OpenCV

Saumitro Dasgupta

CS231M • Stanford University

Roadmap

- Introduction to OpenCV
- Basic OpenCV datatypes
- Accessing your device's camera
- Realtime image processing
- Using JNI and Android NDK
- Native OpenCV
- Further resources

OpenCV

- Open source computer vision library
- Available on all major platforms
  - Android, iOS, Linux, Mac OS X, Windows…
- Written primarily in C++
  - Bindings available for Java, Python…
- Well documented at http://docs.opencv.org
- Source available at https://github.com/Itseez/opencv

What can it do?

Image Processing	Filters, Histograms, Morphology, Color Ops...
Feature Detection	Edges, Corners, Lines, Circles, SIFT, SURF, ORB...
Object Detection	Haar, Latent SVM, Template Matching...
Machine Learning	SVM, Bayes, Decision Trees, Neural Networks, Clustering, Boosting...
Motion Tracking	Optical flow, Kalman Filters, MeanShift...
Camera Calibration	Homography, Fundamental Matrix...
Your Homework	Project 0, Project 1, Project 2...

Matrices in OpenCV

The Mat class represents a fixed type n-dimensional dense matrix

```
// Create a 100x100 matrix of doubles (64-bit floats)
Mat M(100, 100, CV_64F);
```

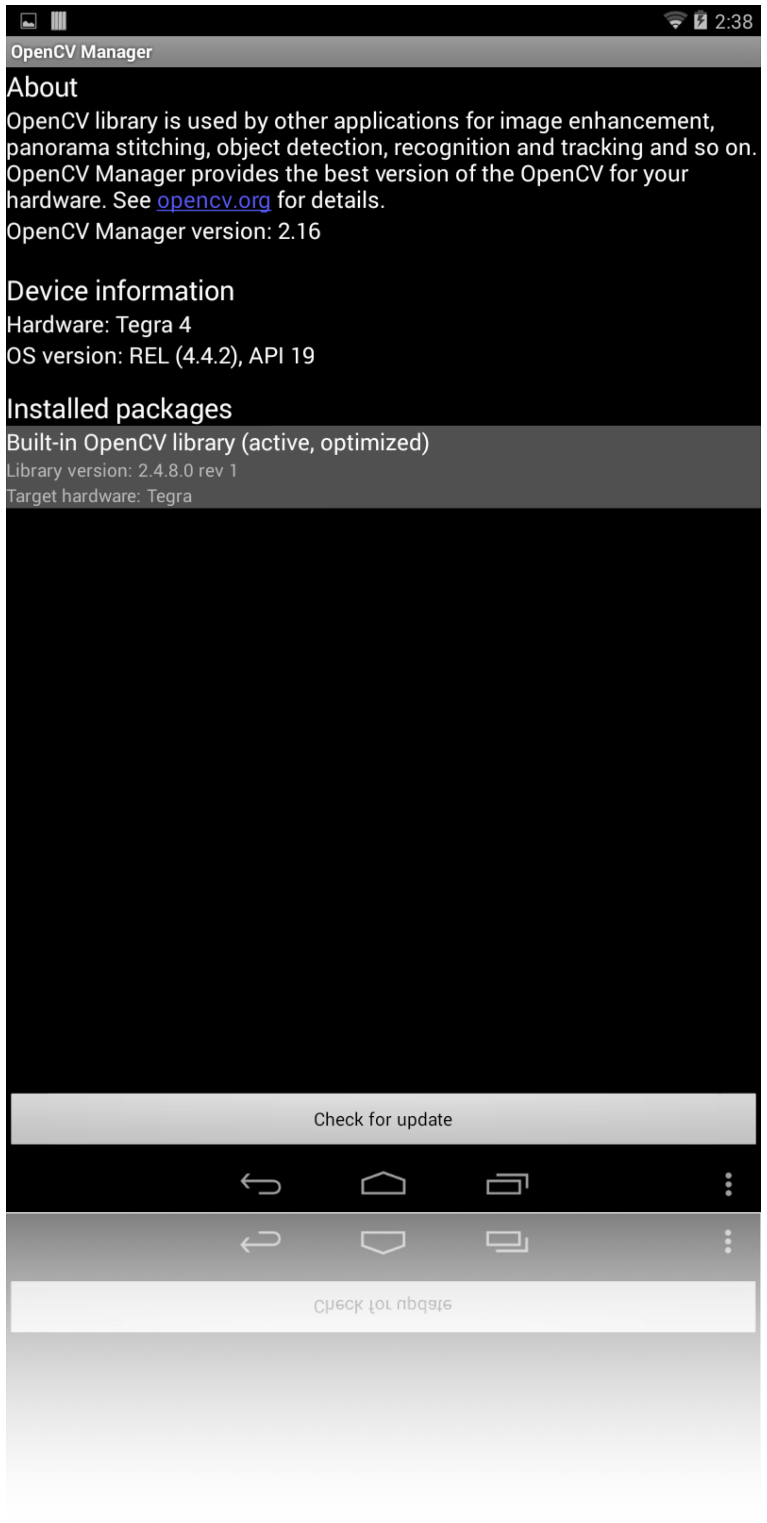
Automatic memory management

```
// M2 and M1 share the same data
Mat M2 = M;
// R also shares the same data
Mat R = M2.row(10);
// M3 references a separate copy of the data
Mat M3 = M.clone();
```

Quick Reference

http://docs.opencv.org/trunk/opencv_cheatsheet.pdf

OpenCV on Android



OpenCV Manager

About

OpenCV library is used by other applications for image enhancement, panorama stitching, object detection, recognition and tracking and so on. OpenCV Manager provides the best version of the OpenCV for your hardware. See opencv.org for details.

OpenCV Manager version: 2.16

Device information

Hardware: Tegra 4

OS version: REL (4.4.2), API 19

Installed packages

Built-in OpenCV library (active, optimized)

Library version: 2.4.8.0 rev 1

Target hardware: Tegra

Install OpenCV Manager

Common OpenCV library shared by apps.

Uses optimized built-in version.

OpenCV on iOS

Option 1: Pre-built framework

http://sourceforge.net/projects/opencvlibrary/files/opencv-ios/

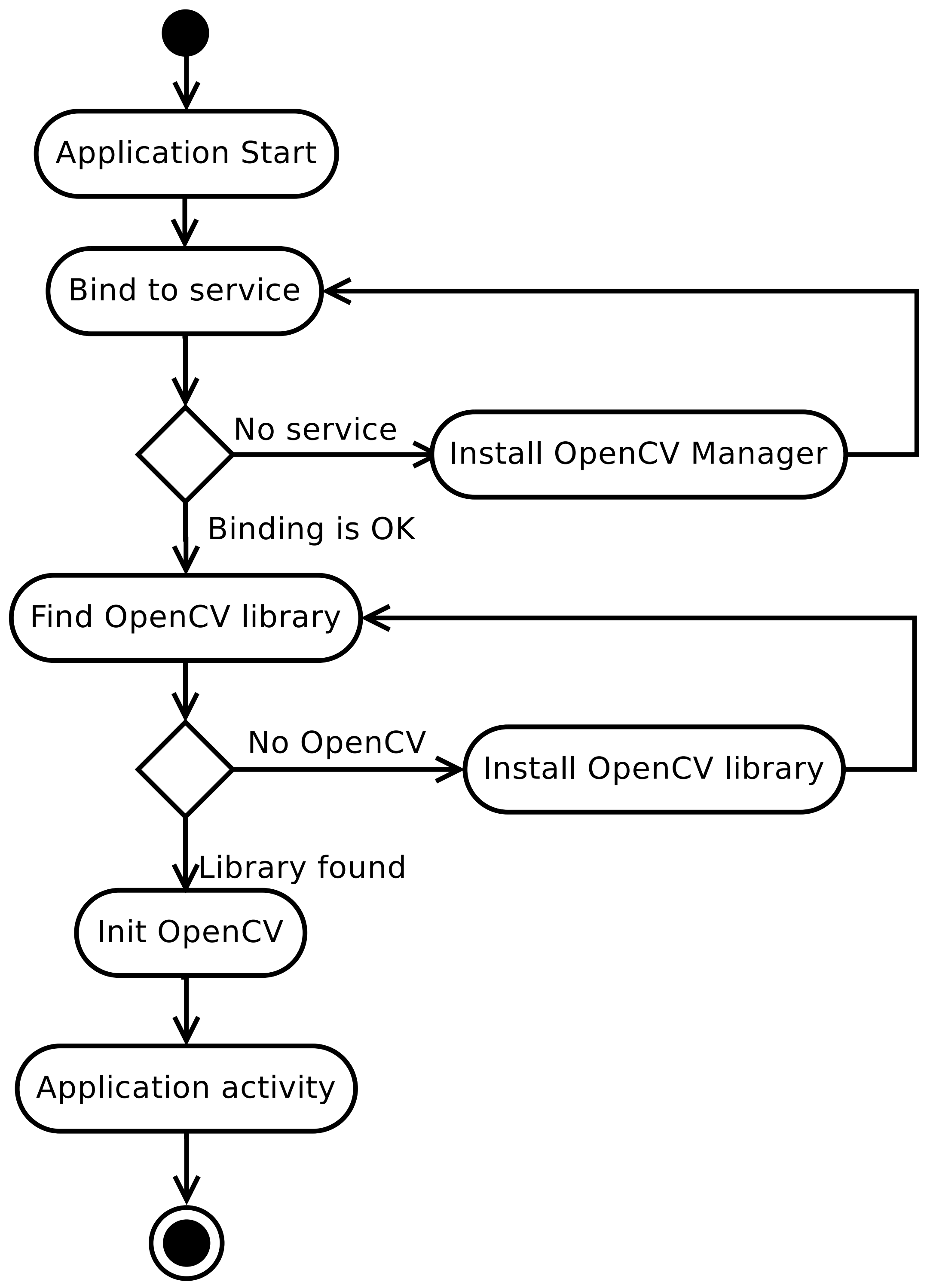
Option 2: Build from Source

http://docs.opencv.org/doc/tutorials/introduction/ios_install/ios_install.html

Live Code

OpenCV Loader Mechanism

Image Attribution:
The OpenCV Dev Team



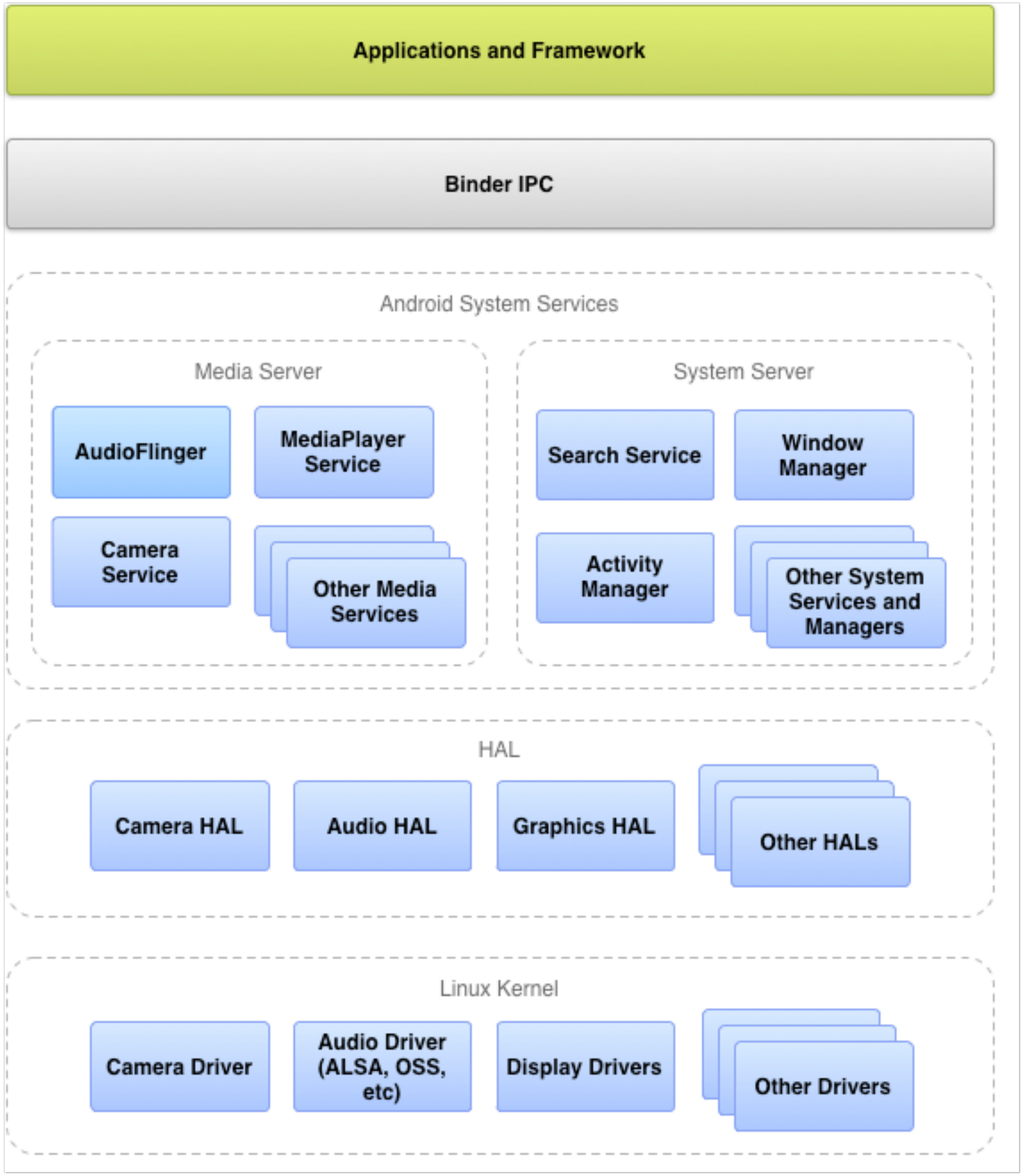
Android NDK

+

Java Native Interface

Android System Architecture

Image Attribution:
The Android Open Source Project



Before downloading the NDK, you should understand that **the NDK will not benefit most apps**.

Android NDK webpage
https://developer.android.com/tools/sdk/ndk/index.html

We should forget about small efficiencies, say about 97% of the time: **premature optimization is the root of all evil.**

Donald Knuth
ACM Computing Surveys, Vol 6, No. 4,December 1974

We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil. **Yet we should not pass up our opportunities in that critical 3%.**

Donald Knuth
ACM Computing Surveys, Vol 6, No. 4,December 1974

Before downloading the NDK, you should understand that the NDK will not benefit most apps. **As a developer, you need to balance its benefits against its drawbacks.** … In general, **you should only use the NDK if it is essential to your app**—never because you simply prefer to program in C/C++.

Android NDK webpage
https://developer.android.com/tools/sdk/ndk/index.html

In Java

Declare the native function signature:

```
native int factorial(int n);
```

Load the native library:

```
static
{
    System.loadLibrary("factorial");
}
```

Call it like a Java function:

```
System.out.println("Result: " + factorial(10));
```

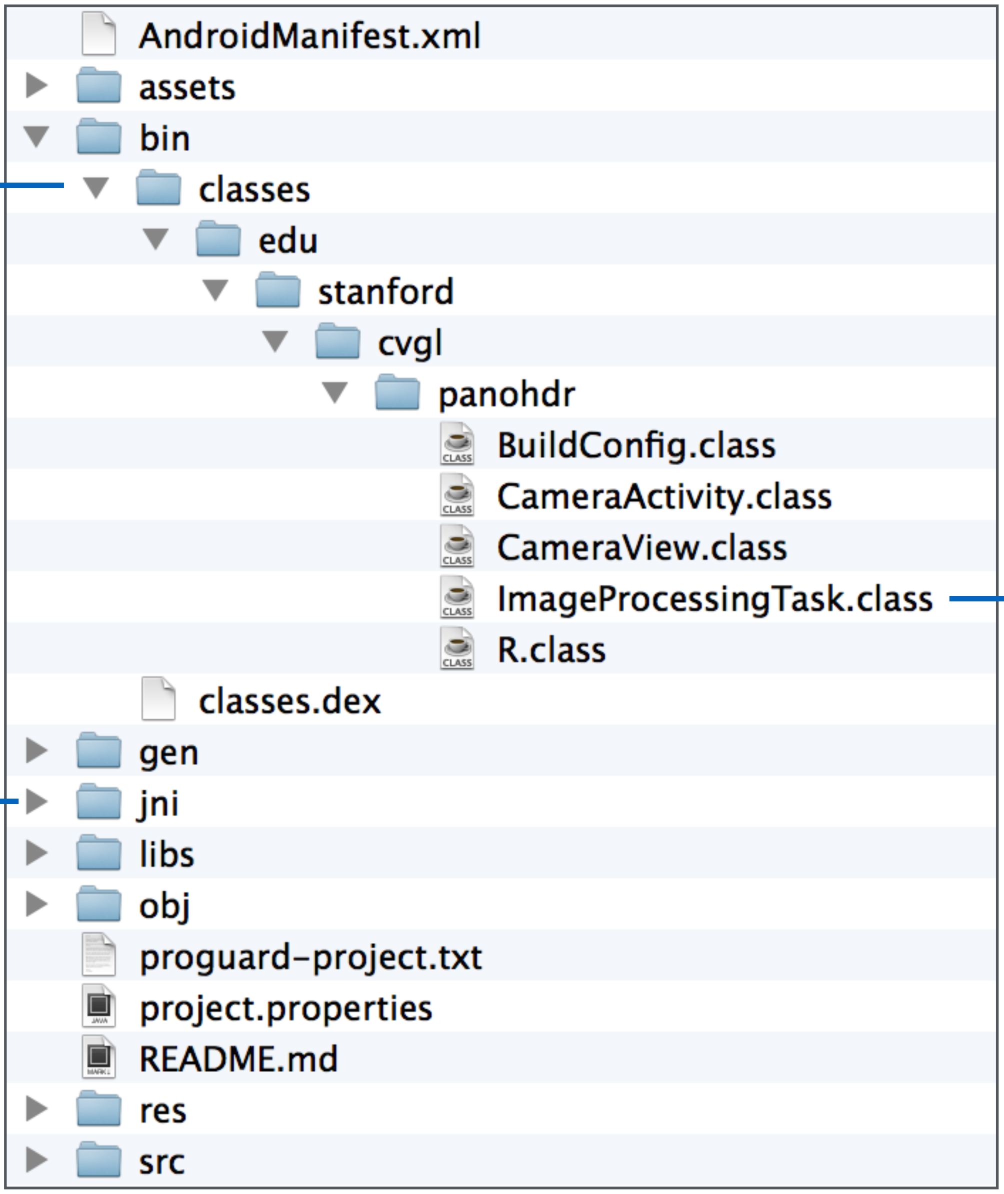
In C++

```
#include <jni.h>

extern "C" JNIEXPORT jint JNICALL
Java_edu_stanford_nativedemo_FactorialActivity_factorial
  (JNIEnv* env, jobject obj, jint n)
{
    int s=1;
    for(int i=1; i<=n; ++i)
    {
        s *= i;
    }
    return s;
}
```

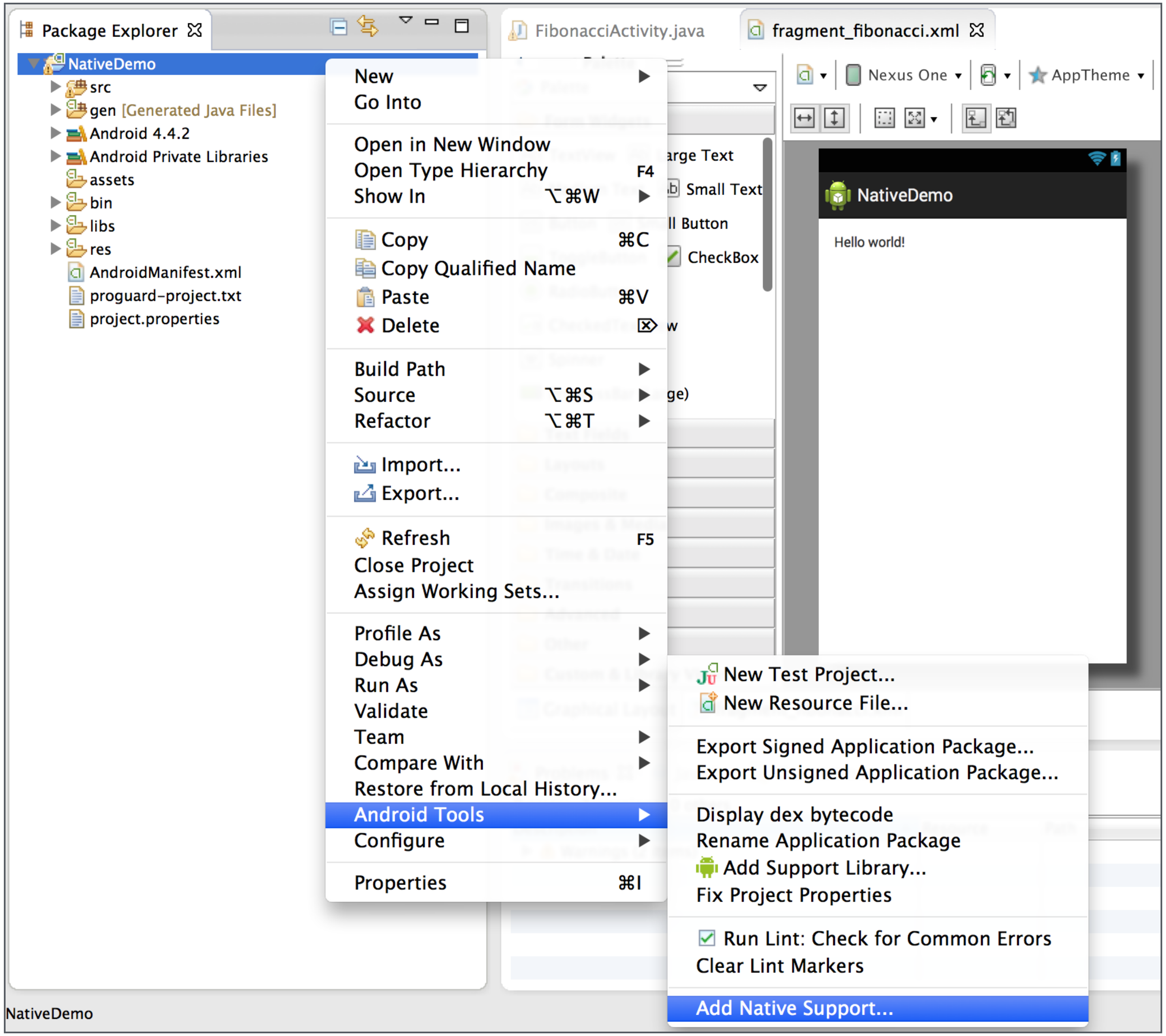
Auto-generate JNI header

```
$ javah -d jni -classpath bin\classes edu.stanford.cvgl.panohdr.ImageProcessingTask
```



```
AndroidManifest.xml
▶ assets
▼ bin
  ▼ classes
    ▼ edu
      ▼ stanford
        ▼ cvgl
          ▼ panohdr
              BuildConfig.class
              CameraActivity.class
              CameraView.class
              ImageProcessingTask.class
              R.class
    classes.dex
▶ gen
▶ jni
▶ libs
▶ obj
  proguard-project.txt
  project.properties
  README.md
▶ res
▶ src
```

Add Native Support



Android.mk

Module specific makefile

```
LOCAL_PATH := $(call my-dir)

include $(CLEAR_VARS)

LOCAL_MODULE    := factorial
LOCAL_SRC_FILES := factorial.cpp

include $(BUILD_SHARED_LIBRARY)
```

Application.mk

Optional Application-wide makefile

```
APP_PLATFORM := android-19
APP_ABI := armeabi-v7a
APP_STL := gnustl_static
APP_CPPFLAGS := -frtti -fexceptions -std=c++11
```

Beyond OpenCV

Android

RenderScript
http://developer.android.com/guide/topics/renderscript

FastCV:
https://developer.qualcomm.com/mobile-development/add-advanced-features/computer-vision-fastcv

iOS

GPUImage
https://github.com/BradLarson/GPUImage

Accelerate Framework + Core Image
https://developer.apple.com/library/ios/navigation/