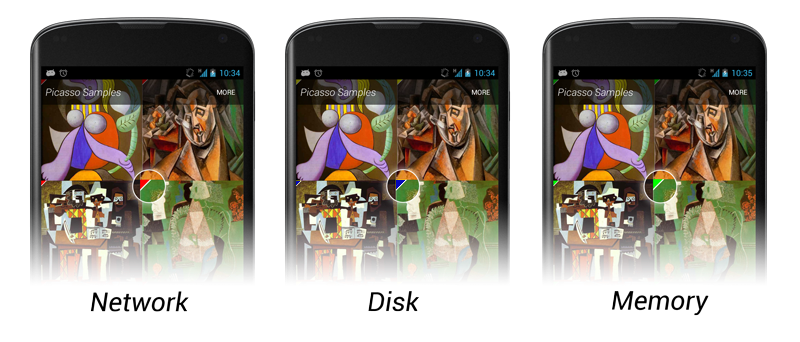
Picasso.with(context).load(R.drawable.landing\_screen).into(imageView1);

Picasso.with(context).load("file:///android\_asset/DvpvklR.png").into(imageView2);

Picasso.with(context).load(new File(...)).into(imageView3);

DEBUG INDICATORS

For development you can enable the display of a colored ribbon which indicates the image source. Call setIndicatorsEnabled(true)on the Picasso instance.



*Download*

[↓ v2.5.2 JAR](http://repo1.maven.org/maven2/com/squareup/picasso/picasso/2.5.2/picasso-2.5.2.jar)

The source code to the Picasso, its samples, and this website is [available on GitHub](http://github.com/square/picasso).

MAVEN

<dependency>

<groupId>com.squareup.picasso</groupId>

<artifactId>picasso</artifactId>

<version>2.5.2</version>

</dependency>

GRADLE

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

*Contributing*

If you would like to contribute code you can do so through GitHub by forking the repository and sending a pull request.

When submitting code, please make every effort to follow existing conventions and style in order to keep the code as readable as possible. Please also make sure your code compiles by running mvn clean verify.

Before your code can be accepted into the project you must also sign the [Individual Contributor License Agreement (CLA)](http://squ.re/sign-the-cla).

*License*

Copyright 2013 Square, Inc.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.