openFrameworks

ofxCanon

ofxCanon

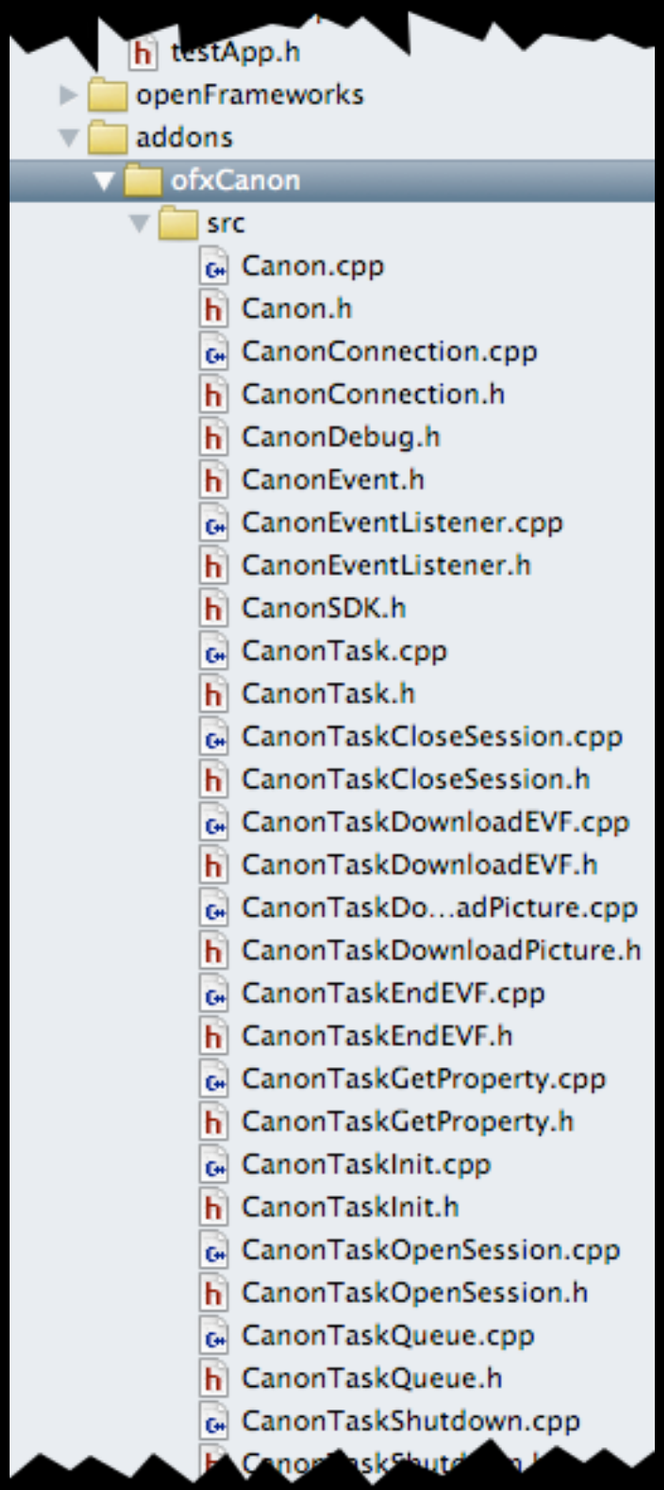
- Wrapper around the Canon EDSK
- You need to apply for the Canon SDK at: http://www.didp.canon-europa.com/ or http://usa.canon.com/cusa/consumer/standard_display/sdk_homepage#SDKQ3
- This keynote will show you how to use the CanonSDK 2.10 on Mac

ofxCanon

- Create a new openFrameworks project by copying the EmptyExample.
- Open the project in XCode (4.x)
- Create a new group under “Addons” and name this “ofxCanon”.
- Download the files for this addon, from: https://github.com/roxlu/ofxCanon/tree/simple
- Drag the “src” directory onto the “ofxCanon” group in XCode.

ofxCanon

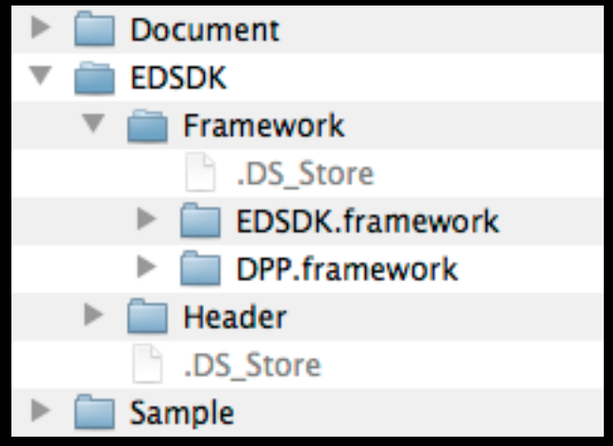
Your project will look something like this in XCode



ofxCanon

Adding the EDSK frameworks

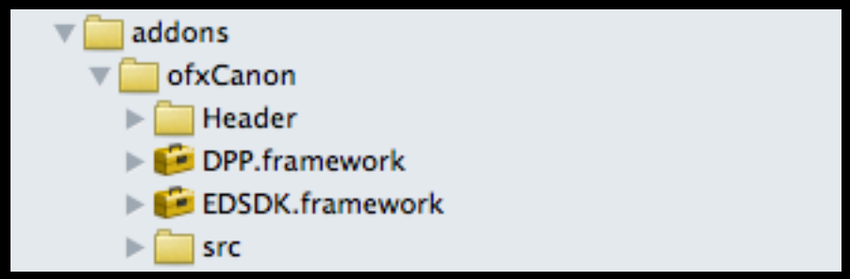
Once you got the EDSK files from Canon you need to add these to your project. I got a .dmg file and once mounted I got these directories.



- Now drag and drop EDSK.framework, DPP.framework and Header into your XCode 4.x project.

ofxCanon

Your project will look something like this in XCode



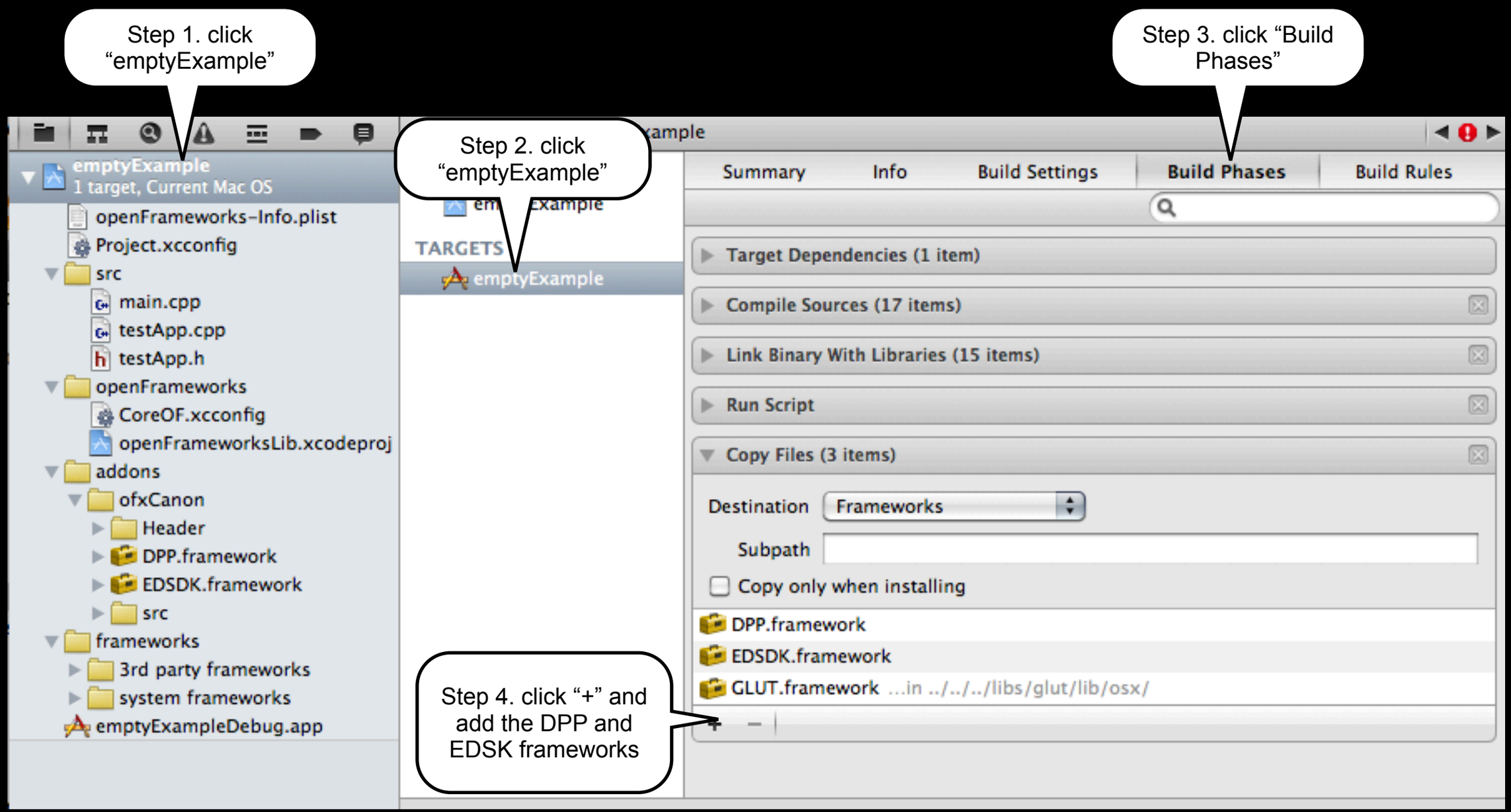
Next step is to make sure the frameworks are copied into your .app.

Check next slide for visual guidance

Click on the blue “emptyExample” project name, then select the “emptyExample” target. Next click “Build phases” and open “Copy files”. Click the “+” button and select DPP.framework and EDSK.framework

ofxCanon

Adding the frameworks to your application bundle (.app)



Using ofxCanon

Taking a picture

Include Canon.h and create a Canon instance in i.e. testApp. Then make sure to call “start()” to initialize the connection with the Camera.

testApp.h

```
#pragma once

#include "ofMain.h"
#include "Canon.h"

class testApp : public ofBaseApp{
   public:
        roxlu::Canon canon;
};
```

testApp.cpp

```
void testApp::setup(){
   canon.start();
}


void testApp::keyPressed(int key){
   if(key == ' ') {
        canon.takePicture();
   }
}
```

Using ofxCanon

Listen for event when image has been downloaded

```
void testApp::setup(){
   canon.start();
   canon.addPictureTakenListener(this, &testApp::onPictureTaken);
}

void testApp::onPictureTaken(roxlu::CanonPictureEvent& ev) {
   cout << ev.getFilePath() << endl;
}
```

Using ofxCanon

Starting live view (get video)

Call `startLiveView()` on your Canon object after the session has been opened. Draw the video using `drawLiveView()` end the live view using `endLiveView()`.

Using ofxCanon

Starting live view (get video)

```
void testApp::setup(){
   canon.start();
   canon.addPictureTakenListener(this, &testApp::onPictureTaken);
}

void testApp::onPictureTaken(roxlu::CanonPictureEvent& ev) {
   cout << ev.getFilePath() << endl;
}

//--------------------------------------------------------------
void testApp::update(){
   if(!canon.isLiveViewActive() && canon.isSessionOpen()) {
       canon.startLiveView();
   }
}

//--------------------------------------------------------------
void testApp::draw(){
   canon.drawLiveView();
}
```

roxlu

www.roxlu.com