How to combine Swift/Kotlin with C/C++

Michał Kowalczyk, TomTom

Why C++???

Dangerous features

Exceptions

We do not use C++ exceptions.

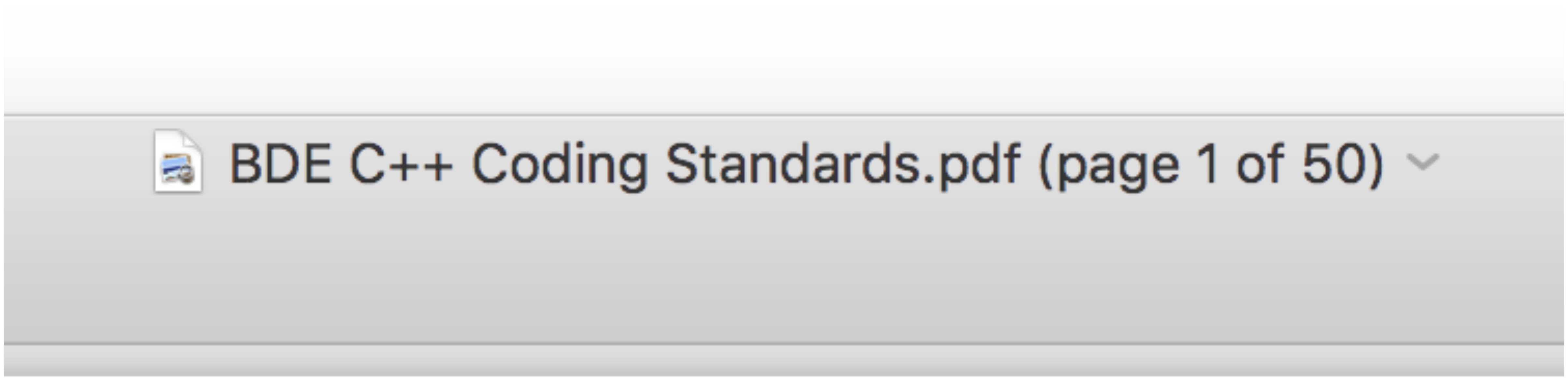
https://google.github.io/styleguide/cppguide.html

Dangerous features

Instead of relying on operator precedence and associativity, add clarifying parentheses around each pair of operands to make expressions unambiguous.

```
a = b * c * d; // ok

a = b * c + d; // not ok

a = (b * c) + d; // ok and unambiguous what
the writer wanted
```

Dangerous features


BDE C++ Coding Standards.pdf (page 1 of 50)

https://github.com/bloomberg/bde/wiki/Introduction-to-BDE-Coding-Standards

Why C++??????

TomTom & C++

TomTom’s clients

- Android
- iOS
- Custom (usually C++)

One app - many langs

- A lot of code duplication
- Expensive maintenance

Language bindings

- C++ -> Objective-C++ -> Objective-C
- C++ + Java Native Interface -> Java

How about Kotlin/Swift

- C++ -> Objective-C++ + Bridging header -> Swift
- C++ + Java Native Interface -> Java -> Kotlin

Example

Push-ups!

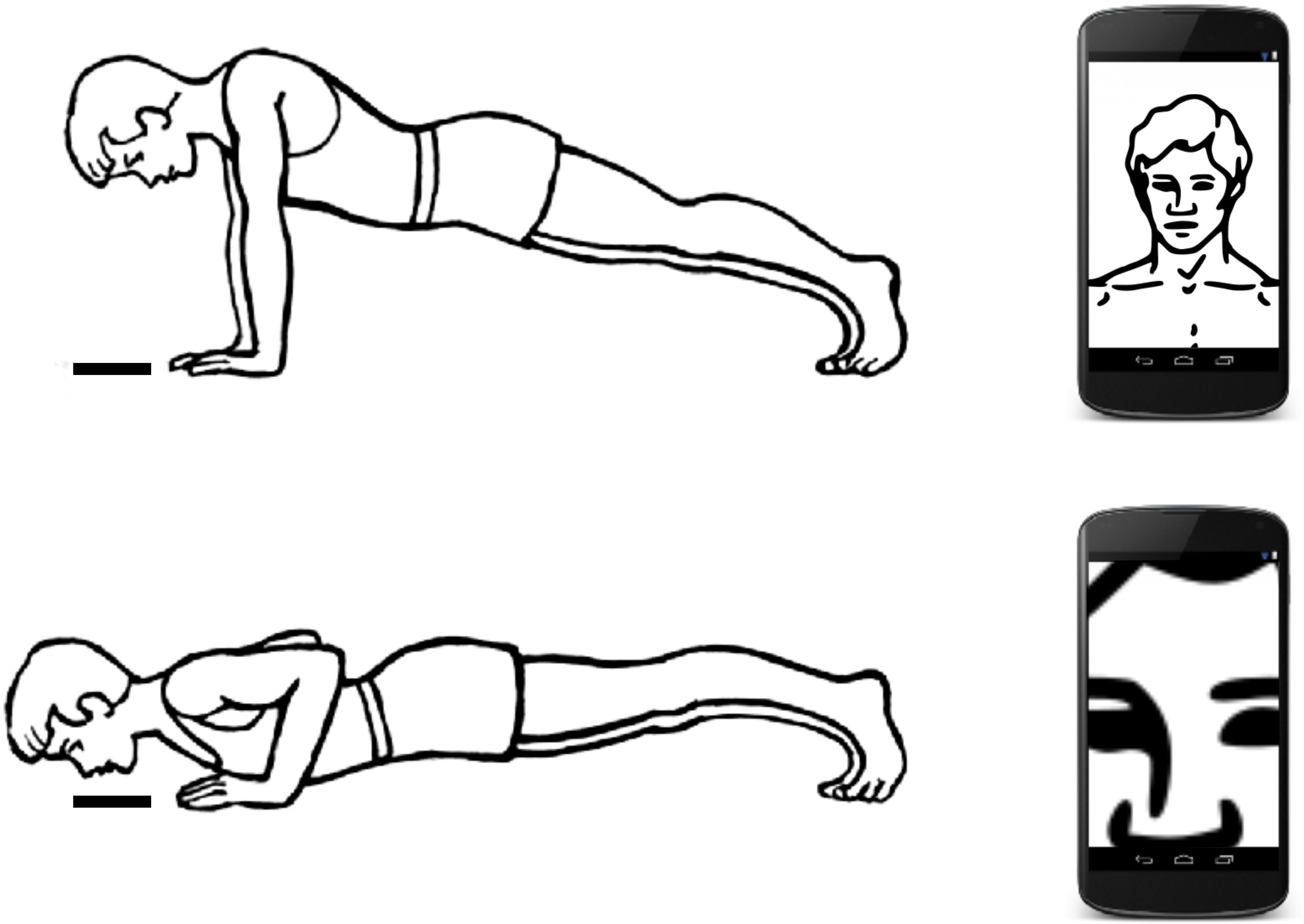
Push-ups!

http://www.barbralingerie.com/product-p/bra-ilys-ib1290-hotpink-parent.htm

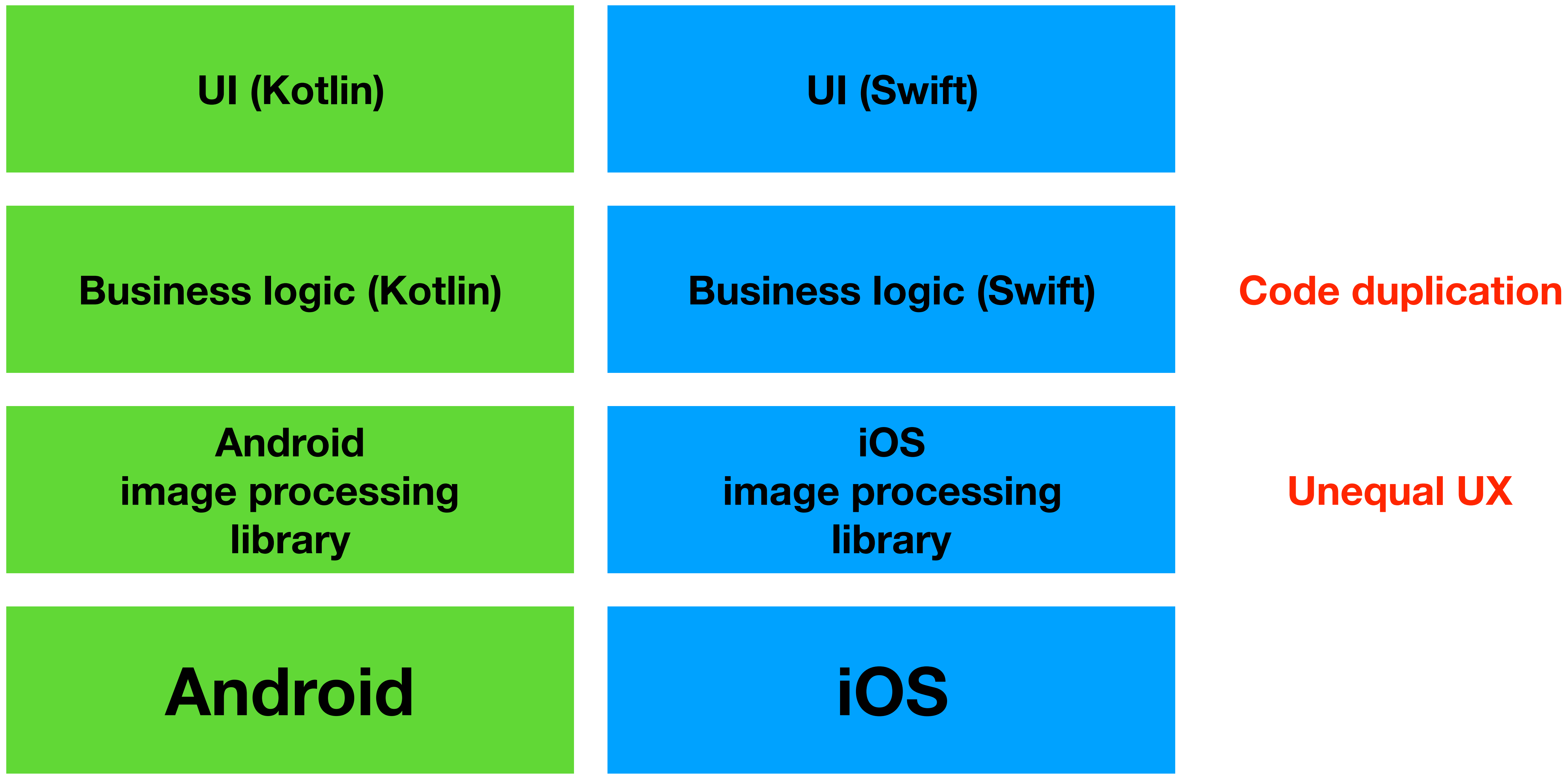
Push-ups!

https://www.pexels.com/photo/adult-athlete-body-exercise-176782

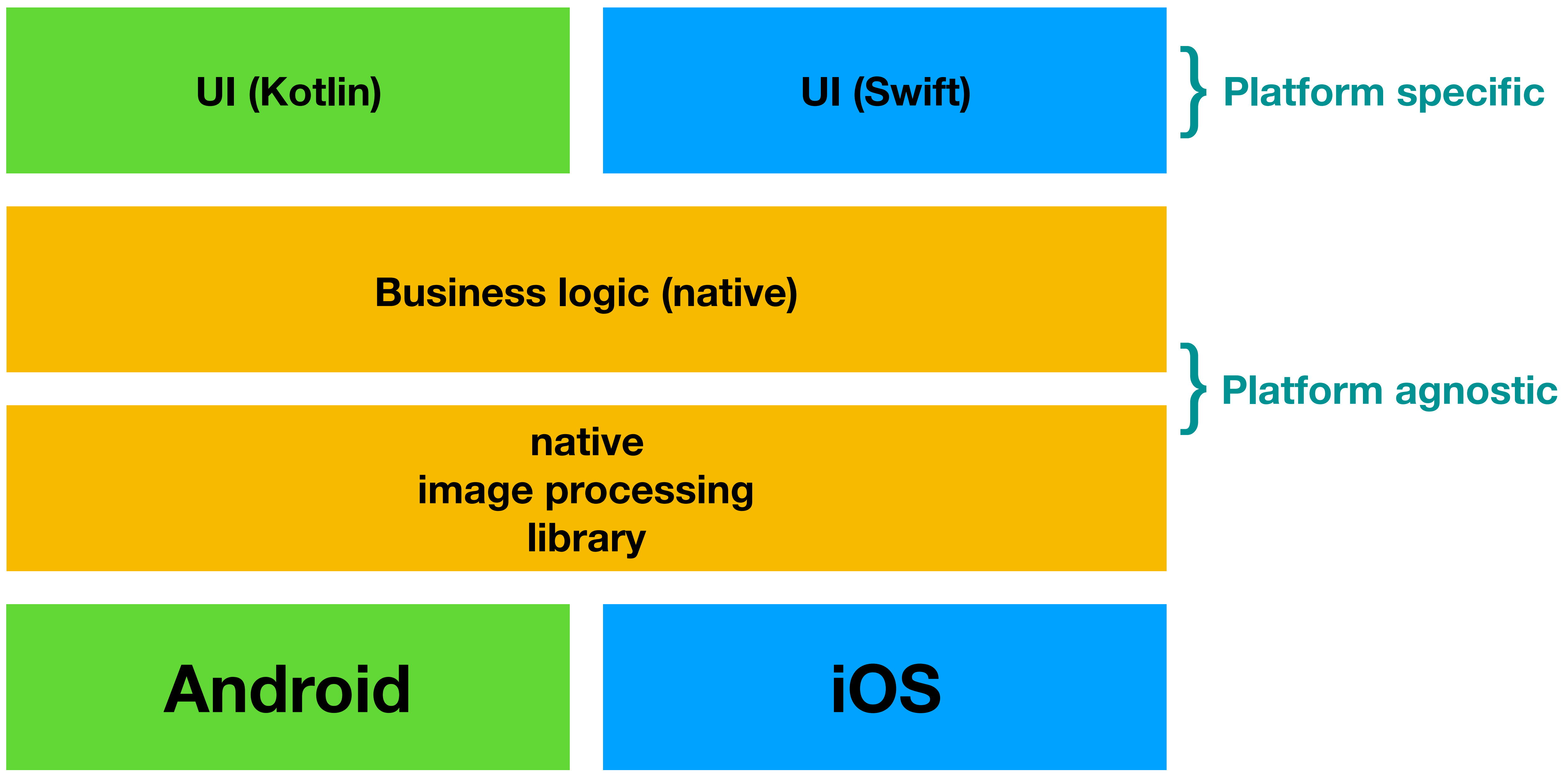
Push-up detection



Architecture



Architecture



OpenCV

OpenCV

- written in C
- interface in C++
- wrappers for Python and Java

OpenCV without C++?

- no Objective-C++ wrappers

Basic level - OpenCV's Java API only.

[…]

Pros

- Fast ramp-up, the simplest way to develop for Android with OpenCV.
- Easy development in a language with garbage collection.

Cons

- Complex CV logic (with many calls to OpenCV) will work slowly because of additional cost of JNI calls.
- Not 100% availability of OpenCV C++ API.

https://opencv.org/platforms/android/opencv4android-usage-models.html

Advanced level - OpenCV's native interface.

[…]

Pros

- Maximum of the performance.
- 100% availability of OpenCV C++ API.
- Development of the core CV functionality on a host platform.
- You can run your code on other platforms: iOS, Windows Phone...
- You can mix Java and native code.

Cons

- You should study Android NDK.
- Development becomes a bit more complicated. But only in the beginning.

https://opencv.org/platforms/android/opencv4android-usage-models.html

stackoverflow matrix

questions in technologies	Java	C++
Android	192 718	4 492
OpenCV	2 598	15 020

Platform specific

Android/iOS specific APIs:

- UI
- Animations
- Text to Speech
- etc.

Variant of realisation

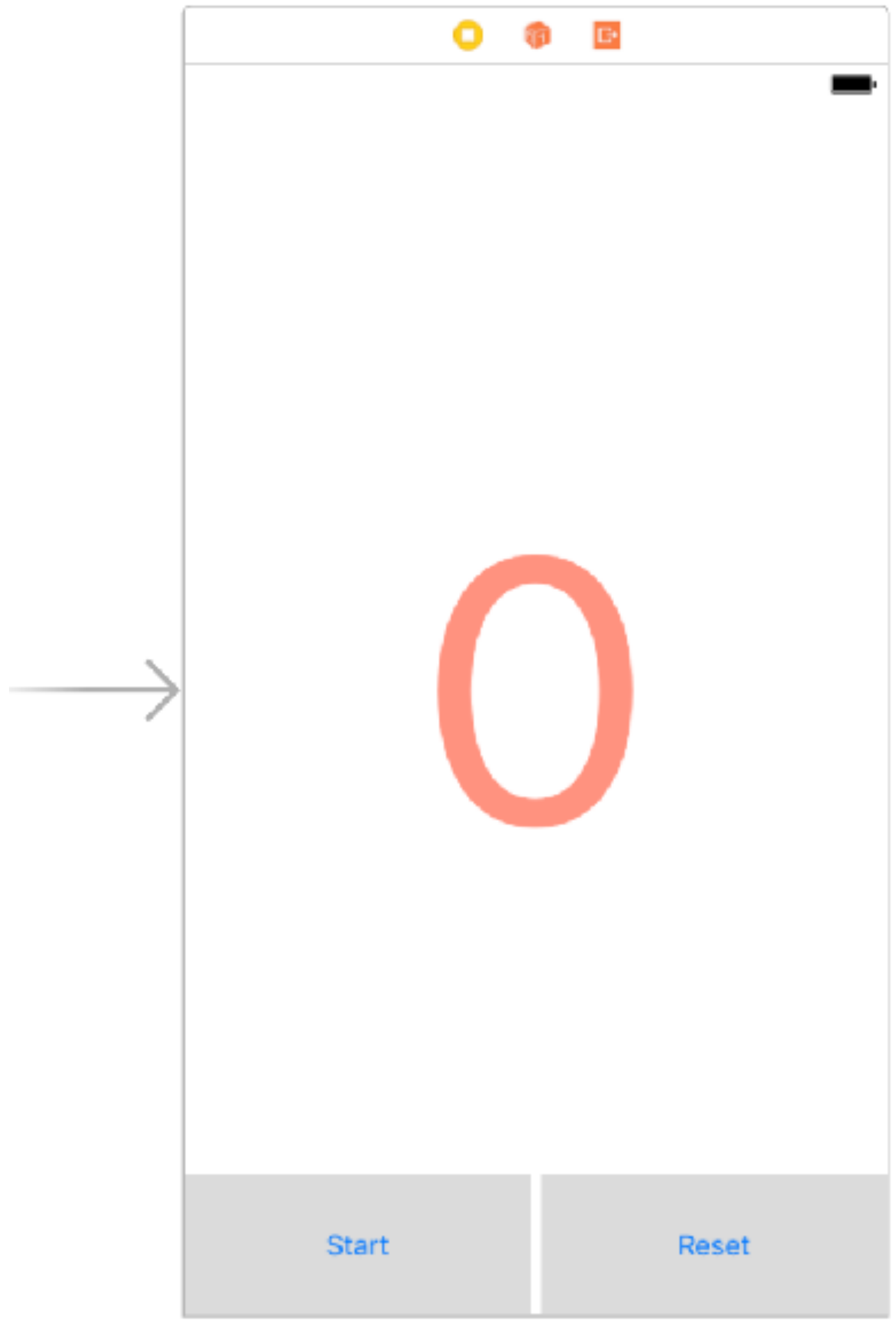
- Platform specific code in Kotlin/Swift
- OpenCV & business logic in C++
- Glue code hand-written?

Glue code generators

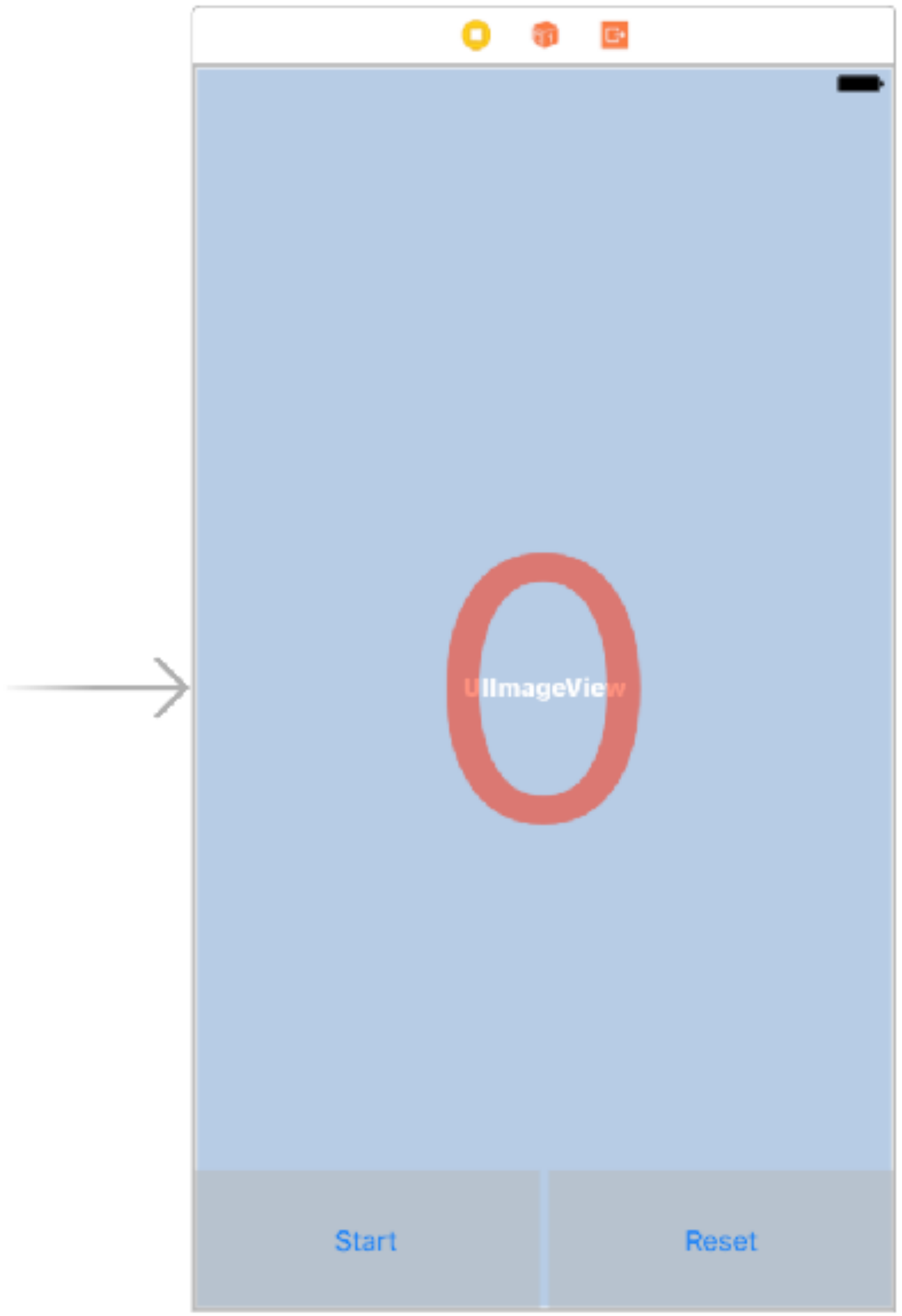
- SWIG
  - Started in 1995
  - Allegro CL, C#, CFFI, CLISP, Chicken, D, Go, Guile, **Java**, Javascript, Lua, Modula-3, Mzscheme, OCAML, Octave, Perl, PHP, Python, R, Ruby, Scilab, Tcl, UFFI
  - In: configuration file (may include C++ headers)
  - Out: glue code for a given language

- SWIG
  - Started in 1995
  - Allegro CL, C#, CFFI, CLISP, Chicken, D, Go, Guile, **Java**, Javascript, Lua, Modula-3, Mzscheme, OCAML, Octave, Perl, PHP, Python, R, Ruby, Scilab, Tcl, UFFI
  - In: configuration file (may include C++ headers)
  - Out: glue code for a given language
- djinni
  - Started in 2014
  - **Objective-C**, **Java**, Python
  - In: configuration file (IDL)
  - Out: glue code for a given language + C++ interfaces

PushUpPal - GUI



PushUpPal - GUI



Djinni’s IDL

```
PushUpPalApp = interface +c {









}
```

```
PushUpPalApp = interface +c {

    start();

    stop();

    reset();

    isStarted(): bool;



}
```

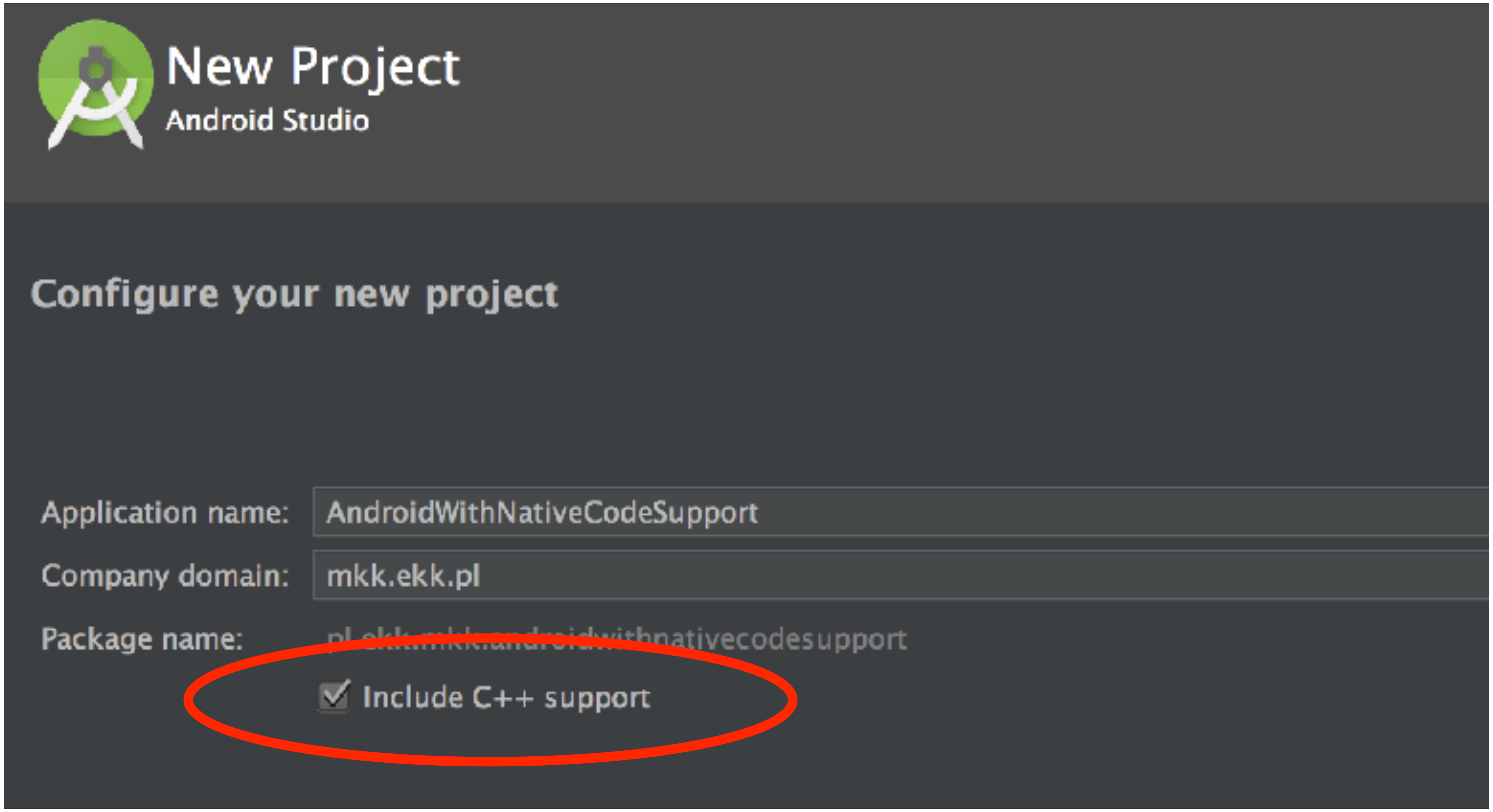
```
PushUpPalApp = interface +c {
    start();
    stop();
    reset();
    isStarted(): bool;
    setListener(listener: PushUpListener);

}

PushUpListener = interface +j +o {
    onPushUp(rep: i32);
}
```

```
PushUpPalApp = interface +c {

    start();

    stop();

    reset();

    isStarted(): bool;

    setListener(listener: PushUpListener);

    static create(classifierFilePath: string): PushUpPalApp;

}


PushUpListener = interface +j +o {

    onPushUp(rep: i32);

}
```

Run Djinni

```
deps/djinni/src/run-assume-built \
deps/djinni/src/run-assume-built \

    --java-out android/app/src/main/java/pl/ekk/mkk/pushuppal/gen \

    --java-package pl.ekk.mkk.pushuppal.generated \

    --jni-out "native/PushUpPal/glue-code/jni/generated" \
deps/djinni/src/run-assume-built \

    --java-out android/app/src/main/java/pl/ekk/mkk/pushuppal/gen \

    --java-package pl.ekk.mkk.pushuppal.generated \

    --jni-out "native/PushUpPal/glue-code/jni/generated" \

    \

    --cpp-out "native/PushUpPal/glue-code/interfaces/generated" \

    --cpp-namespace generated \
deps/djinni/src/run-assume-built \

    --java-out android/app/src/main/java/pl/ekk/mkk/pushuppal/gen \

    --java-package pl.ekk.mkk.pushuppal.generated \

    --jni-out "native/PushUpPal/glue-code/jni/generated" \

    \

    --cpp-out "native/PushUpPal/glue-code/interfaces/generated" \

    --cpp-namespace generated \

    \

    --objc-out "native/PushUpPal/glue-code/objc/generated" \

    --objc-type-prefix PUP \

    --objcpp-out "native/PushUpPal/glue-code/objc/generated" \
deps/djinni/src/run-assume-built \

    --java-out android/app/src/main/java/pl/ekk/mkk/pushuppal/gen \

    --java-package pl.ekk.mkk.pushuppal.generated \

    --jni-out "native/PushUpPal/glue-code/jni/generated" \

    \

    --cpp-out "native/PushUpPal/glue-code/interfaces/generated" \

    --cpp-namespace generated \

    \

    --objc-out "native/PushUpPal/glue-code/objc/generated" \

    --objc-type-prefix PUP \

    --objcpp-out "native/PushUpPal/glue-code/objc/generated" \

    --objc-swift-bridging-header "PushUpPal-Bridging-Header" \
deps/djinni/src/run-assume-built \

    --java-out android/app/src/main/java/pl/ekk/mkk/pushuppal/gen \

    --java-package pl.ekk.mkk.pushuppal.generated \

    --jni-out "native/PushUpPal/glue-code/jni/generated" \

    \

    --cpp-out "native/PushUpPal/glue-code/interfaces/generated" \

    --cpp-namespace generated \

    \

    --objc-out "native/PushUpPal/glue-code/objc/generated" \

    --objc-type-prefix PUP \

    --objcpp-out "native/PushUpPal/glue-code/objc/generated" \

    --objc-swift-bridging-header "PushUpPal-Bridging-Header" \

    \

    --idl PushUpPal.djinni
```

Android - project setup


New Project
Android Studio

Configure your new project

Application name: AndroidWithNativeCodeSupport
Company domain: mkk.ekk.pl
Package name: pl.ekk.mkk.androidwithnativecodesupport

☑ Include C++ support

# NDK support in **local.properties**:

```
ndk.dir=/Users/kowalczm/Library/Android/sdk/ndk-bundle

sdk.dir=/Users/kowalczm/Library/Android/sdk
```

Android - Kotlin

```kotlin
class MainActivity : Activity() {

    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)















    }







}
```

```
class MainActivity : Activity() {
    private var mPushUpPalApp: PushUpPalApp? = null

    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        mPushUpPalApp = PushUpPalApp.create(
            ResourcePathAccessor.getClassifierFilePath(this@MainActivity))





















    }






}
```

```kotlin
class MainActivity : Activity() {
    private var mPushUpPalApp: PushUpPalApp? = null

    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        mPushUpPalApp = PushUpPalApp.create(
            ResourcePathAccessor.getClassifierFilePath(this@MainActivity))

        mPushUpPalApp!!.setListener(object : PushUpListener() {
            override fun onPushUp(rep: Int) {
                runOnUiThread {
                    repsTextView.text = Integer.valueOf(rep)!!.toString()
                }
            }
        }












    }







}
```

```kotlin
class MainActivity : Activity() {
    private var mPushUpPalApp: PushUpPalApp? = null

    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        mPushUpPalApp = PushUpPalApp.create(
            ResourcePathAccessor.getClassifierFilePath(this@MainActivity))

        mPushUpPalApp!!.setListener(object : PushUpListener() {
            override fun onPushUp(rep: Int) {
                runOnUiThread {
                    repsTextView.text = Integer.valueOf(rep)!!.toString()
                }
            }
        }

        startStopButton.setOnClickListener {
            if (mPushUpPalApp!!.isStarted) {
                mPushUpPalApp!!.stop()
                startStopButton.text = "Start"
            } else {
                mPushUpPalApp!!.start()
                startStopButton.text = "Stop"
            }
        }

        resetButton.setOnClickListener {
            mPushUpPalApp!!.reset()
            startStopButton.text = "Start"
        }
    }




}
```

```kotlin
class MainActivity : Activity() {
    private var mPushUpPalApp: PushUpPalApp? = null

    public override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        mPushUpPalApp = PushUpPalApp.create(
            ResourcePathAccessor.getClassifierFilePath(this@MainActivity))

        mPushUpPalApp!!.setListener(object : PushUpListener() {
            override fun onPushUp(rep: Int) {
                runOnUiThread {
                    repsTextView.text = Integer.valueOf(rep)!!.toString()
                }
            }
        }

        startStopButton.setOnClickListener {
            if (mPushUpPalApp!!.isStarted) {
                mPushUpPalApp!!.stop()
                startStopButton.text = "Start"
            } else {
                mPushUpPalApp!!.start()
                startStopButton.text = "Stop"
            }
        }

        resetButton.setOnClickListener {
            mPushUpPalApp!!.reset()
            startStopButton.text = "Start"
        }
    }

    companion object {
        init {
            System.loadLibrary("native-pushuppal")
        }
    }
}
```

CMake

```
file(GLOB_RECURSE SRC_FILES FOLLOW_SYMLINKS
     ../../../native/PushUpPal/src/*.cpp
     ../../../native/PushUpPal/glue-code/jni/*.cpp
     ../../../deps/djinni/support-lib/*.cpp)
```

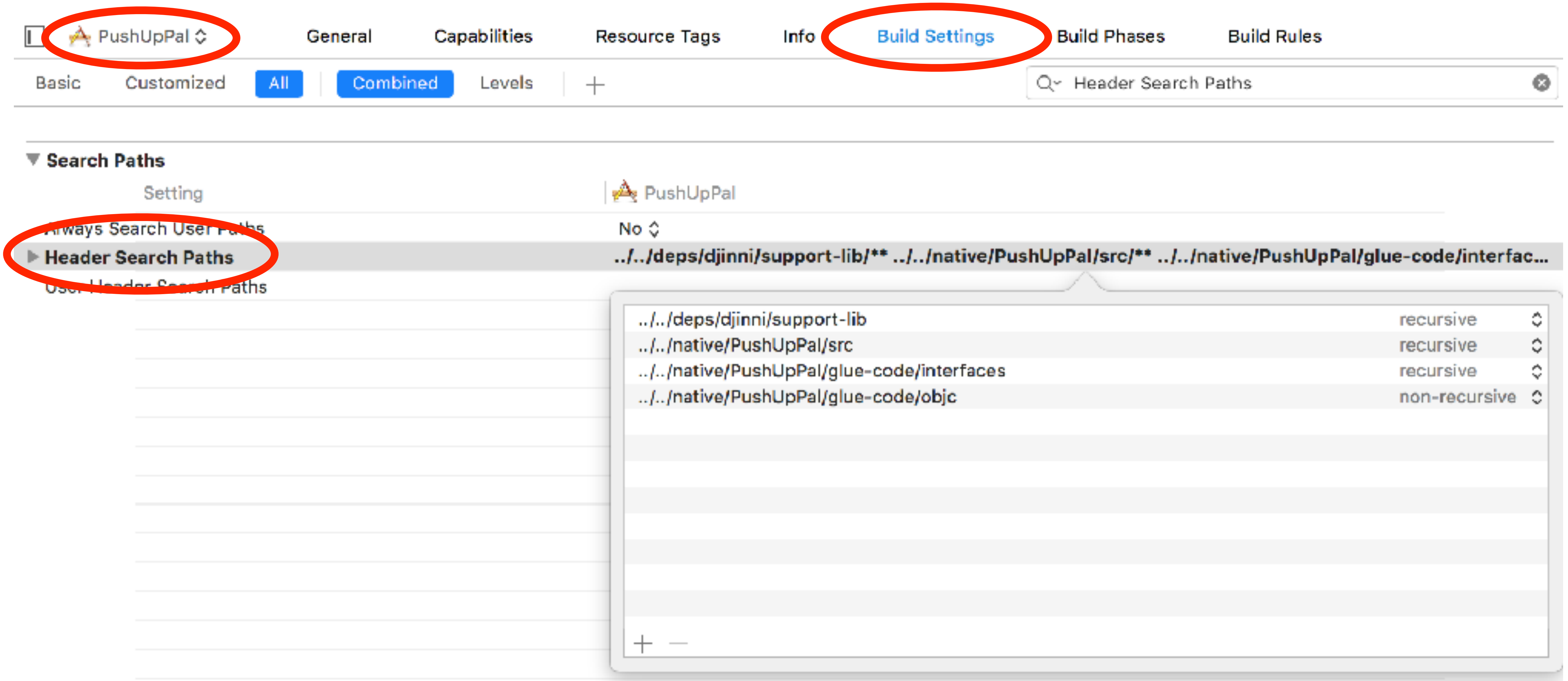
```
file(GLOB_RECURSE SRC_FILES FOLLOW_SYMLINKS
     ../../../native/PushUpPal/src/*.cpp
     ../../../native/PushUpPal/glue-code/jni/*.cpp
     ../../../deps/djinni/support-lib/*.cpp)

include_directories(native/PushUpPal/src
                    native/PushUpPal/glue-code/interfaces/generated
                    native/PushUpPal/glue-code/jni
                    deps/djinni/support-lib
                    deps/djinni/support-lib/jni
                    deps/OpenCV-android-sdk/sdk/native/jni/include)
```

```
file(GLOB_RECURSE SRC_FILES FOLLOW_SYMLINKS
     ../../../native/PushUpPal/src/*.cpp
     ../../../native/PushUpPal/glue-code/jni/*.cpp
     ../../../deps/djinni/support-lib/*.cpp)

include_directories(native/PushUpPal/src
                    native/PushUpPal/glue-code/interfaces/generated
                    native/PushUpPal/glue-code/jni
                    deps/djinni/support-lib
                    deps/djinni/support-lib/jni
                    deps/OpenCV-android-sdk/sdk/native/jni/include)

add_library(native-pushuppal SHARED ${SRC_FILES})
```

```
file(GLOB_RECURSE SRC_FILES FOLLOW_SYMLINKS
     ../../../native/PushUpPal/src/*.cpp
     ../../../native/PushUpPal/glue-code/jni/*.cpp
     ../../../deps/djinni/support-lib/*.cpp)

include_directories(native/PushUpPal/src
                    native/PushUpPal/glue-code/interfaces/generated
                    native/PushUpPal/glue-code/jni
                    deps/djinni/support-lib
                    deps/djinni/support-lib/jni
                    deps/OpenCV-android-sdk/sdk/native/jni/include)

add_library(native-pushuppal SHARED ${SRC_FILES})

add_library(lib-opencv SHARED IMPORTED)
set_target_properties(lib-opencv PROPERTIES IMPORTED_LOCATION
                      ${CMAKE_SOURCE_DIR}/src/main/jniLibs/${ANDROID_ABI}/
                          libopencv_java3.so)
```

```
file(GLOB_RECURSE SRC_FILES FOLLOW_SYMLINKS
     ../../../native/PushUpPal/src/*.cpp
     ../../../native/PushUpPal/glue-code/jni/*.cpp
     ../../../deps/djinni/support-lib/*.cpp)

include_directories(native/PushUpPal/src
                    native/PushUpPal/glue-code/interfaces/generated
                    native/PushUpPal/glue-code/jni
                    deps/djinni/support-lib
                    deps/djinni/support-lib/jni
                    deps/OpenCV-android-sdk/sdk/native/jni/include)

add_library(native-pushuppal SHARED ${SRC_FILES})

add_library(lib-opencv SHARED IMPORTED)
set_target_properties(lib-opencv PROPERTIES IMPORTED_LOCATION
                      ${CMAKE_SOURCE_DIR}/src/main/jniLibs/${ANDROID_ABI}/
                          libopencv_java3.so)

target_link_libraries(native-pushuppal
                      lib-opencv)
```

iOS - headers

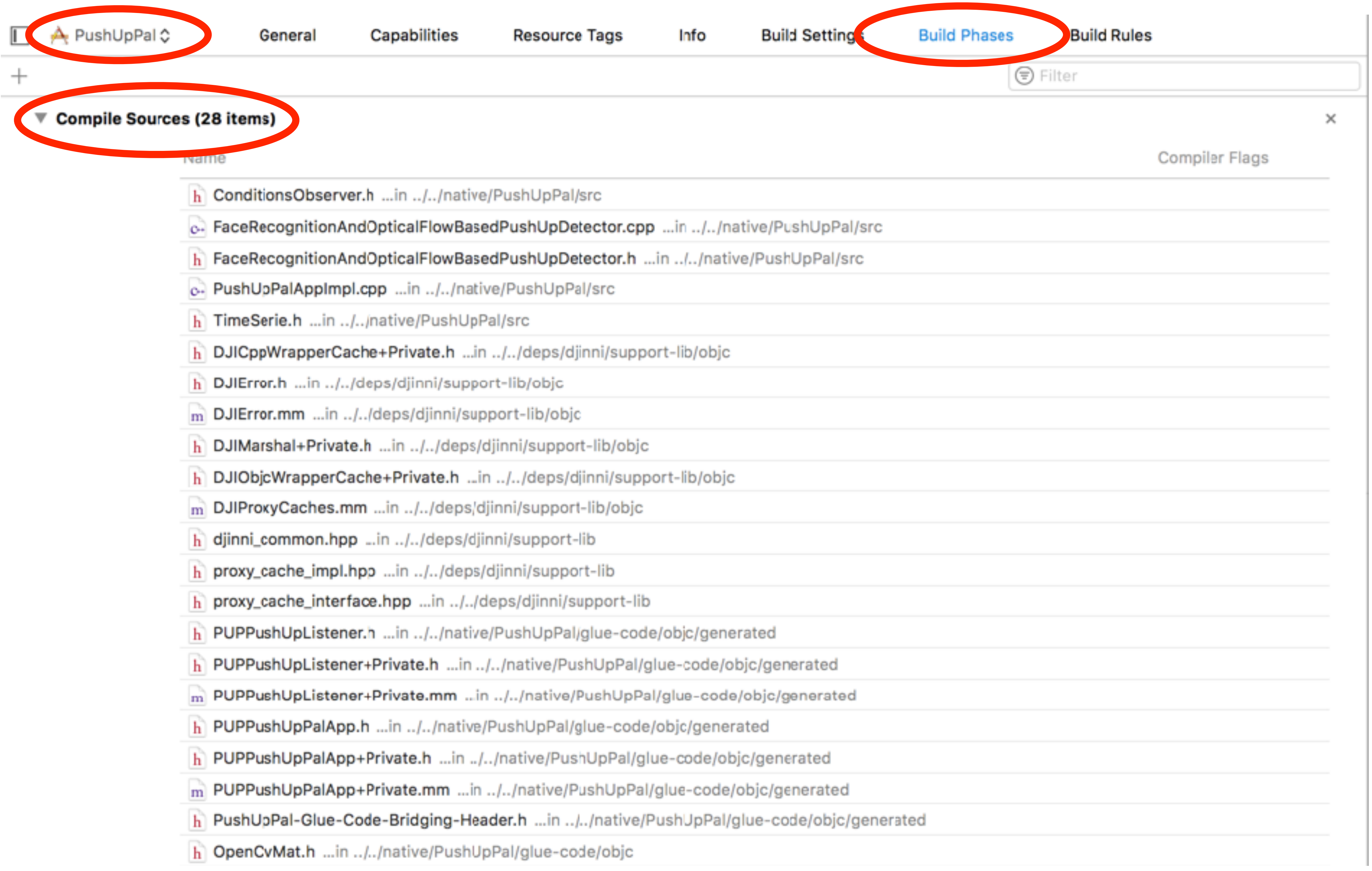


Search Paths

Setting	PushUpPal
Always Search User Paths	No
Header Search Paths	../../deps/djinni/support-lib/** ../../native/PushUpPal/src/** ../../native/PushUpPal/glue-code/interfac...
User Header Search Paths	

../../deps/djinni/support-lib	recursive
../../native/PushUpPal/src	recursive
../../native/PushUpPal/glue-code/interfaces	recursive
../../native/PushUpPal/glue-code/objc	non-recursive

iOS - sources

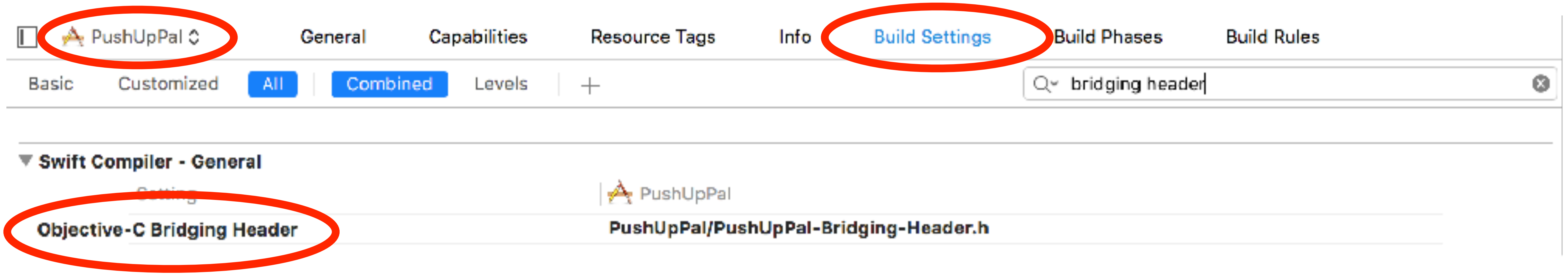


Compile Sources (28 items)

Name	Compiler Flags
ConditionsObserver.h ...in ../../native/PushUpPal/src	
FaceRecognitionAndOpticalFlowBasedPushUpDetector.cpp ...in ../../native/PushUpPal/src	
FaceRecognitionAndOpticalFlowBasedPushUpDetector.h ...in ../../native/PushUpPal/src	
PushUpPalAppImpl.cpp ...in ../../native/PushUpPal/src	
TimeSerie.h ...in ../../native/PushUpPal/src	
DJICppWrapperCache+Private.h ...in ../../deps/djinni/support-lib/objc	
DJIError.h ...in ../../deps/djinni/support-lib/objc	
DJIError.mm ...in ../../deps/djinni/support-lib/objc	
DJIMarshal+Private.h ...in ../../deps/djinni/support-lib/objc	
DJIObjcWrapperCache+Private.h ...in ../../deps/djinni/support-lib/objc	
DJIProxyCaches.mm ...in ../../deps/djinni/support-lib/objc	
djinni_common.hpp ...in ../../deps/djinni/support-lib	
proxy_cache_impl.hpp ...in ../../deps/djinni/support-lib	
proxy_cache_interface.hpp ...in ../../deps/djinni/support-lib	
PUPPushUpListener.h ...in ../../native/PushUpPal/glue-code/objc/generated	
PUPPushUpListener+Private.h ...in ../../native/PushUpPal/glue-code/objc/generated	
PUPPushUpListener+Private.mm ...in ../../native/PushUpPal/glue-code/objc/generated	
PUPPushUpPalApp.h ...in ../../native/PushUpPal/glue-code/objc/generated	
PUPPushUpPalApp+Private.h ...in ../../native/PushUpPal/glue-code/objc/generated	
PUPPushUpPalApp+Private.mm ...in ../../native/PushUpPal/glue-code/objc/generated	
PushUpPal-Glue-Code-Bridging-Header.h ...in ../../native/PushUpPal/glue-code/objc/generated	
OpenCvMat.h ...in ../../native/PushUpPal/glue-code/objc	

iOS - bridging header

```
// AUTOGENERATED FILE - DO NOT MODIFY!
// This file generated by Djinni

#import "PUPPushUpListener.h"
#import "PUPPushUpPalApp.h"
```



Objective-C Bridging Header PushUpPal/PushUpPal-Bridging-Header.h

iOS - Swift

```
class ViewController: UIViewController {
  @IBOutlet weak var repsLabel: UILabel!
  @IBOutlet weak var startStopButton: UIButton!
  @IBOutlet weak var resetButton: UIButton!



  override func viewDidLoad() {
    super.viewDidLoad()






  }























}
```

```
class ViewController: UIViewController {
  @IBOutlet weak var repsLabel: UILabel!
  @IBOutlet weak var startStopButton: UIButton!
  @IBOutlet weak var resetButton: UIButton!

  var pushUpPalApp: PUPPushUpPalApp?

  override func viewDidLoad() {
    super.viewDidLoad()

    let classifierFilePath = Bundle.main.path(forResource: “haarcascade_frontalface_alt2",
                                              ofType: "xml", inDirectory: "")!
    pushUpPalApp = PUPPushUpPalApp.create(classifierFilePath)!


  }









}
```

```
class ViewController: UIViewController, PUPPushUpListener {
  @IBOutlet weak var repsLabel: UILabel!
  @IBOutlet weak var startStopButton: UIButton!
  @IBOutlet weak var resetButton: UIButton!

  var pushUpPalApp: PUPPushUpPalApp?

  override func viewDidLoad() {
    super.viewDidLoad()

    let classifierFilePath = Bundle.main.path(forResource: "haarcascade_frontalface_alt2",
                                              ofType: "xml", inDirectory: "")!
    pushUpPalApp = PUPPushUpPalApp.create(classifierFilePath)!

    pushUpPalApp!.setListener(self)
  }

  func onPushUp(_ rep: Int32) {
    DispatchQueue.global(qos: .userInitiated).async {
      DispatchQueue.main.async {
        self.repsLabel.text = rep.description
      }
    }
  }







}
```

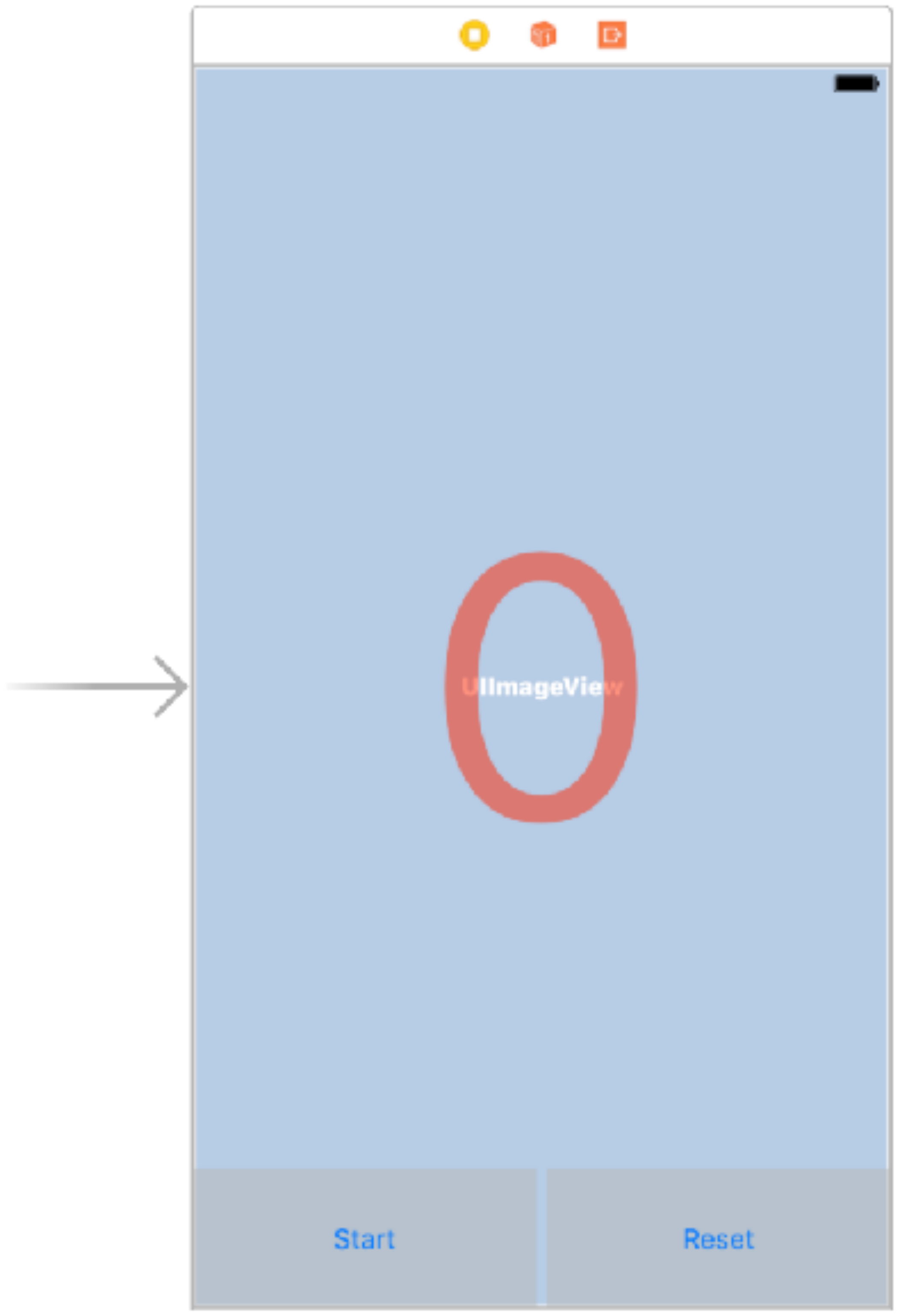
```
class ViewController: UIViewController, PUPPushUpListener {
  @IBOutlet weak var repsLabel: UILabel!
  @IBOutlet weak var startStopButton: UIButton!
  @IBOutlet weak var resetButton: UIButton!

  var pushUpPalApp: PUPPushUpPalApp?

  override func viewDidLoad() {
    super.viewDidLoad()

    let classifierFilePath = Bundle.main.path(forResource: "haarcascade_frontalface_alt2",
                                               ofType: "xml", inDirectory: "")!
    pushUpPalApp = PUPPushUpPalApp.create(classifierFilePath)!

    pushUpPalApp!.setListener(self)
  }

  func onPushUp(_ rep: Int32) {
    DispatchQueue.global(qos: .userInitiated).async {
      DispatchQueue.main.async {
        self.repsLabel.text = rep.description
      }
    }
  }

  @IBAction func startStopButtonOnTouchUpInside(_ sender: UIButton, forEvent event: UIEvent) {
    if pushUpPalApp!.isStarted() {
      pushUpPalApp!.stop()
      startStopButton.setTitle("Start", for: .normal)
    } else {
      pushUpPalApp!.start()
      startStopButton.setTitle("Stop", for: .normal)
    }
  }

  @IBAction func resetButtonOnTouchUpInside(_ sender: UIButton, forEvent event: UIEvent) {
    pushUpPalApp!.reset()
    startStopButton.setTitle("Start", for: .normal)
  }
}
```

Native - C++

```cpp
#include "generated/PushUpPalApp.hpp"

class PushUpPalAppImpl : public generated::PushUpPalApp {
public:
    PushUpPalAppImpl(const std::string& classifierFilePath);

    ~PushUpPalAppImpl() override;

    void start() override;

    void stop() override;

    bool isStarted() override;

    void reset() override;

    void setListener(const std::shared_ptr<generated::PushUpListener>&
        listener) override;

private:
    class Impl;
    std::unique_ptr<Impl> impl_;
};
```

```cpp
#include "generated/PushUpPalApp.hpp"

class PushUpPalAppImpl : public generated::PushUpPalApp {
public:
    PushUpPalAppImpl(const std::string& classifierFilePath);

    ~PushUpPalAppImpl() override;

    void start() override;

    void stop() override;

    bool isStarted() override;

    void reset() override;

    void setListener(const std::shared_ptr<generated::PushUpListener>&
        listener) override;

private:
    class Impl;
    std::unique_ptr<Impl> impl_;
};

std::shared_ptr<generated::PushUpPalApp>
generated::PushUpPalApp::create(const std::string& classifierFilePath)
{
    return std::make_shared<PushUpPalAppImpl>(classifierFilePath);
}
```

What about debugging?



Display camera image

Android:

- Create JavaSurfaceView
- Register for camera updates
- Get a frame (Mat)

iOS:

- Construct CvVideoCamera with UIImageView
- Register for camera updates
- Get a frame (Mat)

Djinni’s IDL: onFrame

```
PushUpPalApp = interface +c {

    start();

    stop();

    reset();

    isStarted(): bool;

    setListener(listener: PushUpListener);

    onFrame(frame: Mat);

    static create(classifierFilePath: string): PushUpPalApp;

}


PushUpListener = interface +j +o {

    onPushUp(rep: i32);

}
```

Djinni: supported types

- types defined in the IDL
- boolean, integer, float, strings, blob, date, list, set, map
- no support by default for foreign types

Djinni: what is a Mat?

- Define all Djinni must know in a YAML file
- Converters for Java Native Interface / Objective-C++:
  - toCpp
  - fromCpp (not required by our IDL, just for completion)

Djinni YAML file

```
name: Mat
typedef: 'record'
params: []
prefix: ''
```

```yaml
name: Mat
typedef: 'record'
params: []
prefix: ''
cpp:
    typename: '::cv::Mat'
    header: '<opencv2/core/mat.hpp>'
    byValue: false
```

```
name: Mat
typedef: 'record'
params: []
prefix: ''
cpp:
    typename: '::cv::Mat'
    header: '<opencv2/core/mat.hpp>'
    byValue: false
objc:
    typename: 'OpenCvMat'
    header: '"OpenCvMat.h"'
    boxed: 'OpenCvMat'
    pointer: true
    hash: '%s.hash'
objcpp:
    translator: '::cv::djinni::objc::OpenCvMat'
    header: '"OpenCvMat+Private.h"'
```

```
name: Mat
typedef: 'record'
params: []
prefix: ''
cpp:
    typename: '::cv::Mat'
    header: '<opencv2/core/mat.hpp>'
    byValue: false
objc:
    typename: 'OpenCvMat'
    header: '"OpenCvMat.h"'
    boxed: 'OpenCvMat'
    pointer: true
    hash: '%s.hash'
objcpp:
    translator: '::cv::djinni::objc::OpenCvMat'
    header: '"OpenCvMat+Private.h"'
java:
    typename: 'org.opencv.core.Mat'
    boxed: 'org.opencv.core.Mat'
    reference: true
    generic: true
    hash: '%s.hashCode()'
jni:
    translator: '::cv::djinni::jni::NativeMat'
    header: '"NativeMat.hpp"'
    typename: jobject
    typeSignature: 'Lorg/opencv/core/Mat;'
```

Mat on Java side

```
package org.opencv.core;

public class Mat {









}
```

Mat on Java side

```
package org.opencv.core;

public class Mat {
    public final long nativeObj;









}
```

Mat on Java side

```java
package org.opencv.core;

public class Mat {
    public final long nativeObj;

    public Mat(long addr) {
        if (addr == 0)
            throw new
                UnsupportedOperationException(
                "Native object address is NULL");
        nativeObj = addr;
    }

    // ...
}
```

Java Native Interface

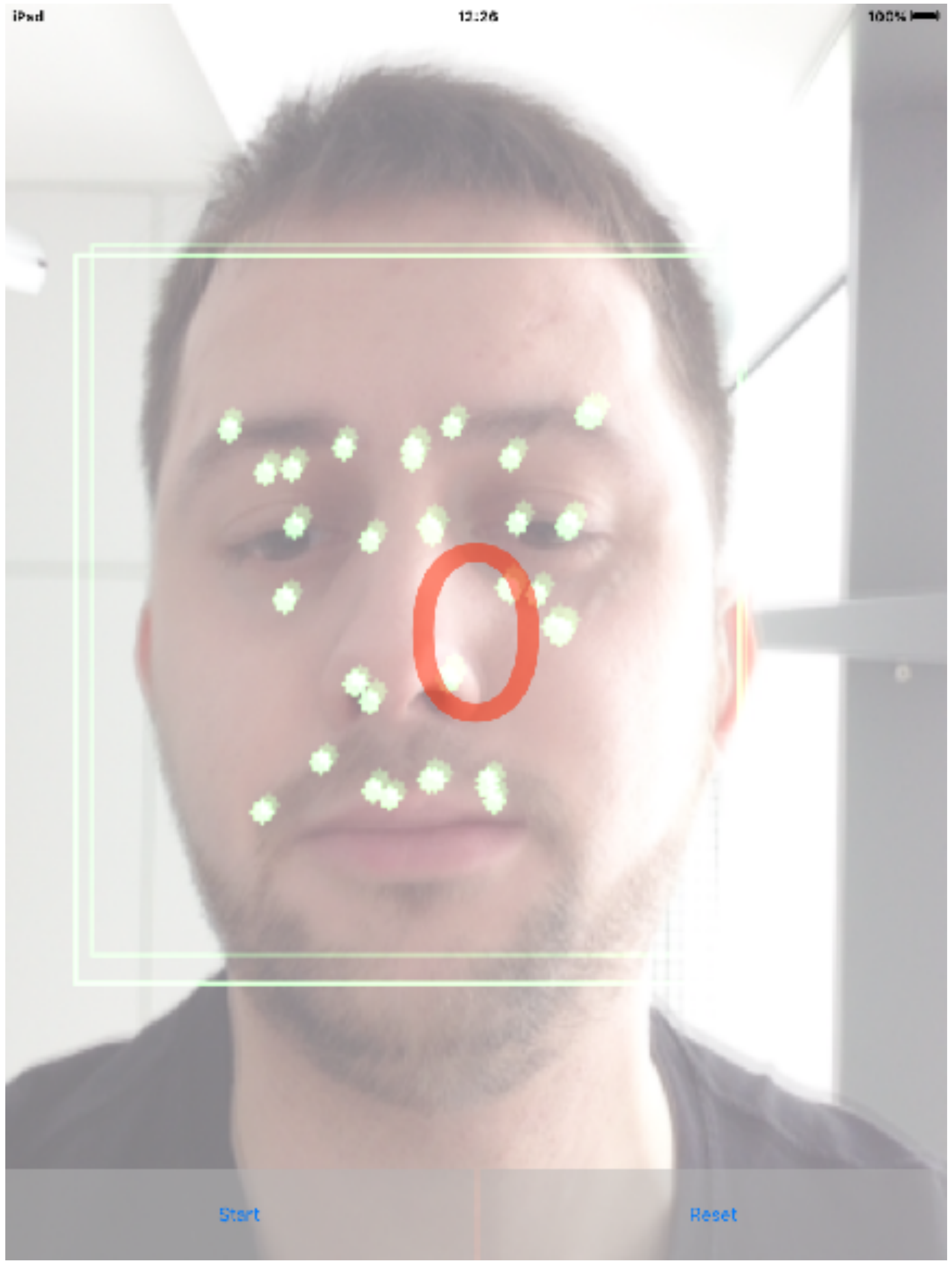
```
#include <opencv2/core/mat.hpp>
#include "djinni_support.hpp"

class NativeMat final :
    ::djinni::JniInterface<::cv::Mat, NativeMat> {
public:
    using CppType = ::cv::Mat;
    using JniType = jobject;
    using Boxed = NativeMat;

    ~NativeMat();

    static CppType toCpp(JNIEnv* jniEnv, JniType matObj) {







    }

    static ::djinni::LocalRef<JniType> fromCpp(
        JNIEnv* jniEnv, const CppType& c) {
        // ...
    }
};
```

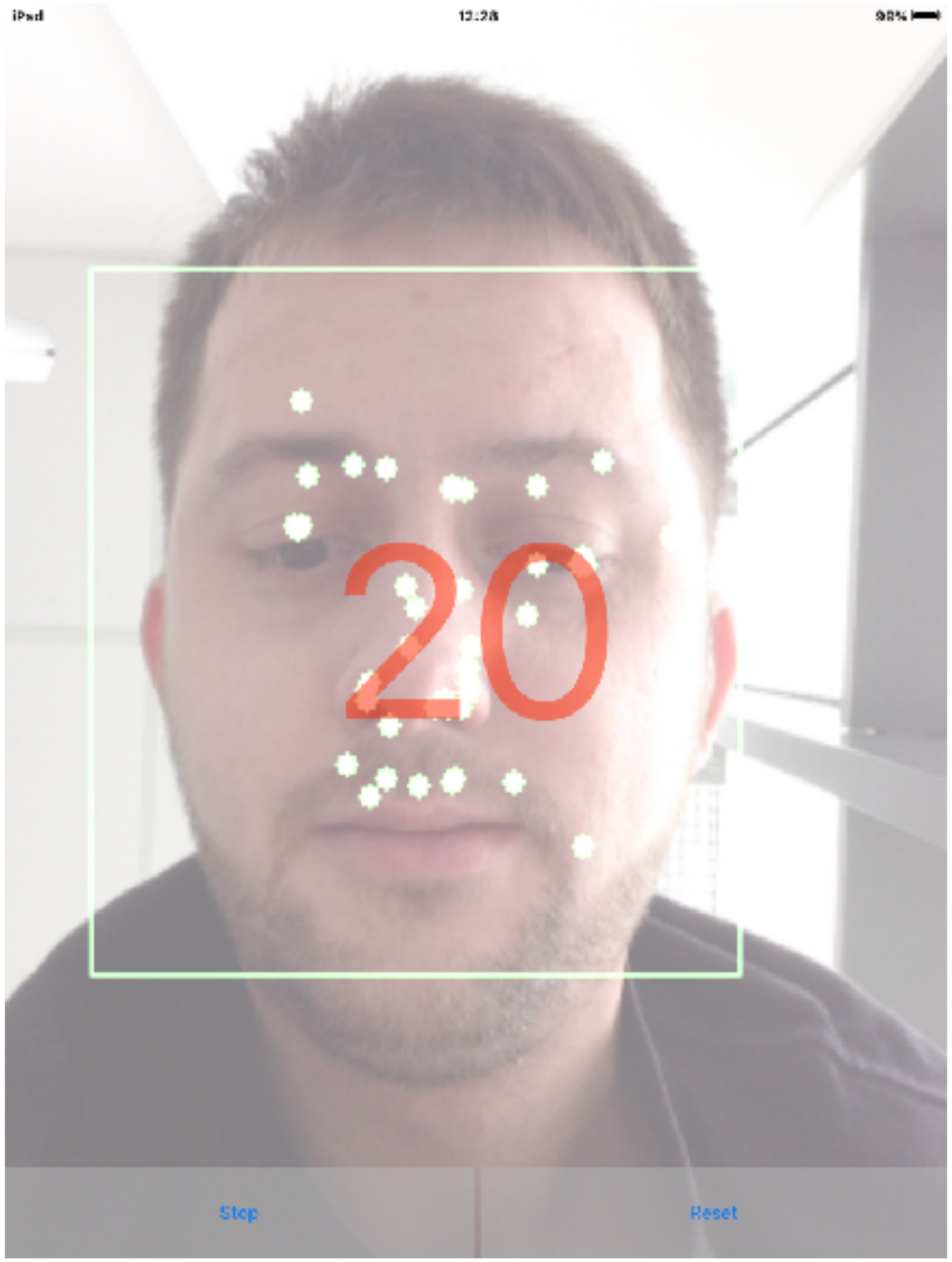
```cpp
#include <opencv2/core/mat.hpp>
#include "djinni_support.hpp"

class NativeMat final :
    ::djinni::JniInterface<::cv::Mat, NativeMat> {
public:
    using CppType = ::cv::Mat;
    using JniType = jobject;
    using Boxed = NativeMat;

    ~NativeMat();

    static CppType toCpp(JNIEnv* jniEnv, JniType matObj) {
        auto matClass = jniEnv->GetObjectClass(matObj);






    }

    static ::djinni::LocalRef<JniType> fromCpp(
        JNIEnv* jniEnv, const CppType& c) {
        // ...
    }
};
```

```cpp
#include <opencv2/core/mat.hpp>
#include "djinni_support.hpp"

class NativeMat final :
    ::djinni::JniInterface<::cv::Mat, NativeMat> {
public:
    using CppType = ::cv::Mat;
    using JniType = jobject;
    using Boxed = NativeMat;

    ~NativeMat();

    static CppType toCpp(JNIEnv* jniEnv, JniType matObj) {
        auto matClass = jniEnv->GetObjectClass(matObj);
        jfieldID nativeObj = jniEnv->GetFieldID(
            matClass, "nativeObj", "J");
        long matPtr = jniEnv->GetLongField(matObj, nativeObj);


    }

    static ::djinni::LocalRef<JniType> fromCpp(
        JNIEnv* jniEnv, const CppType& c) {
        // ...
    }
};
```

```cpp
#include <opencv2/core/mat.hpp>
#include "djinni_support.hpp"

class NativeMat final :
    ::djinni::JniInterface<::cv::Mat, NativeMat> {
public:
    using CppType = ::cv::Mat;
    using JniType = jobject;
    using Boxed = NativeMat;

    ~NativeMat();

    static CppType toCpp(JNIEnv* jniEnv, JniType matObj) {
        auto matClass = jniEnv->GetObjectClass(matObj);
        jfieldID nativeObj = jniEnv->GetFieldID(
            matClass, "nativeObj", "J");
        long matPtr = jniEnv->GetLongField(matObj, nativeObj);
        CppType& mat = *((::cv::Mat*)matPtr);
        return mat;
    }

    static ::djinni::LocalRef<JniType> fromCpp(
        JNIEnv* jniEnv, const CppType& c) {
        // ...
    }
};
```

How it works?





Conclusions

- Reuse of existing native code in mobile applications
- Take advantage of native libraries in Kotlin/Swift
- Reduce code duplication to minimum
- Mostly seamless multi-platform development with Djinni
- Djinni’s support for third party libraries

Not covered

- Kotlin-native
- Swift - C communication
- Numerous alternative approaches for generating bridging code between Swift and C++

Links

- developer.android.com/ndk
- github.com/dropbox/djinni
- mobilecpptutorials.com
- swig.org
- opencv.org
- github.com/michal-kowalczyk/pushuppal
- github.com/michal-kowalczyk/trambambule-helper

Questions?

mkk@ekk.pl
mkk.ekk.pl