

IDD Media lab. 2015

openFrameworks はじめの一歩

2015年4月21日
田所淳

リサーチ課題補足

リサーチ課題補足

- ▶ いくつか、リサーチ課題の参考になりそうなページを見つけたので紹介

リサーチ課題補足

- ▶ 60210-A • ELECTRONIC MEDIA STUDIO (INTERACTIVITY)
- ▶ カーネギーメロン大、Golan Levin教授を中心に運営されているスタジオ

60210-A • ELECTRONIC MEDIA STUDIO (INTERACTIVITY)

Professor Golan Levin • Fall 2014

[ABOUT](#) ▾ [ASSIGNMENTS](#) ▾ [LECTURES](#) ▾ [LOGISTICS](#) ▾ [RESOURCES](#) ▾ [STUDENT AREA](#)

ABOUT

Electronic Media Studio: Introduction to Interactivity (60-210) is an introduction to software programming and physical computing within the context of the arts. In this course students develop the skills and confidence to produce interactive artworks using audiovisual, networked and tangible media. Section A (taught by Golan Levin) has a partial emphasis on generative form and interactive visualization.

Class Time: Mondays and Wednesdays, 6:30-9:20pm.

Location: CFA-318 (College of Fine Arts building, CMU)

Instructor: Golan Levin ([contact information](#))

Office Hours: M/W afternoons, 1:30-3:00pm

Office Location: CFA-111 ([Frank-Ratchye STUDIO for Creative Inquiry](#))



AUTHORS

[A Bear of Similar Size](#)

[AAW](#)

[BRL](#)

[CAC](#)

[Charlotte](#)

[David Gordon](#)

[Elizabeth](#)

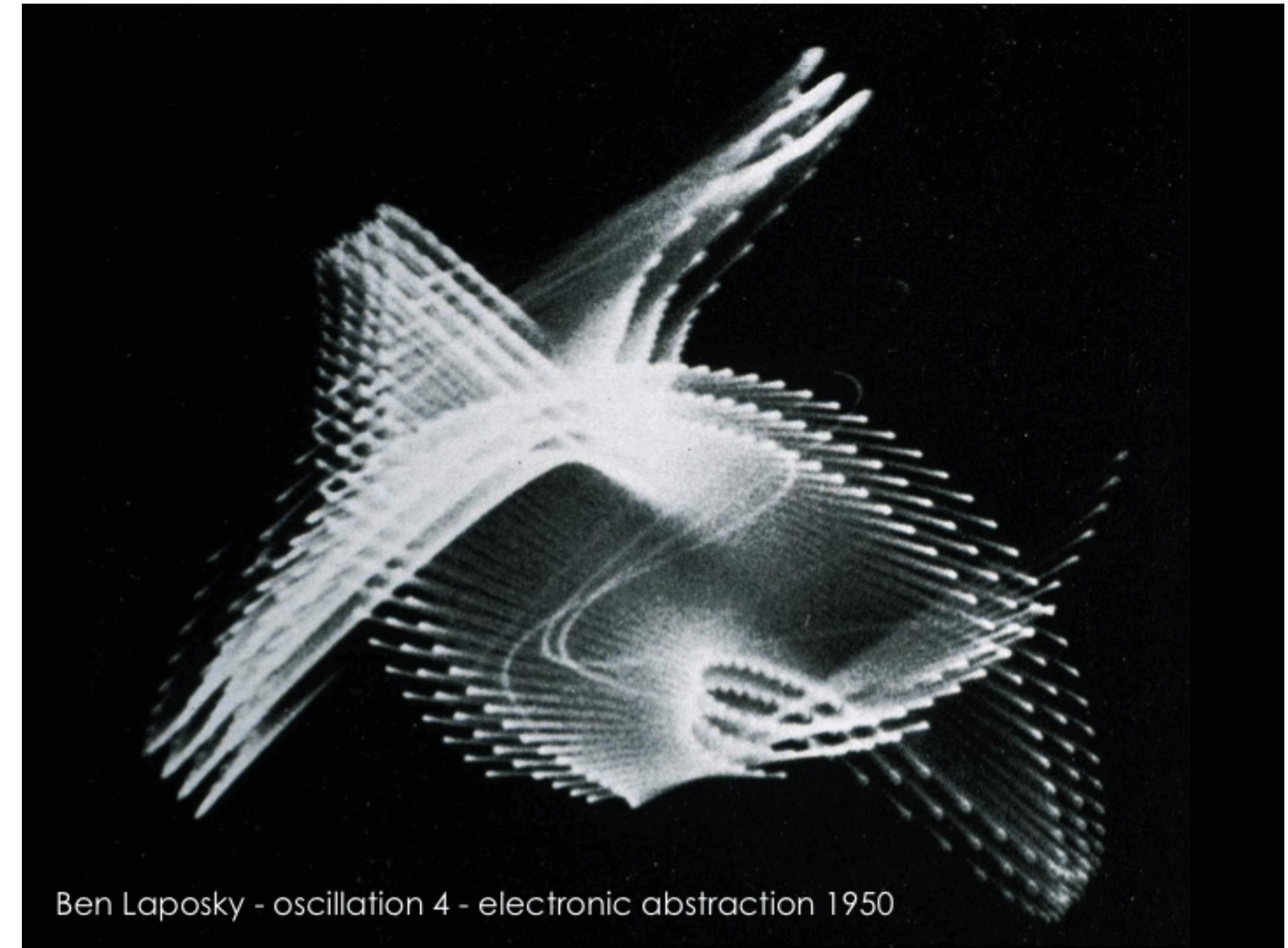
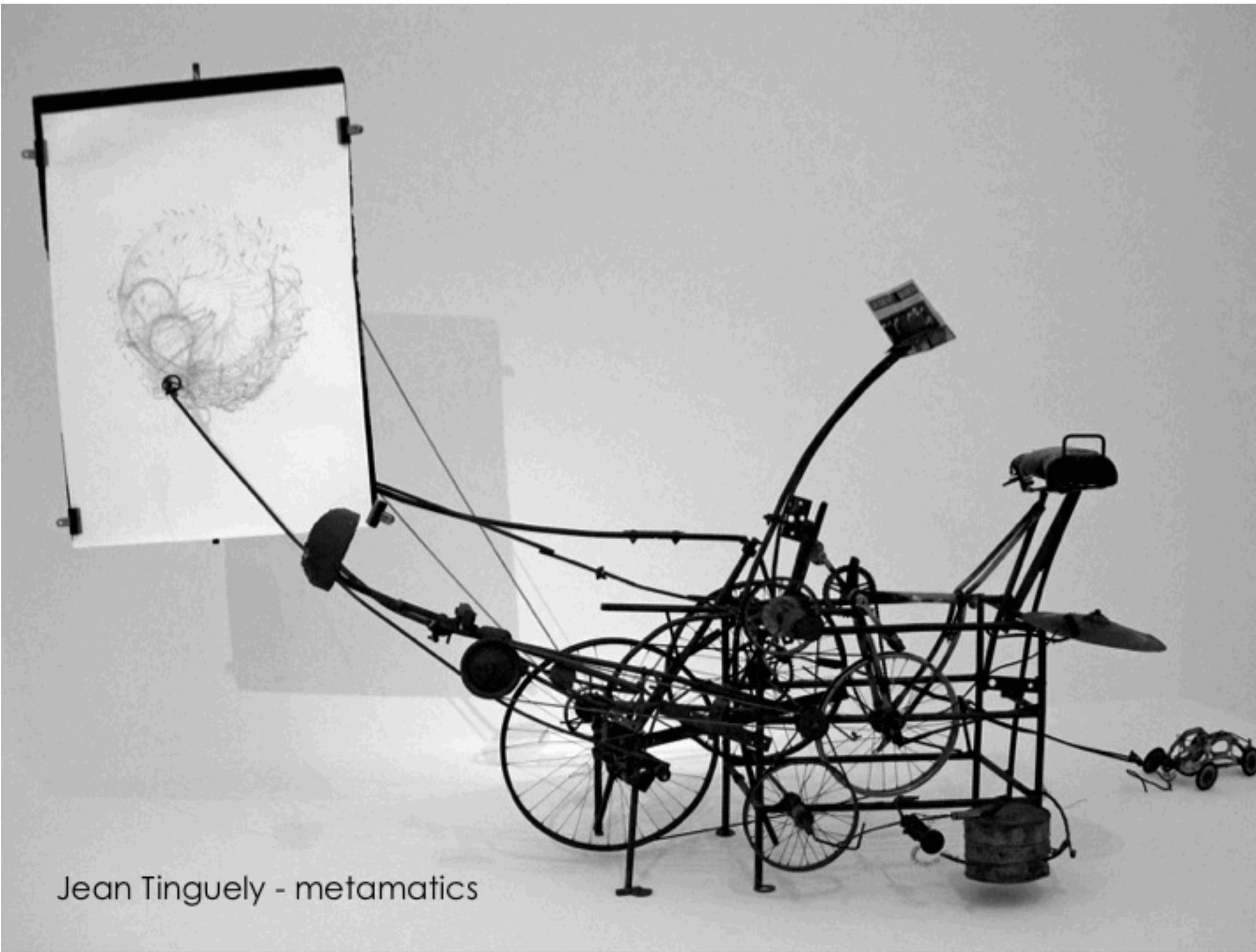
[Hizal](#)

[Isabella Antolic-Soban](#)

[John Choi](#)

リサーチ課題補足

- ▶ LECTURE 01: MACHINE ART
- ▶ 「ドローイング・ボット」のヒントになりそう

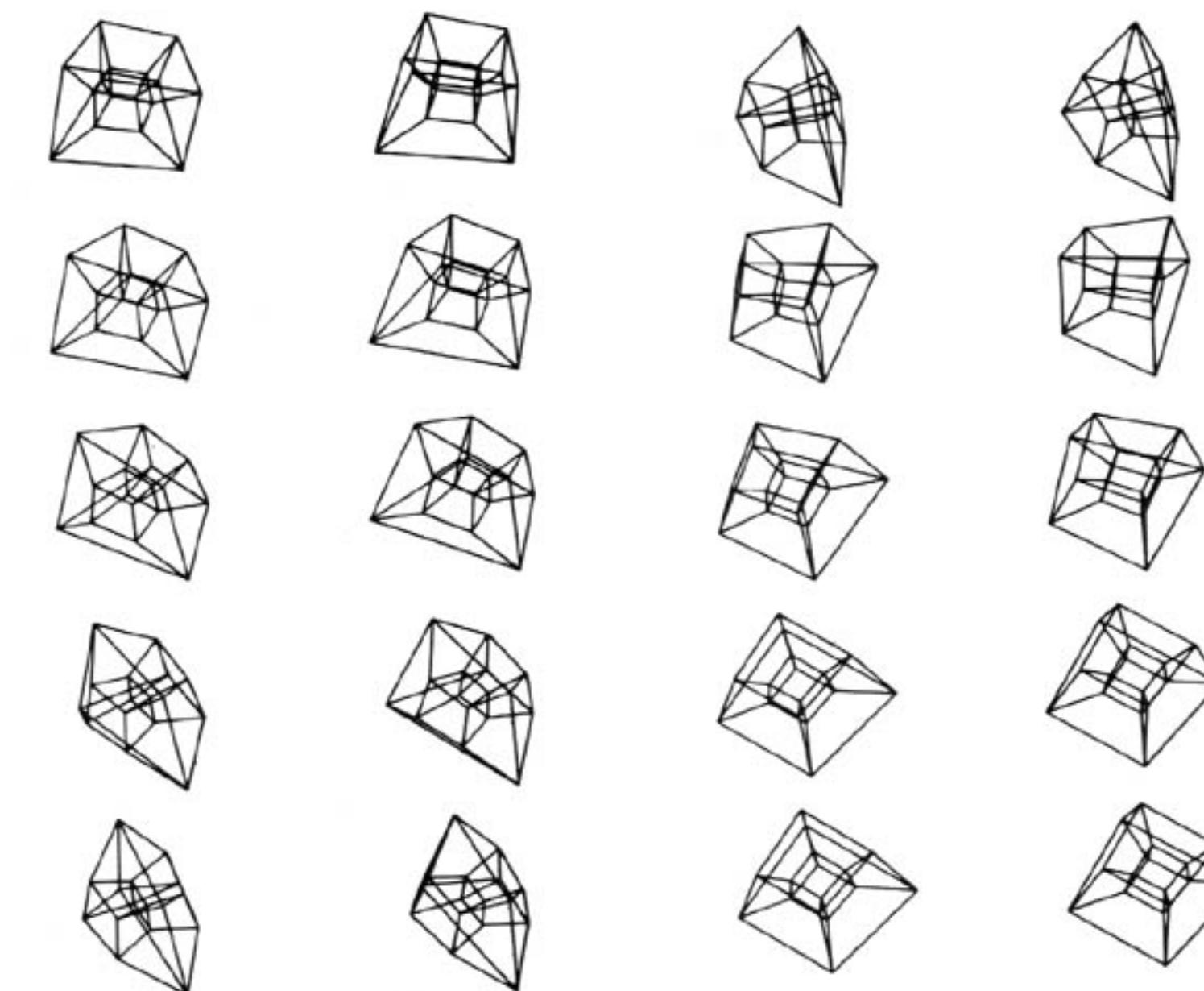
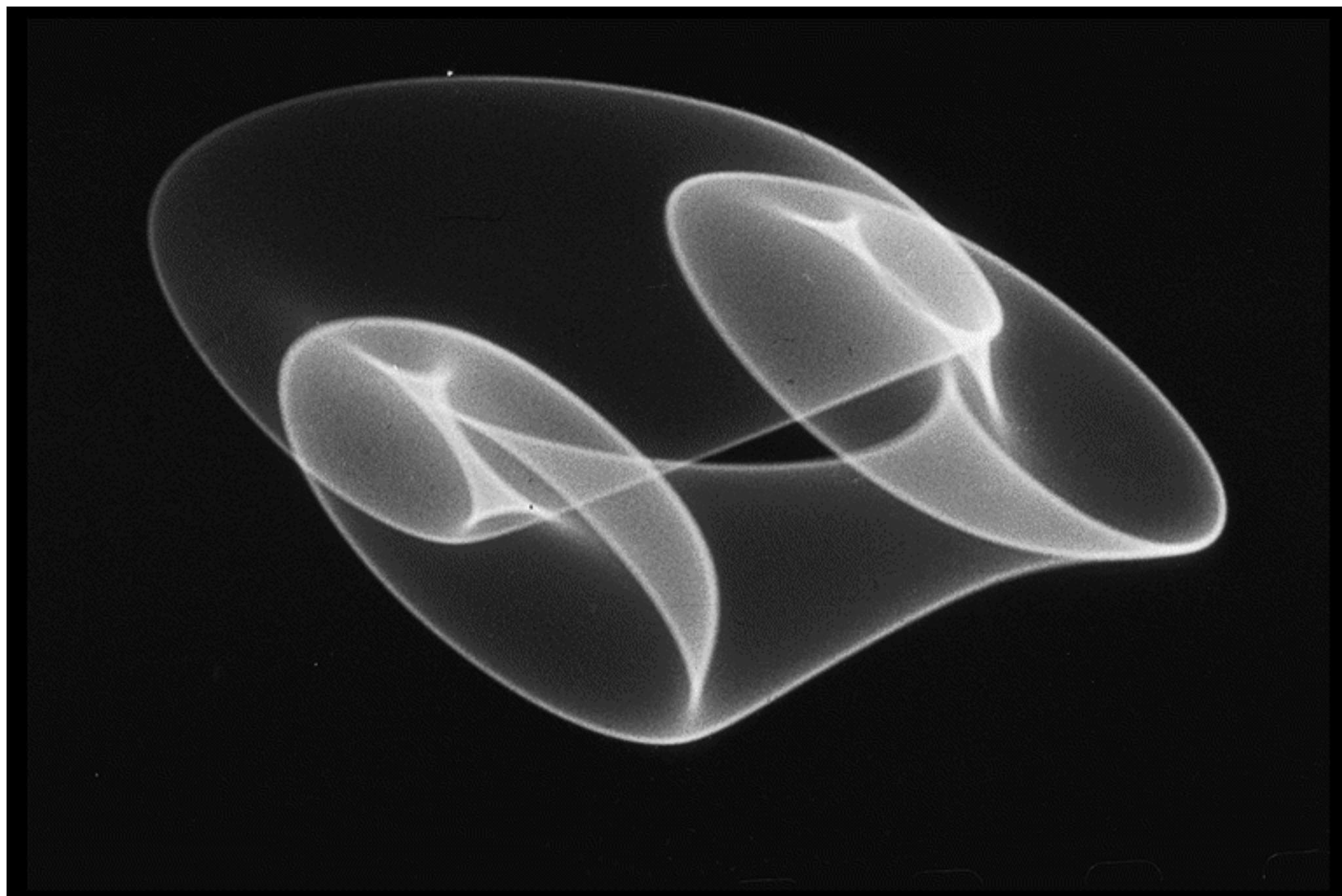


リサーチ課題補足

- ▶ 参考:
- ▶ Jean Tinguely, Metamatics - <https://youtu.be/G0o5uq2fH6g>
- ▶ Ivan Sutherland, Sketchpad - <https://youtu.be/495nCzxM9Pl>
- ▶ Ben F Laposky, Oscillons (Electronic Abstractions) - <http://www.atariarchives.org/artist/sec6.php>
- ▶ Desmond Paul Henry, <http://www.desmondhenry.com/>
- ▶ Sol LeWitt, A Wall Drawing Retrospective - <https://www.youtube.com/watch?v=c4cgB4vJ2XY>
- ▶ The Responsive Eye, Part 1 Mike Wallace 1965 - <https://youtu.be/XSVQqJo0Pmk>

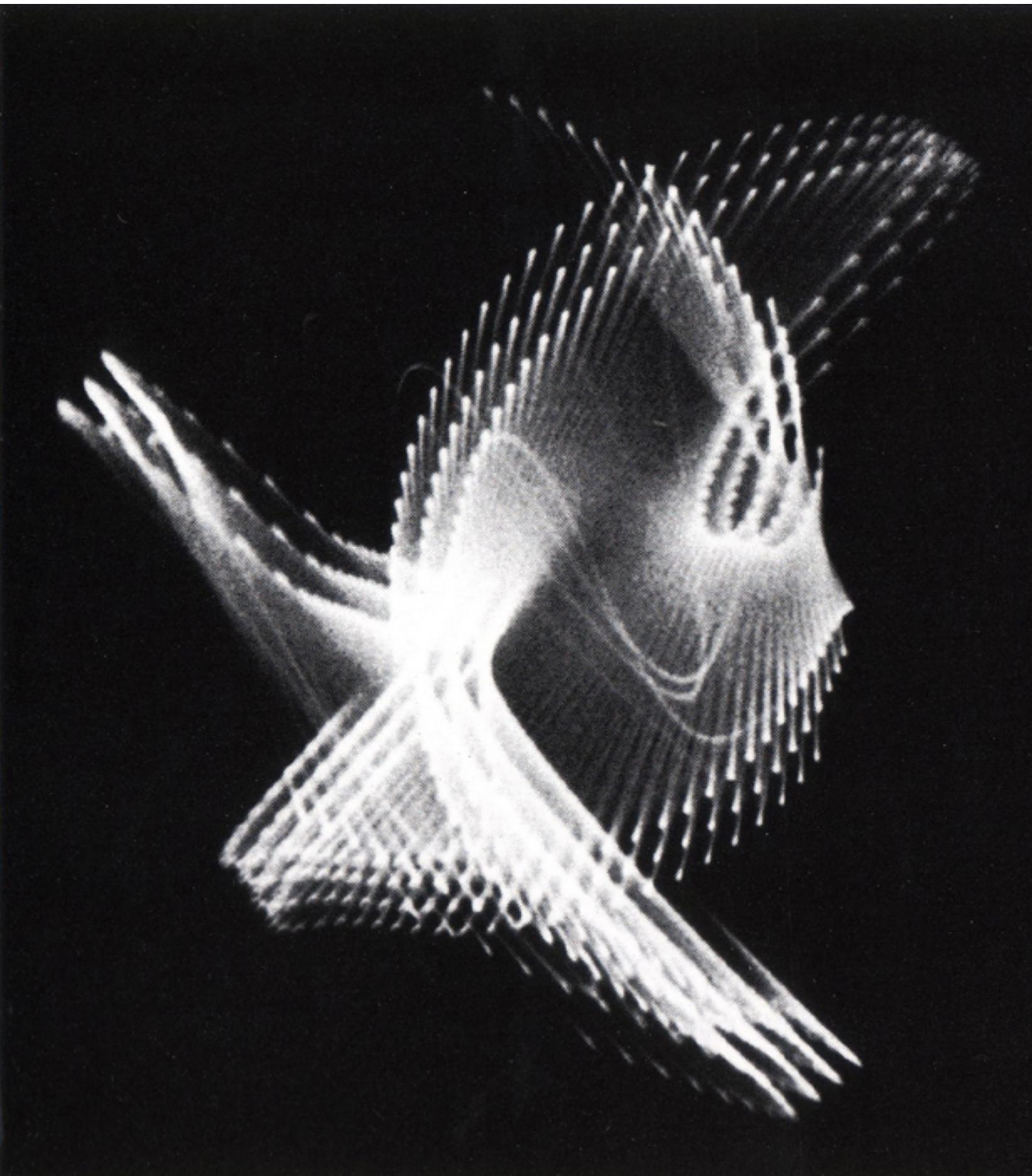
リサーチ課題補足

- ▶ VISUAL AESTHETICS IN EARLY COMPUTING (1950-80)
- ▶ コンピューター黎明期の視覚的な美学



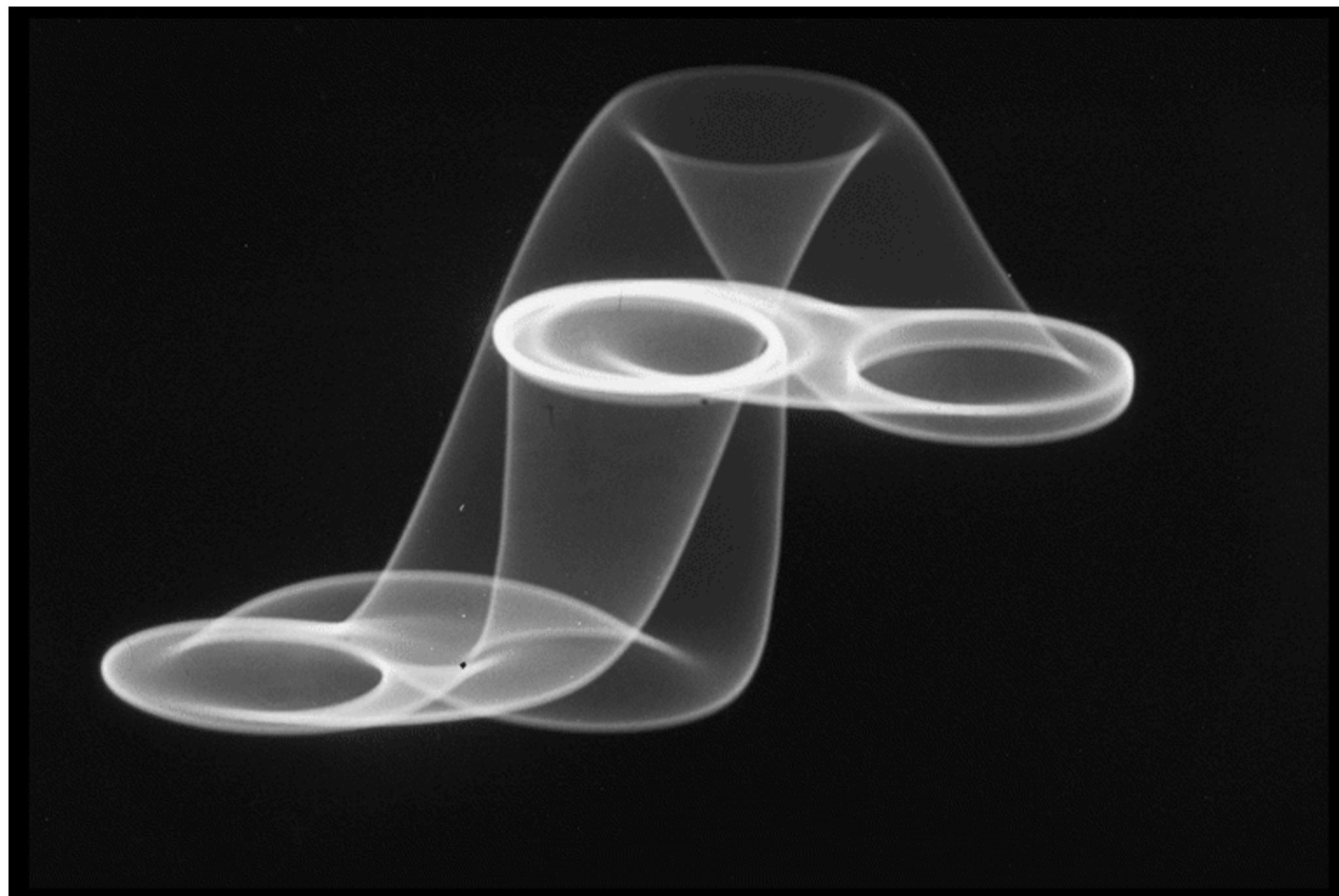
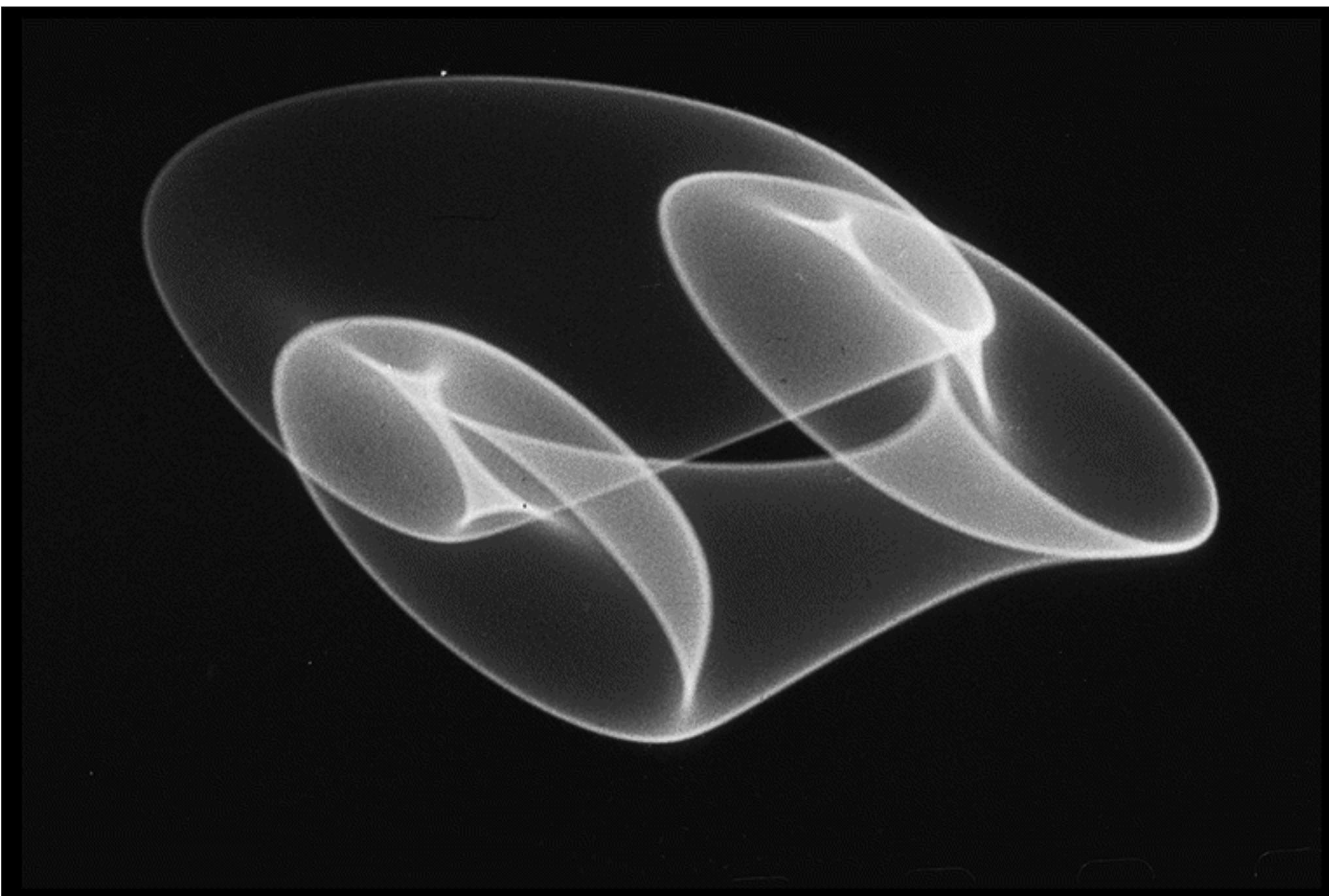
リサーチ課題補足

- ▶ Ben F Laposky : Oscillons, 1952-56



リサーチ課題補足

- ▶ Herbert Franke : Electronic Graphics, 1961-62



リサーチ課題補足

- ▶ A Michael Noll : Computer Composition With Lines, 1964



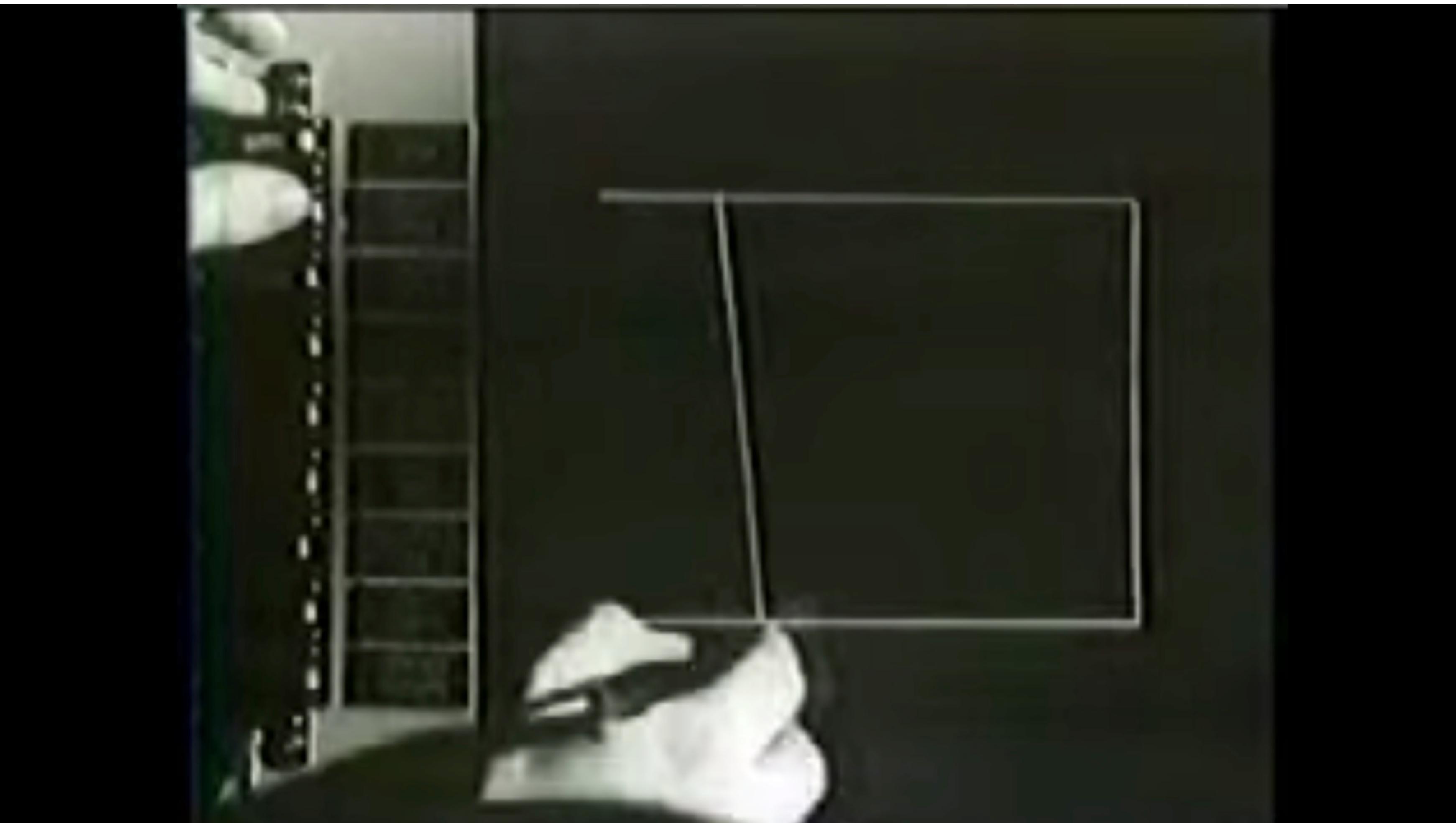
リサーチ課題補足

- ▶ Ivan Sutherland: Sketchpad, 1963



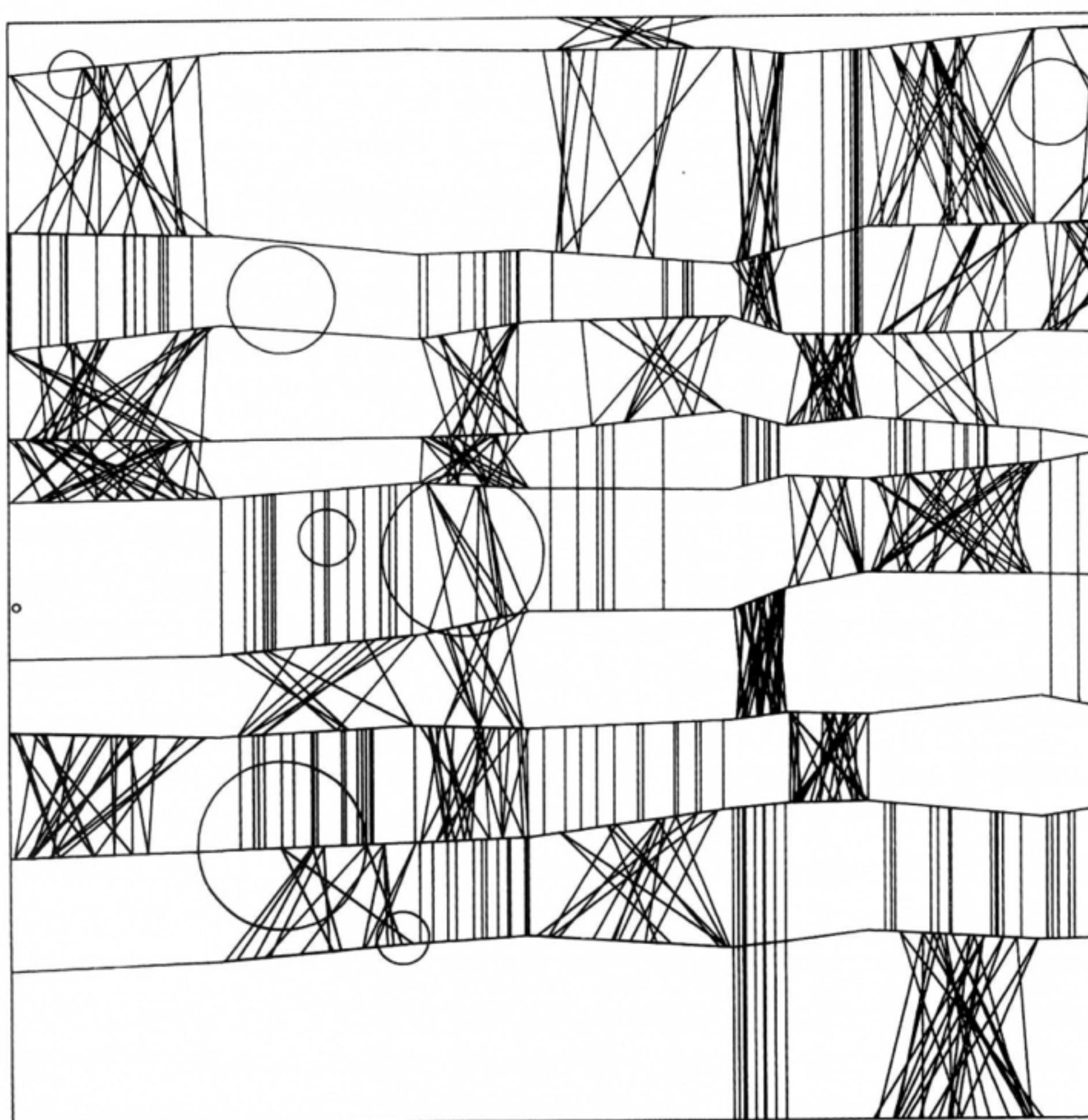
リサーチ課題補足

- ▶ 参考: [Sketchpad, by Dr. Ivan Sutherland with comments by Alan Kay](#) (YouTube)



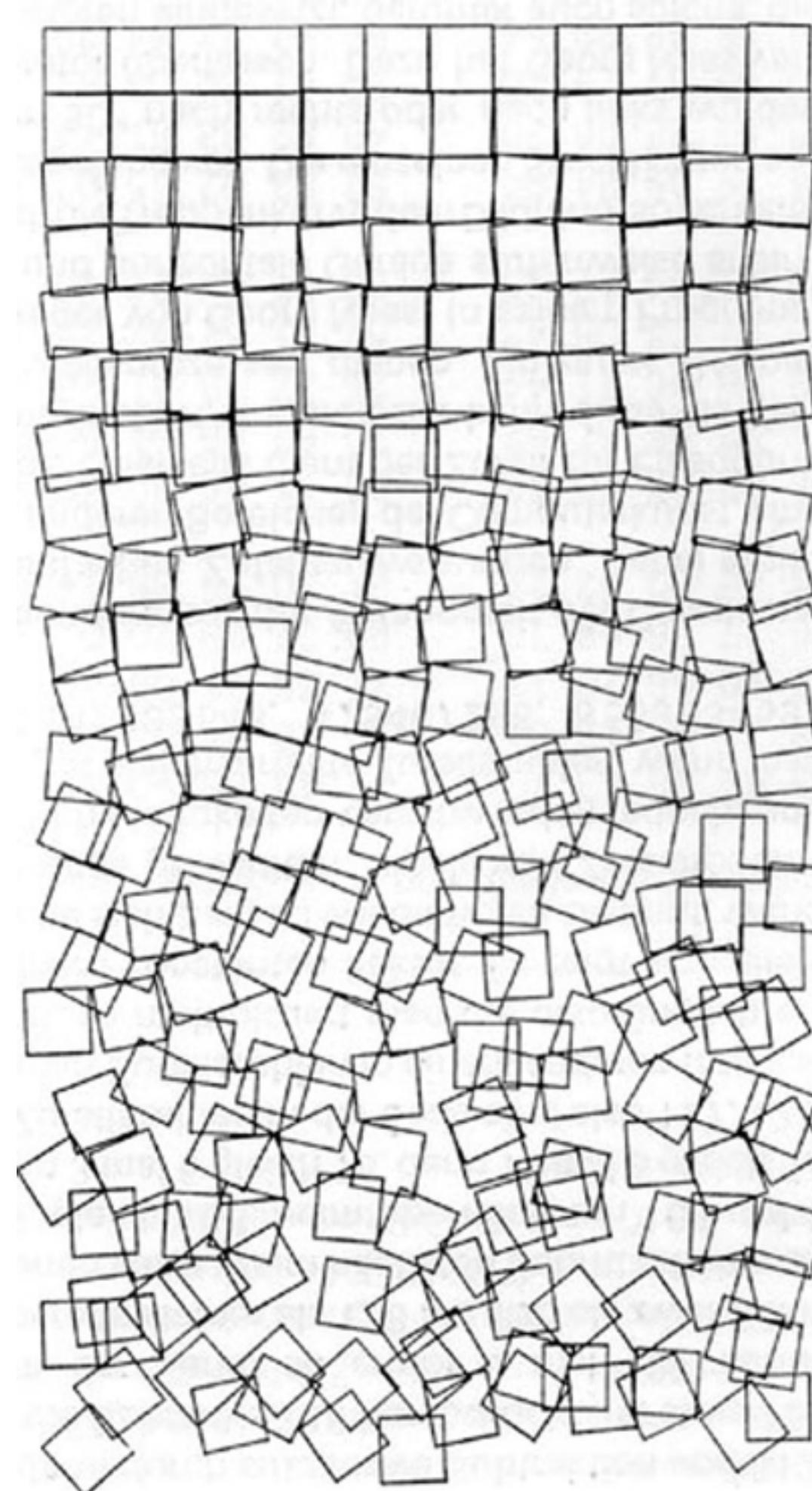
リサーチ課題補足

- ▶ Frieder Nake : 13/9/65 Nr. 2, 1965



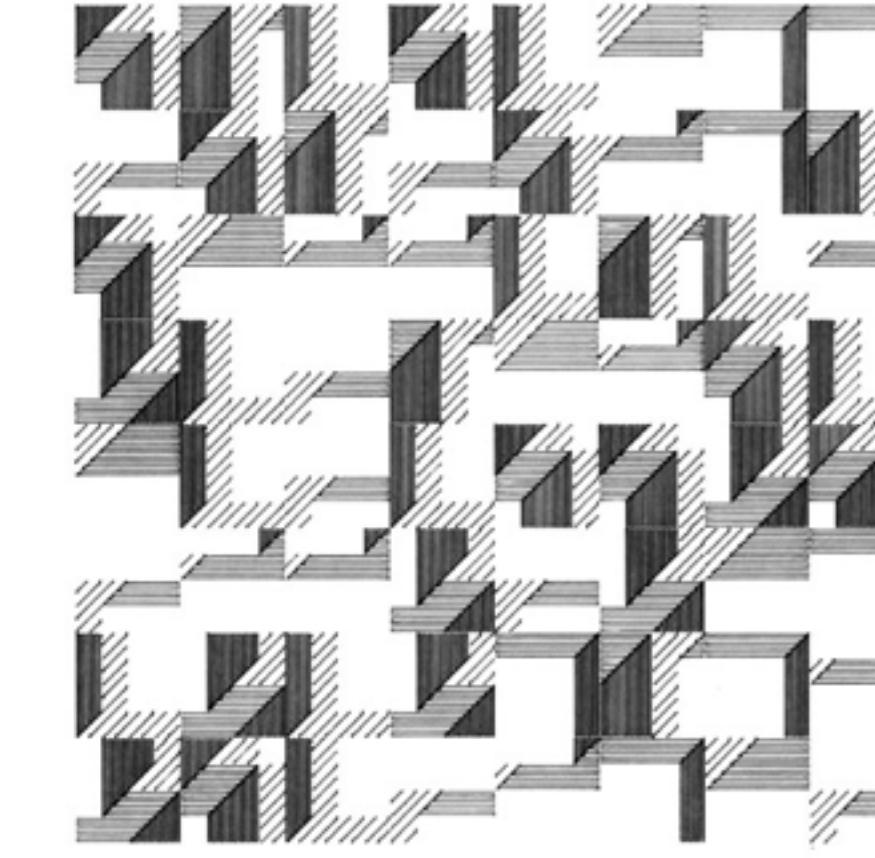
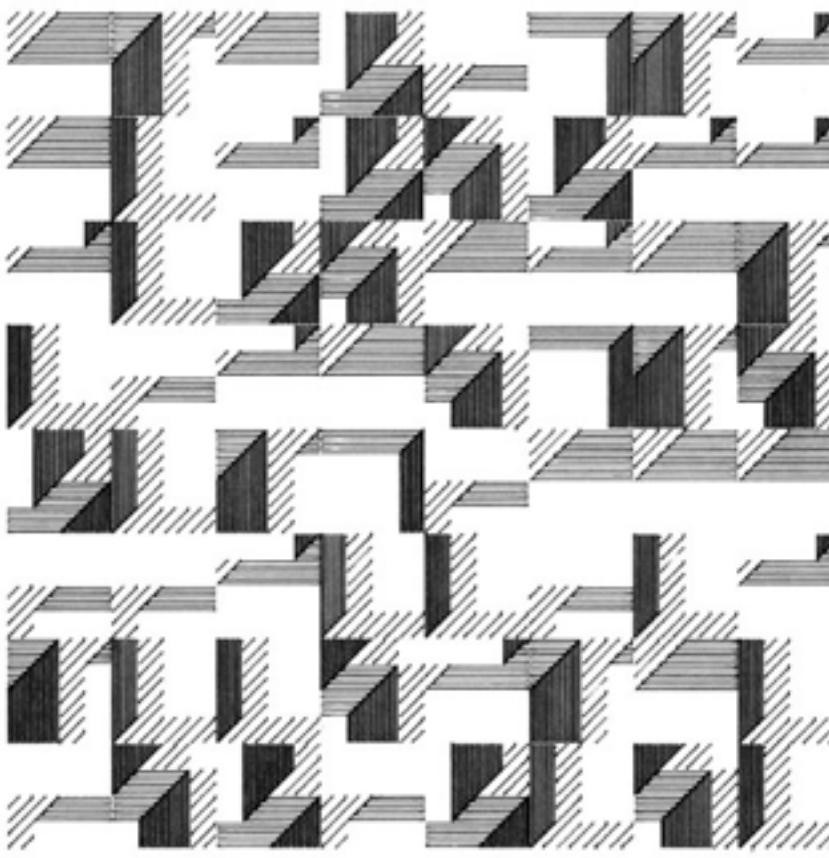
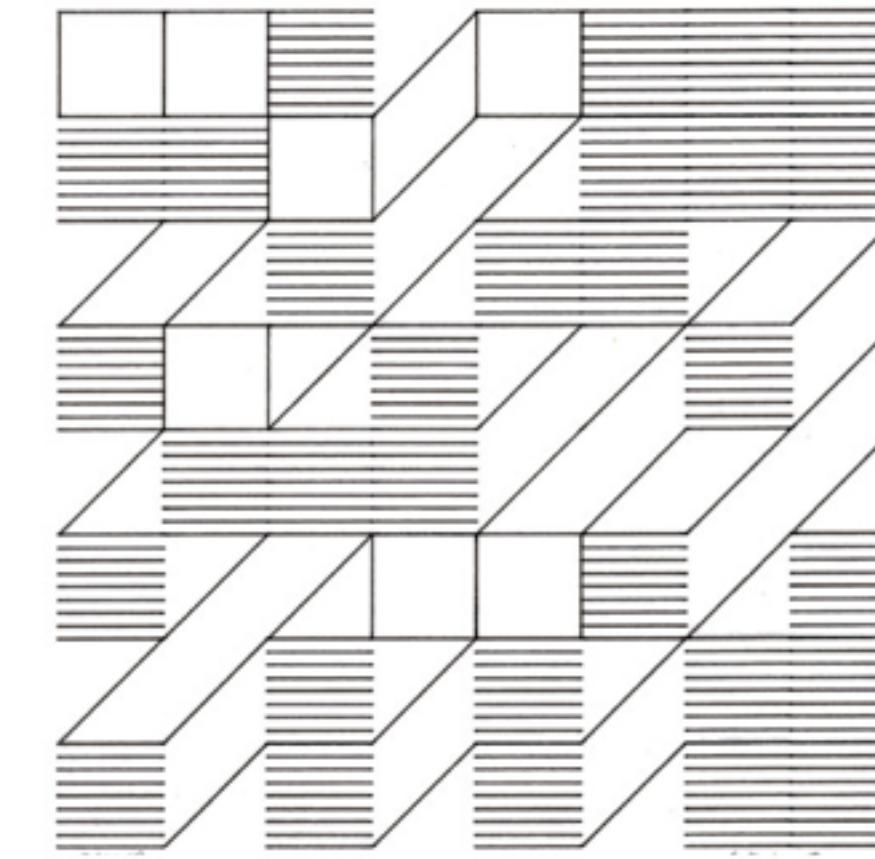
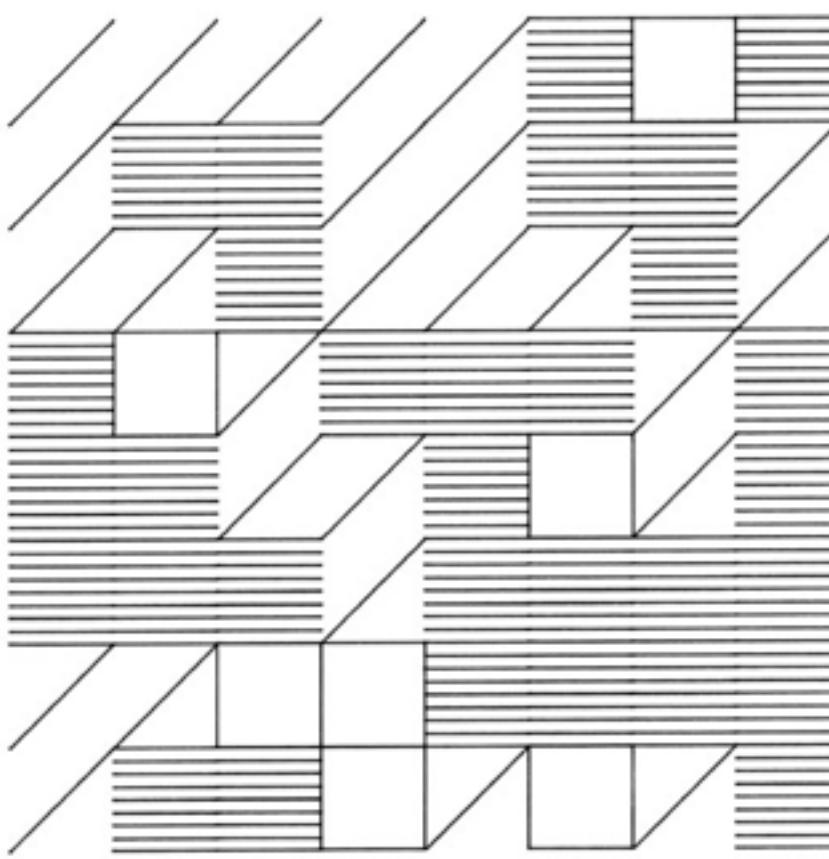
リサーチ課題補足

- ▶ Georg Nees: Wurfel-Unordnung (Cubic Disarray), 1968-71



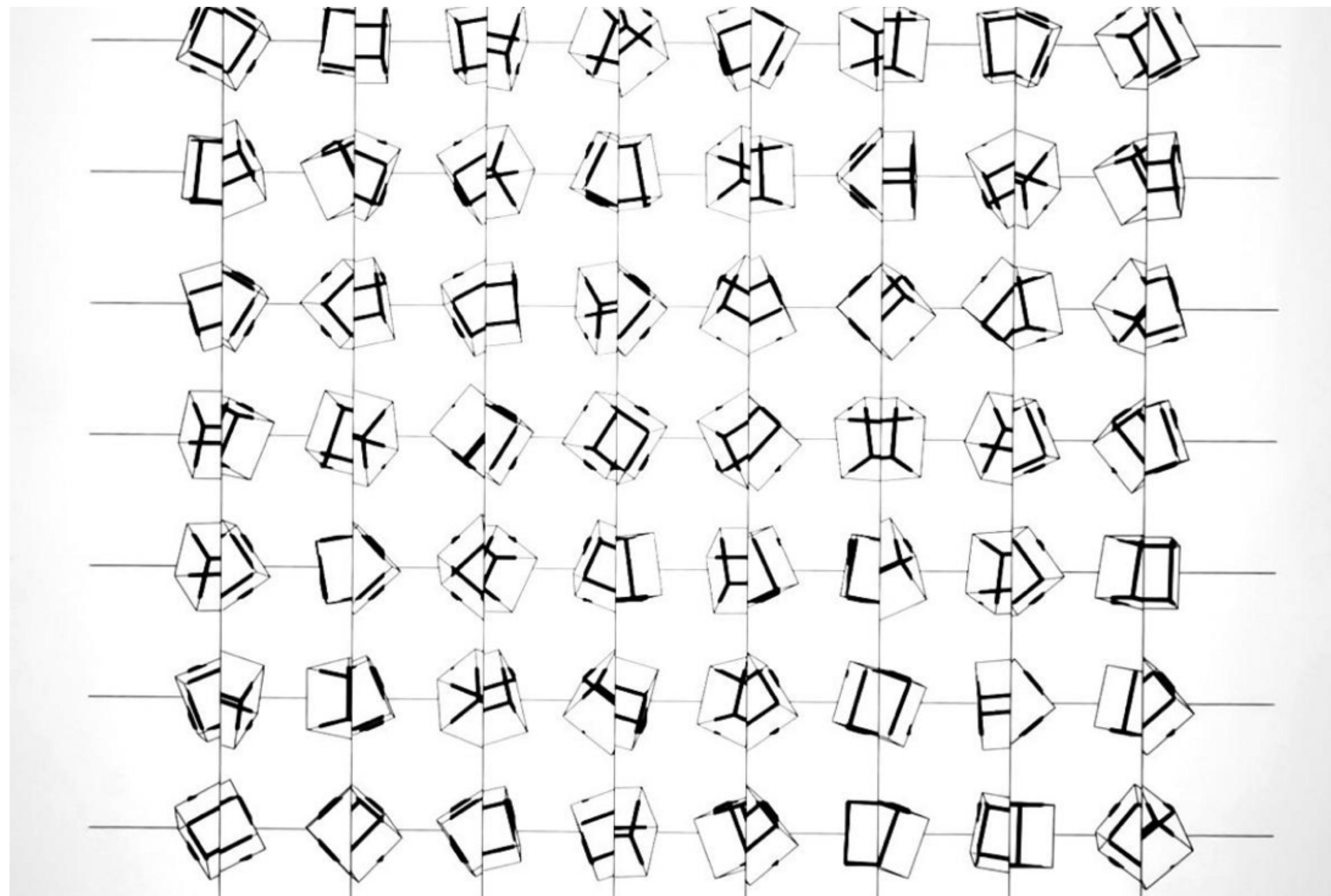
リサーチ課題補足

- ▶ Edward Zajec : RAM 10/3, 10/4, 2/6, 2/9, 1969



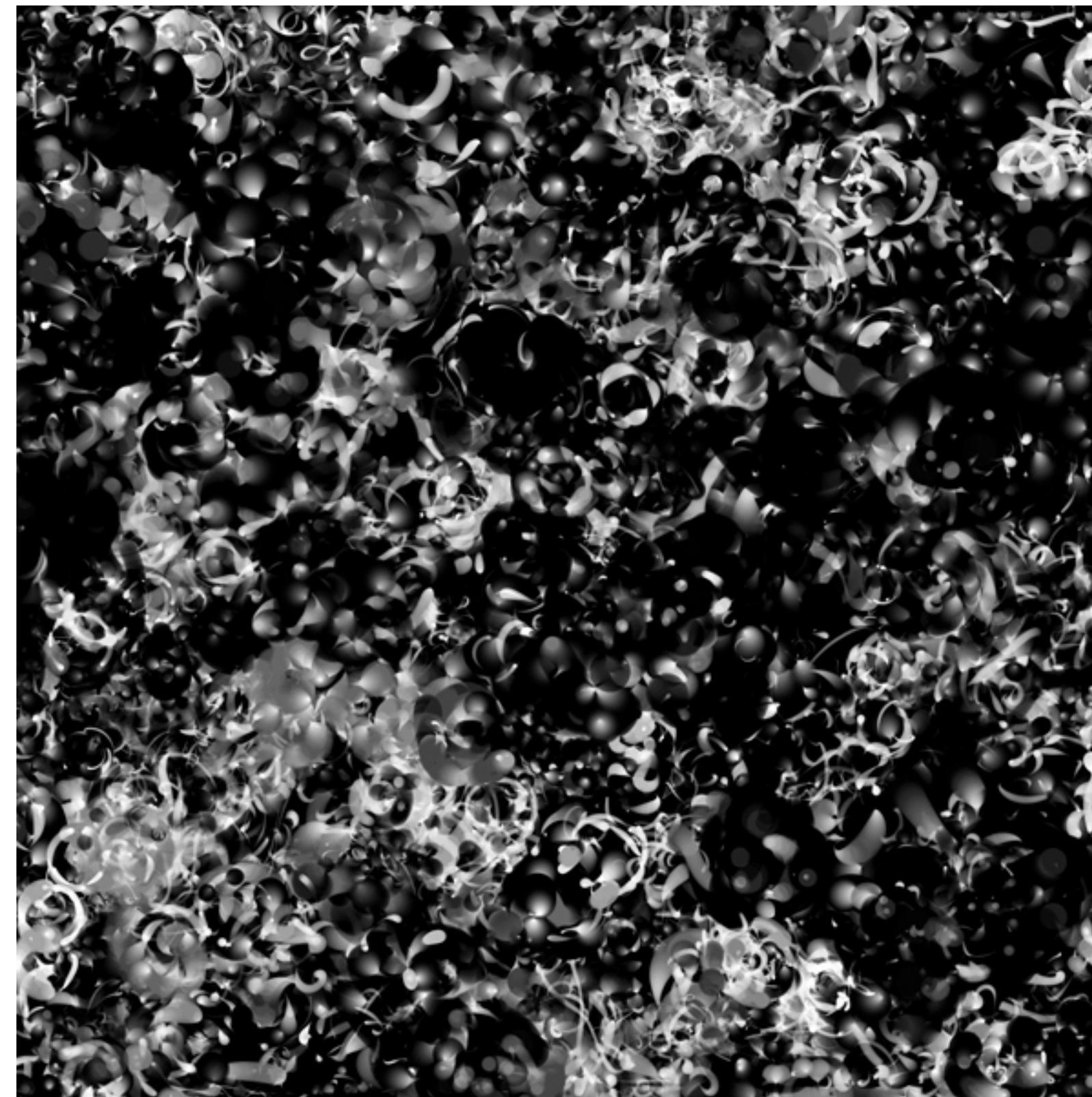
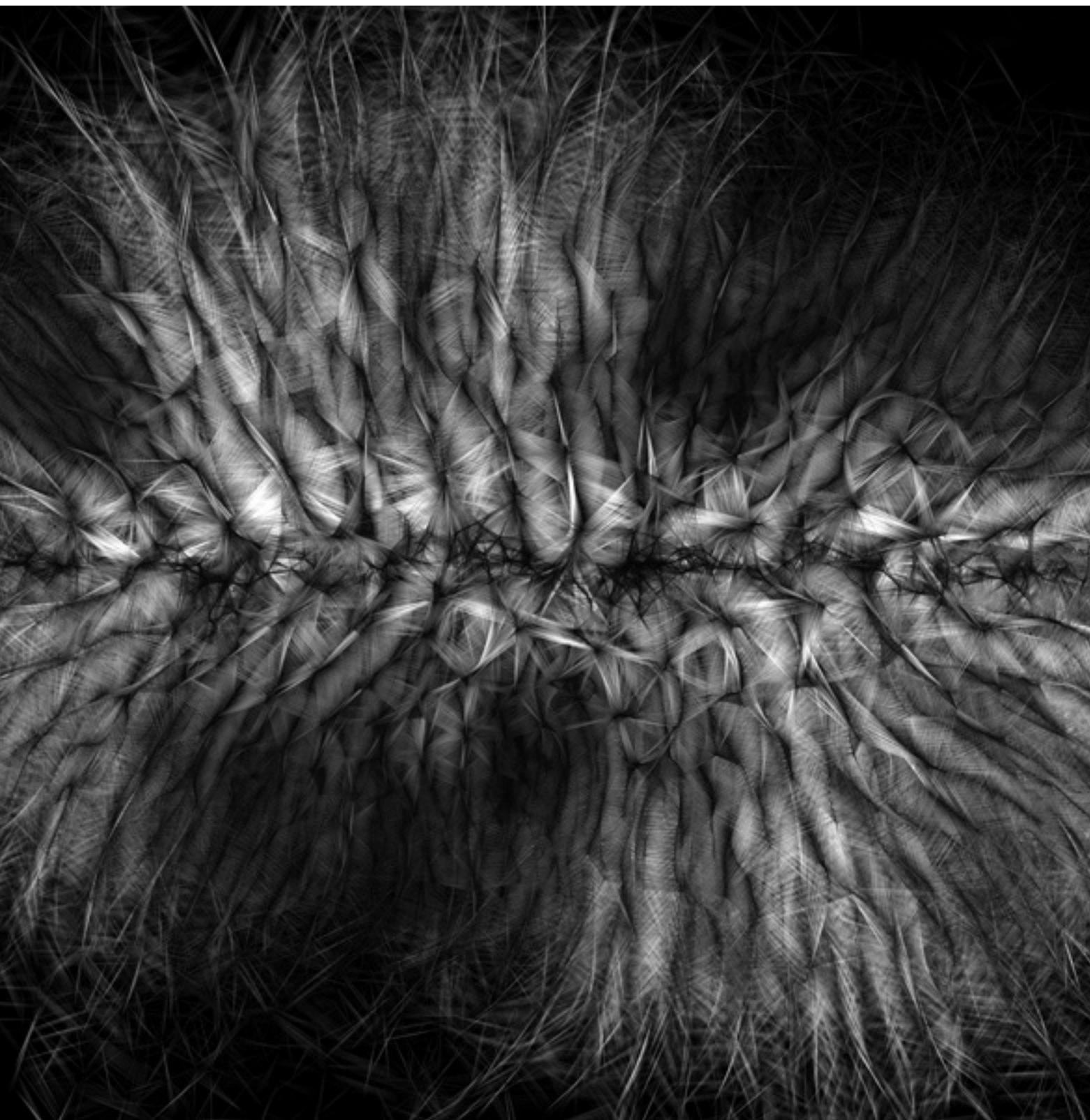
リサーチ課題補足

- ▶ Manfred Mohr : Cubic Limit 1972-77



リサーチ課題補足

- ▶ 現代でも、コードによる美学の探求は進められている
- ▶ Casey Reas : Process Compendium 2004-2010 <https://vimeo.com/22955812>



制作環境の準備： バージョン管理システム - Git と Github

Github

- ▶ Github Pageを運営する、Githubとは？
- ▶ 「Git」 というバージョン管理システムを利用した、ソフトウェア開発のためのWebベースのホスティングサービス
- ▶ Github社が運営
- ▶ 無償で利用可能（有料プランも有り）

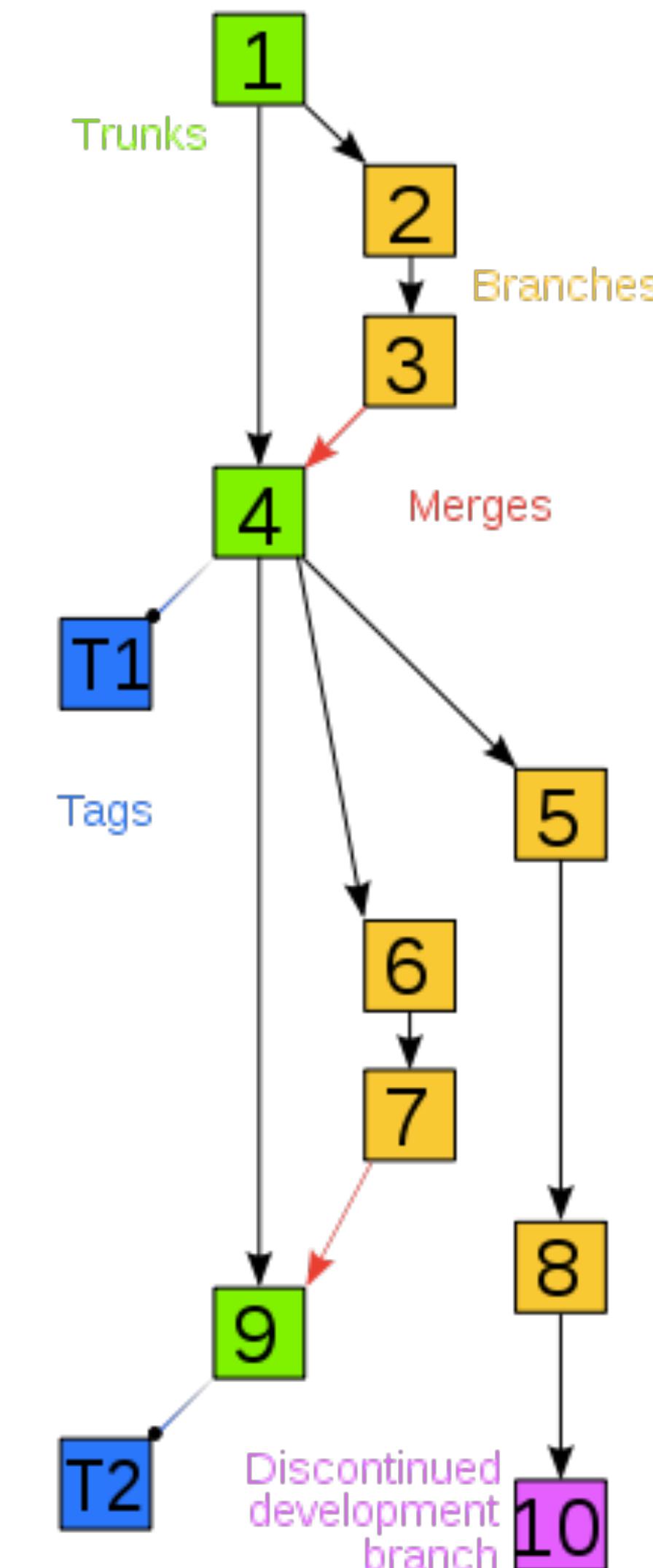


Git

- ▶ Git: バージョン管理システム
- ▶ プログラミングやWebデザインなど
- ▶ 頻繁に修正更新が必要な作業

- ▶ いつでも戻れるよう、作業の履歴を残しておきたい！
- ▶ しかし、全ての履歴を残すのには限界が…
- ▶ → バージョン管理システムを使用するように

- ▶ GNU arch、CVS、Subversionなど
- ▶ Gitはその中でも新しい仕組み



図解でGitを理解する

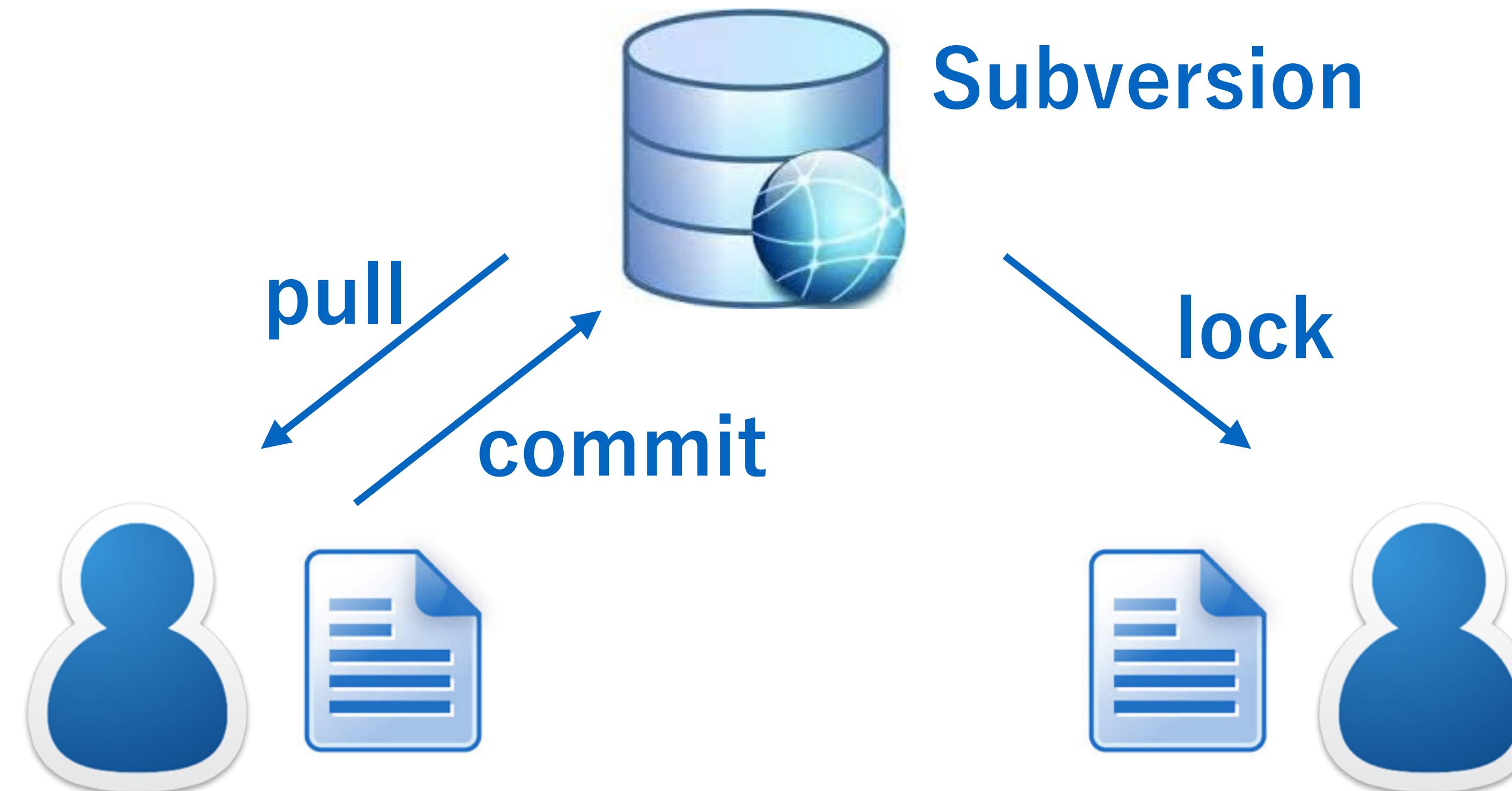
- ▶ Gitの特徴、分散リポジトリ
- ▶ Git以前に主流だったSubversionでは、中央リポジトリを共有していた
- ▶ 「リポジトリ」バージョン履歴を格納した「貯蔵庫」



Subversion

図解でGitを理解する

- ▶ 1. ファイルを取得(Pull)
- ▶ 2. 編集してコミット(Commit)
- ▶ 3. 他人から編集されないように、ロック(Lock)

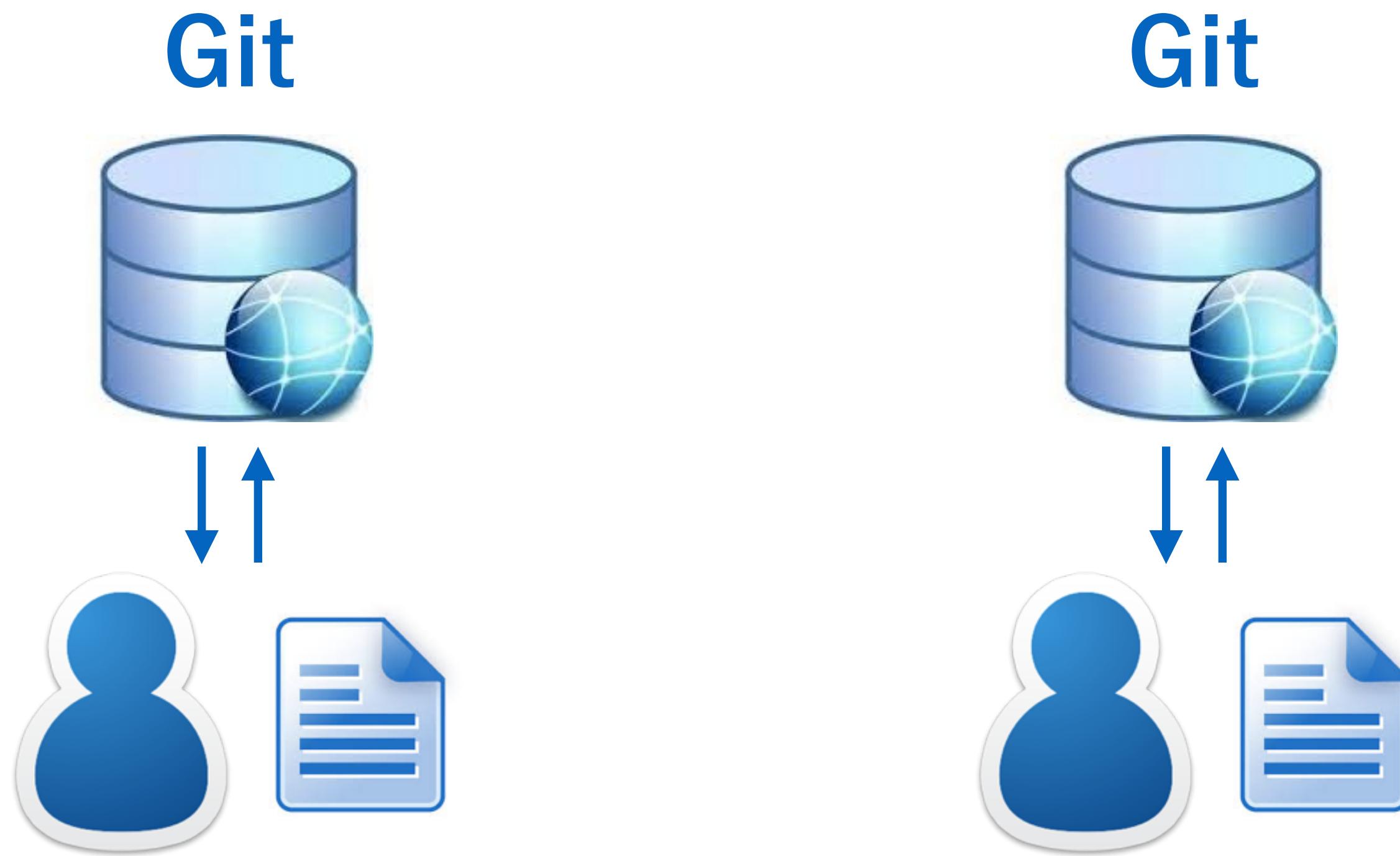


図解でGitを理解する

- ▶ Subversionのデメリット
- ▶ サーバーに接続できないと、新規のバージョンを取得できない
- ▶ リポジトリが壊れると、全てのバージョンが失なわれる…
- ▶ 誰かがロックしてしまうと、誰も変更できない

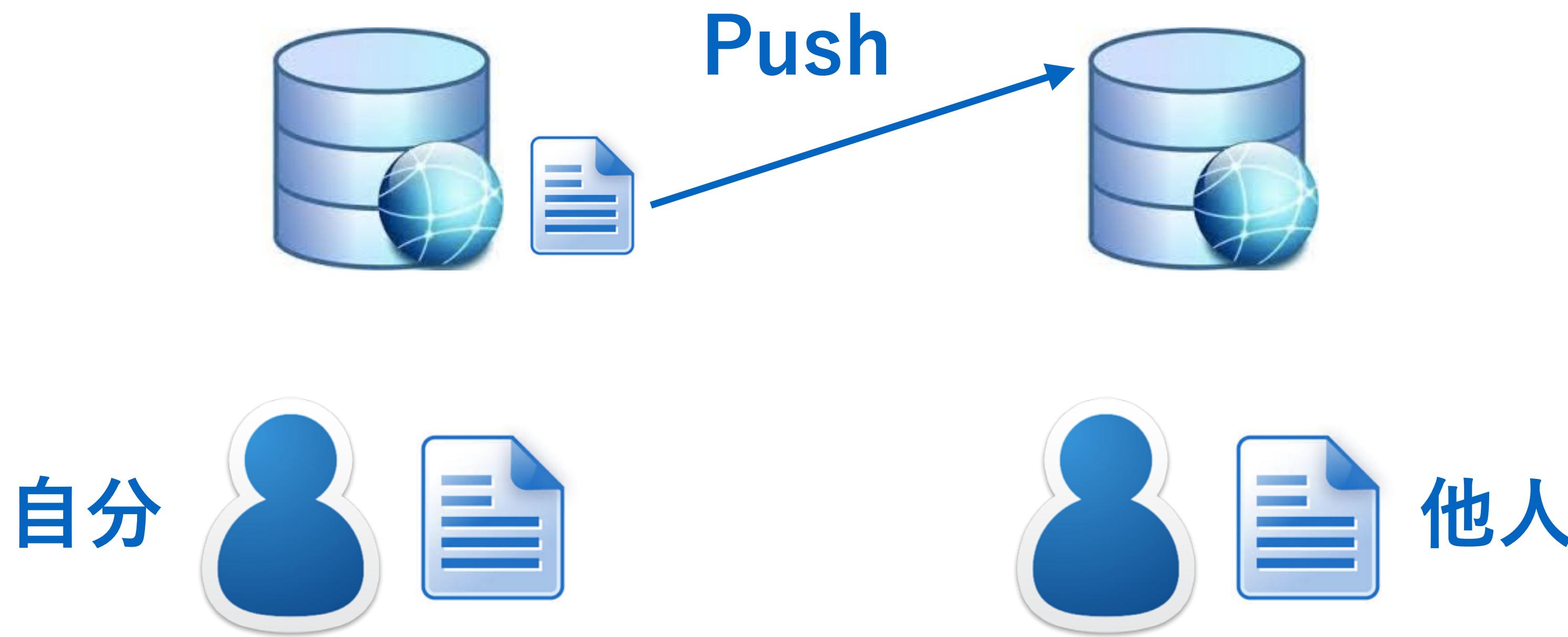
図解でGitを理解する

- ▶ Gitでは、リポジトリが分散している！
- ▶ ローカルで編集して、コミット(commit)



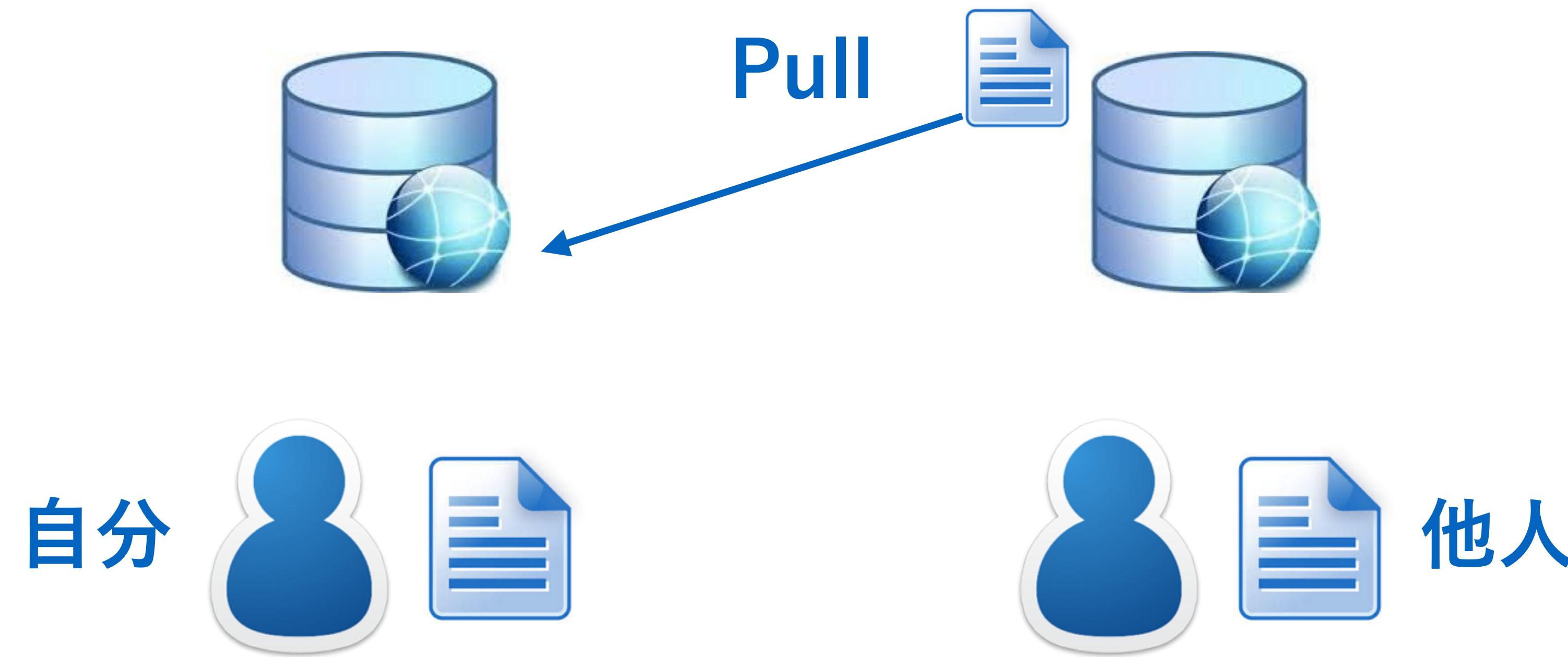
図解でGitを理解する

- ▶ 自分のコミットを、他のリポジトリに送りたい
- ▶ 「Push」する



図解でGitを理解する

- ▶ 他人のコミットを自分のリポジトリに取り込みたい
- ▶ 「Pull」する



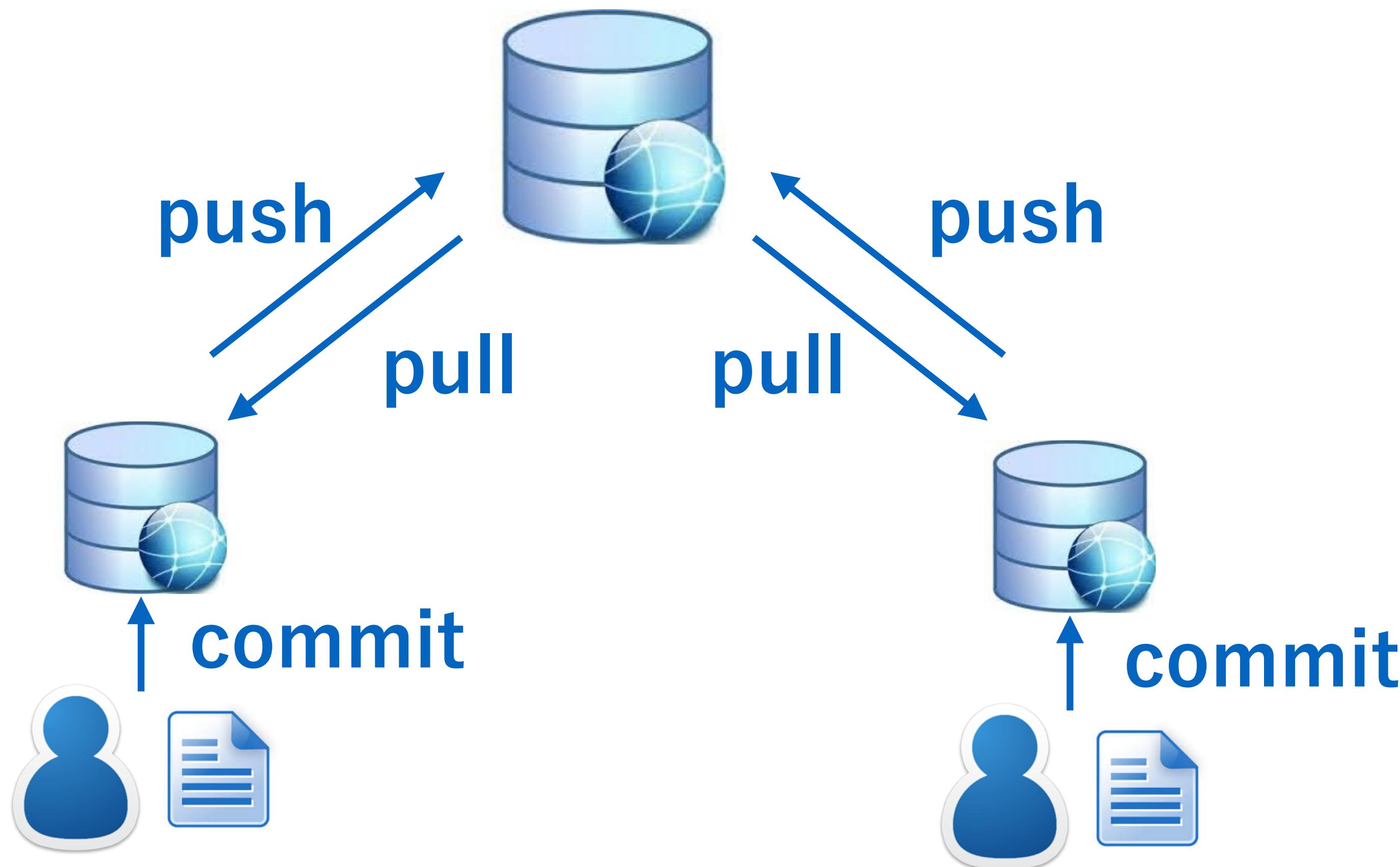
図解でGitを理解する

- ▶ さらに、複数のリポジトリで共有するリポジトリを使用することが可能！



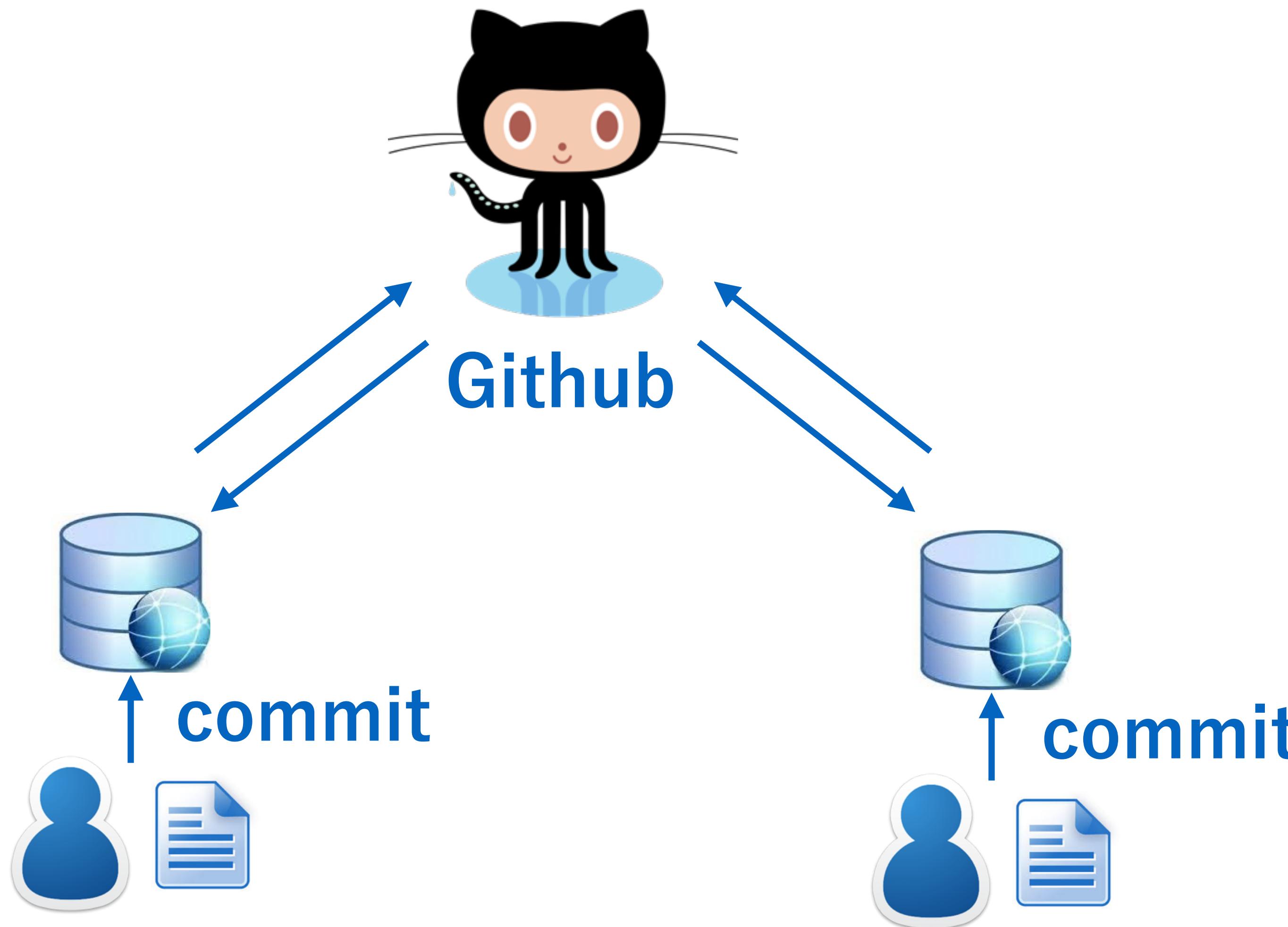
図解でGitを理解する

- ▶ まず、ローカルのリポジトリにcommit
- ▶ commitした結果を、共有リポジトリにPush
- ▶ 共有リポジトリからPullしてcommit履歴を取得



図解でGitを理解する

- ▶ Githubは、この共有リポジトリをWebにホスティングしている



より詳細なGitの解説

- ▶ サルでもわかるGit入門 <http://www.backlog.jp/git-guide/>

The screenshot shows the homepage of the 'Monkey Understands Git' guide. At the top, there's a navigation bar with a monkey icon, the title 'サルでもわかる Git 入門 ~バージョン管理を使いこなそう~', social sharing buttons (Twitter, Facebook), and a 'Table of Contents' button. Below the navigation is a main title 'サルでもわかる Git 入門' with a subtitle '～バージョン管理を使いこなそう～'. To the right is a cartoon monkey wearing a graduation cap, holding a book labeled 'Git'. Below the main title are three green buttons: '入門編 Click ➡' (for beginners), '発展編 Click ➡' (for users who have used Git), and '逆引き Git Click ➡' (for users who don't know what something means). A large text block in the center explains the purpose of the guide and the three levels of content. At the bottom, it says '電子書籍版も公開中!' and shows links for 'ePub版' and 'Kindle版'.

サルでもわかる Git 入門 ~バージョン管理を使いこなそう~

もくじ

入門編 発展編 逆引き Git

サルでもわかる Git 入門

～バージョン管理を使いこなそう～

ようこそ、サルでもわかるGit入門へ。
Gitをつかってバージョン管理ができるようになるために一緒に勉強していきましょう！
コースは3つ。Git初心者の方は「入門編」からどうぞ。
Gitを使った事がある方は「発展編」がおすすめです。
「あれ？何だっけ…？」という時は「逆引きGit」で調べて見て下さいね。

Git 初心者の方

Git を使った事ある方

あれ何だっけ? という時に

入門編 Click ➡

発展編 Click ➡

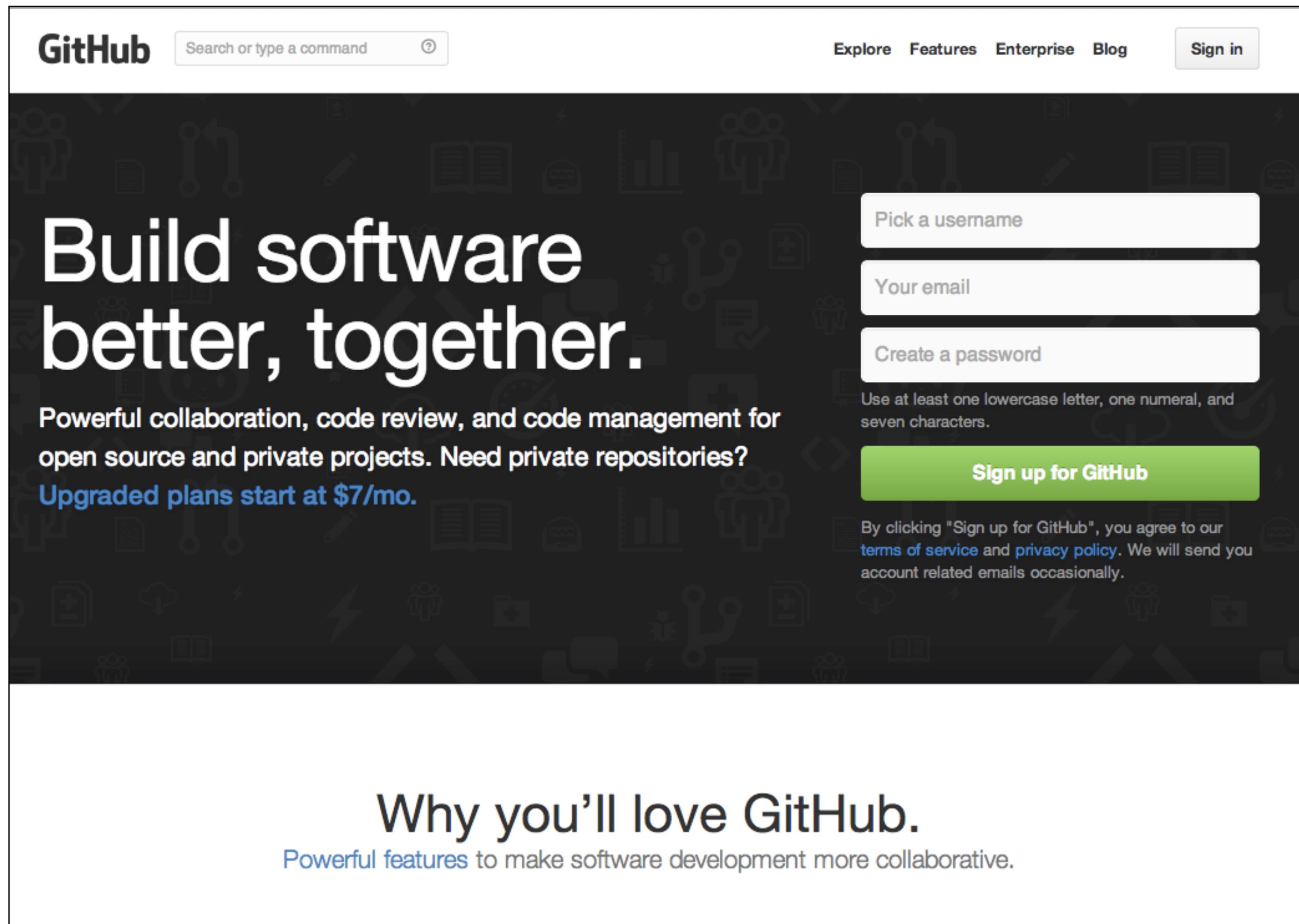
逆引き Git Click ➡

電子書籍版も公開中!

ePub版 (iBooksに直接取り込めます) Kindle版
for コンソール 6.6M for コンソール 2.8M

Githubアカウント作成

- ▶ まずは、Githubのアカウントを登録しましょう
- ▶ <https://github.com/>



クライアントアプリダウンロード&インストール

- ▶ Githubでは、GUIを使用した便利なアプリが用意されている
 - ▶ Mac版：<https://mac.github.com/>
 - ▶ Win版：<https://windows.github.com/>
- ▶ ダウンロードしてインストール



プロジェクトのリポジトリ

- ▶ Media lab. 2015のリポジトリ : https://github.com/tado/idd_medilab15

The screenshot shows the GitHub repository page for 'tado / idd_medilab15'. The repository has 2 commits, 1 branch (master), 0 releases, and 1 contributor (tado). The latest commit is 'update README' by tado, made 19 minutes ago. The repository description is 'Media lab. 2015, at Tama Art University IDD. — Edit'. The repository name is 'idd_medilab15'. The repository URL is 'https://github.com/tado/idd_medilab15'. The repository is public and has 0 stars and 0 forks.

Media lab. 2015, at Tama Art University IDD. — Edit

2 commits 1 branch 0 releases 1 contributor

branch: master / +

update README

tado authored 19 minutes ago latest commit eb81267269

README.md update README 19 minutes ago

README.md

idd_medilab15

多摩美術大学情報デザイン学科メディア芸術コース、Media lab 2015 プロジェクトページ。

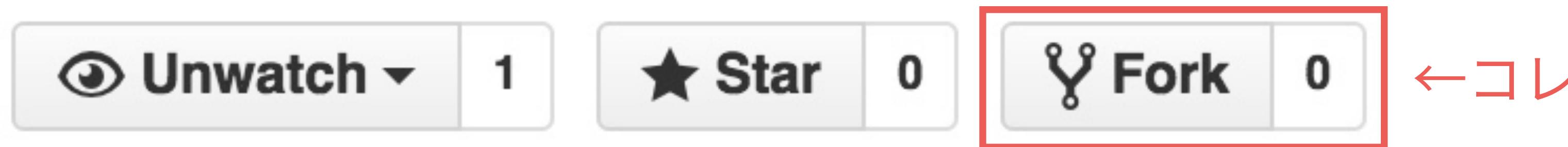
http://yoppa.org/tau_media15

概要

デジタル・ファブリケーションやカメラトラッキングを始めとするさまざまなツールを体験しながら OpenFrameworksプログラミングや映像/音響のコントロール手法を学び、いくつかのプロトタイプ的作品を完成させる。描く事を意識した「ドローイング・(口)ボット」、見ることを意識した「ヴィジョン・マシン」、飛ぶことを意識する「フライング・オブジェクト」のいずれかを中心に、3Dプリンターを使ったフートデザインやプロダクトデザイン、ソフトウェアアートなど、自分で課題を設定し、2つの作品を完成させて、オープンキャンパスなどで発表する。

プロジェクトのリポジトリをforkする

- ▶ 右上の「fork」ボタンを押す
- ▶ 授業用のリポジトリから、自分自身のリポジトリを分岐して生成できる!



- ▶ Gitの環境整備、今日のところはここまで!

何故openFrameworksを使うのか?

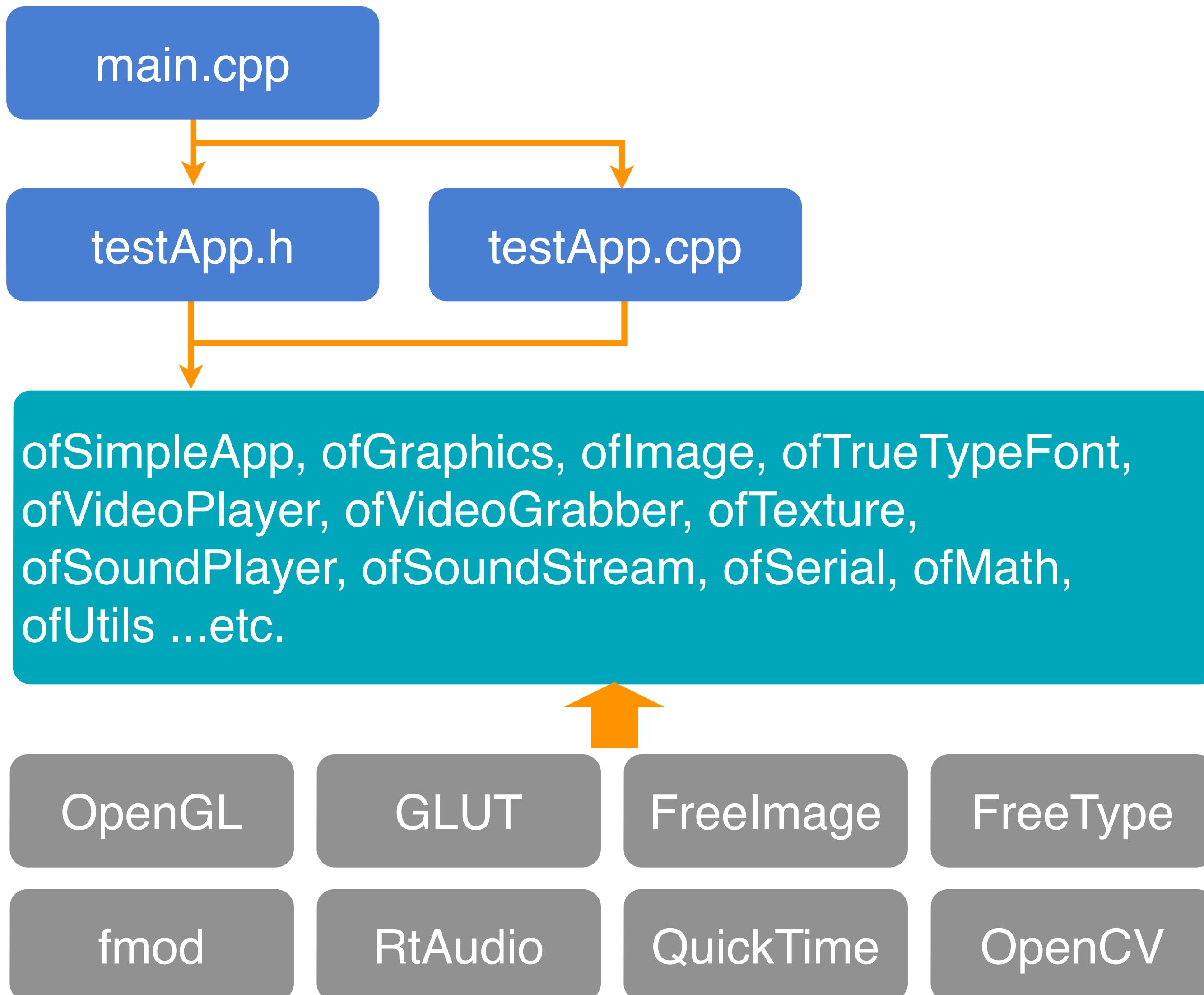
様々なメディアを駆使した作品を作りたい!!

→ 様々な技術に精通しなくてはならない
サウンド、ビデオ、フォント、画像解析...etc.

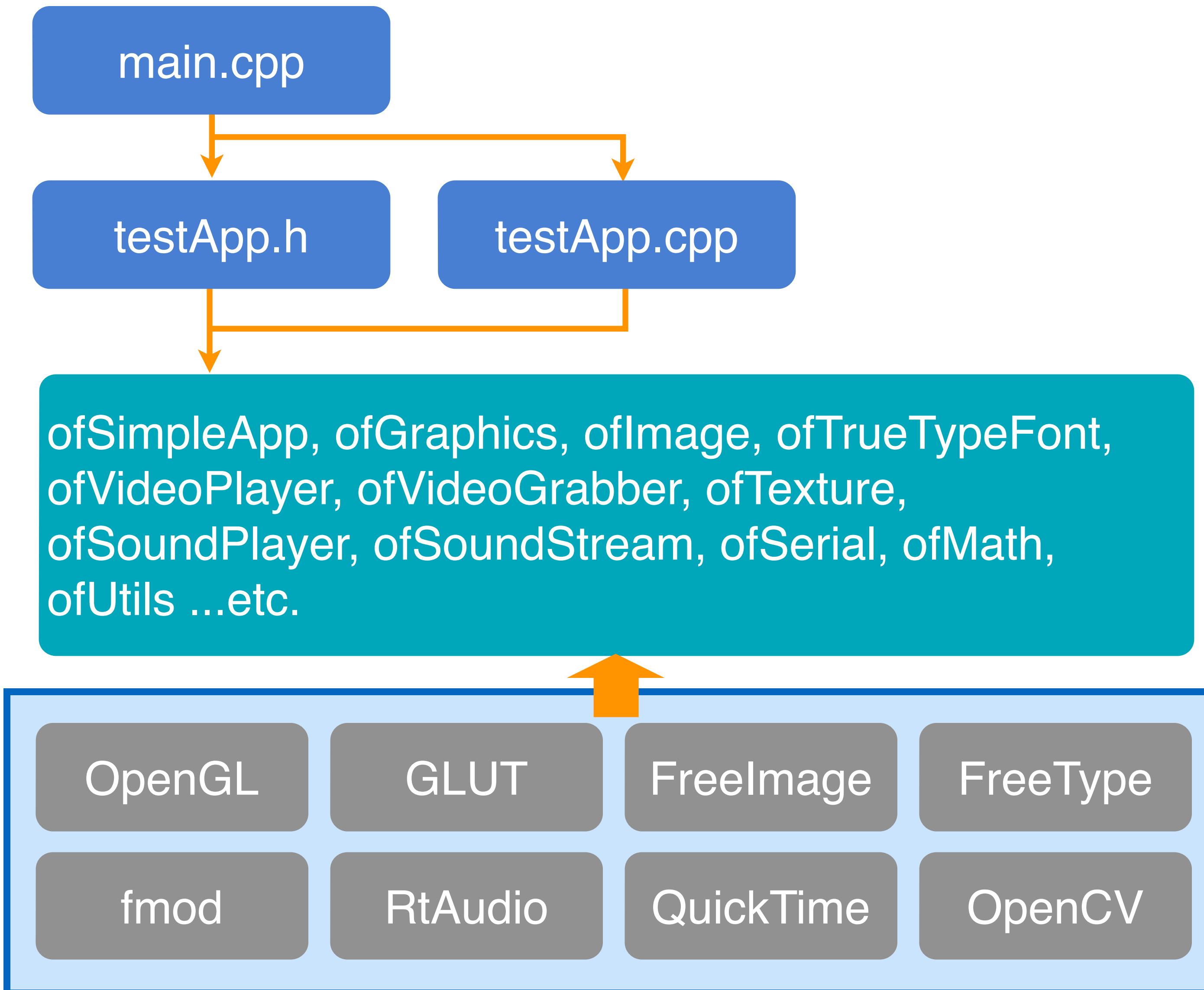
openFrameworksを利用すると…

既存の道具(ライブラリ)を設定なしに使用可能
→ 作品制作のための「糊」

開発のための「糊」

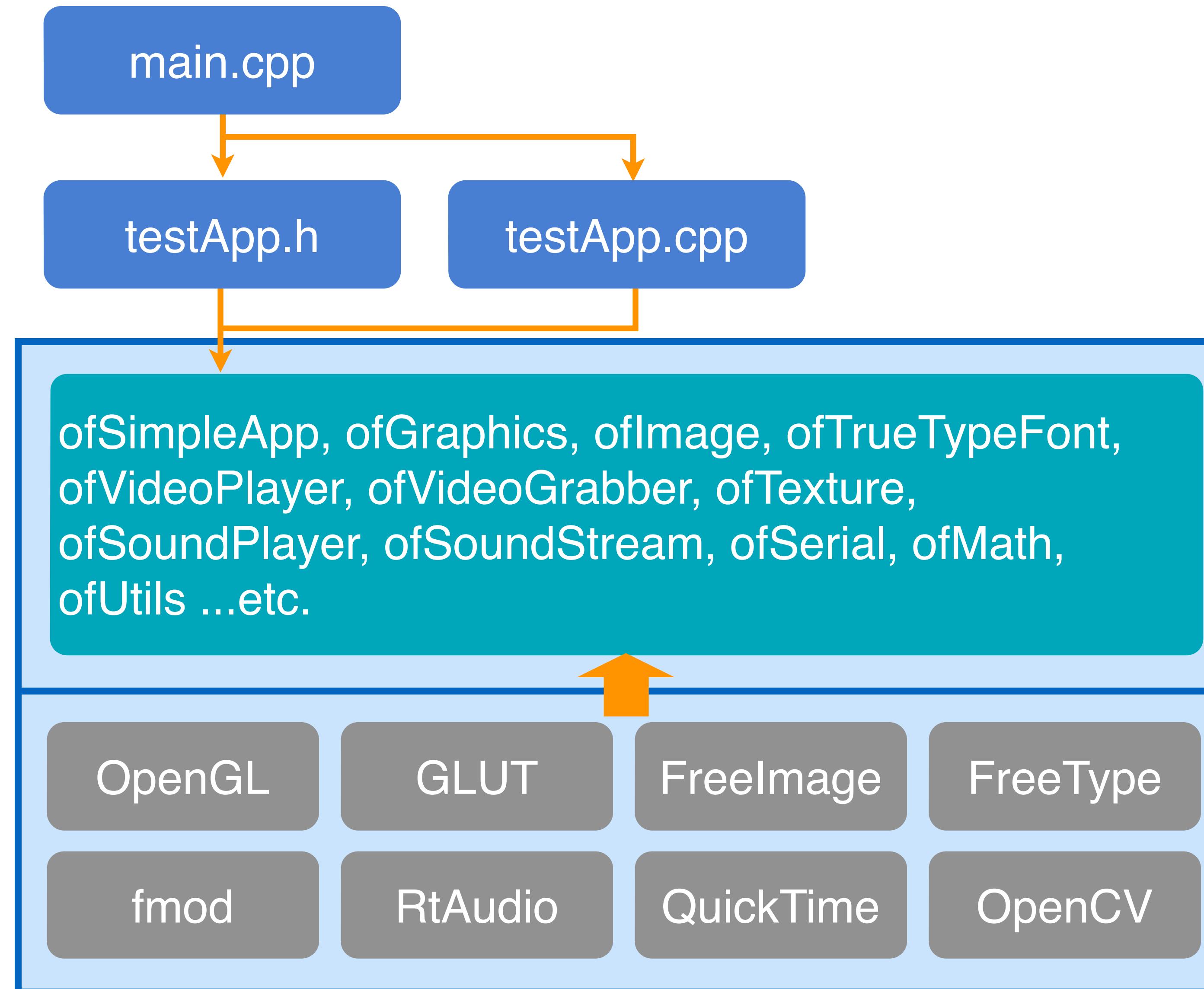


開発のための「糊」



既存のフリーなライブラリ群

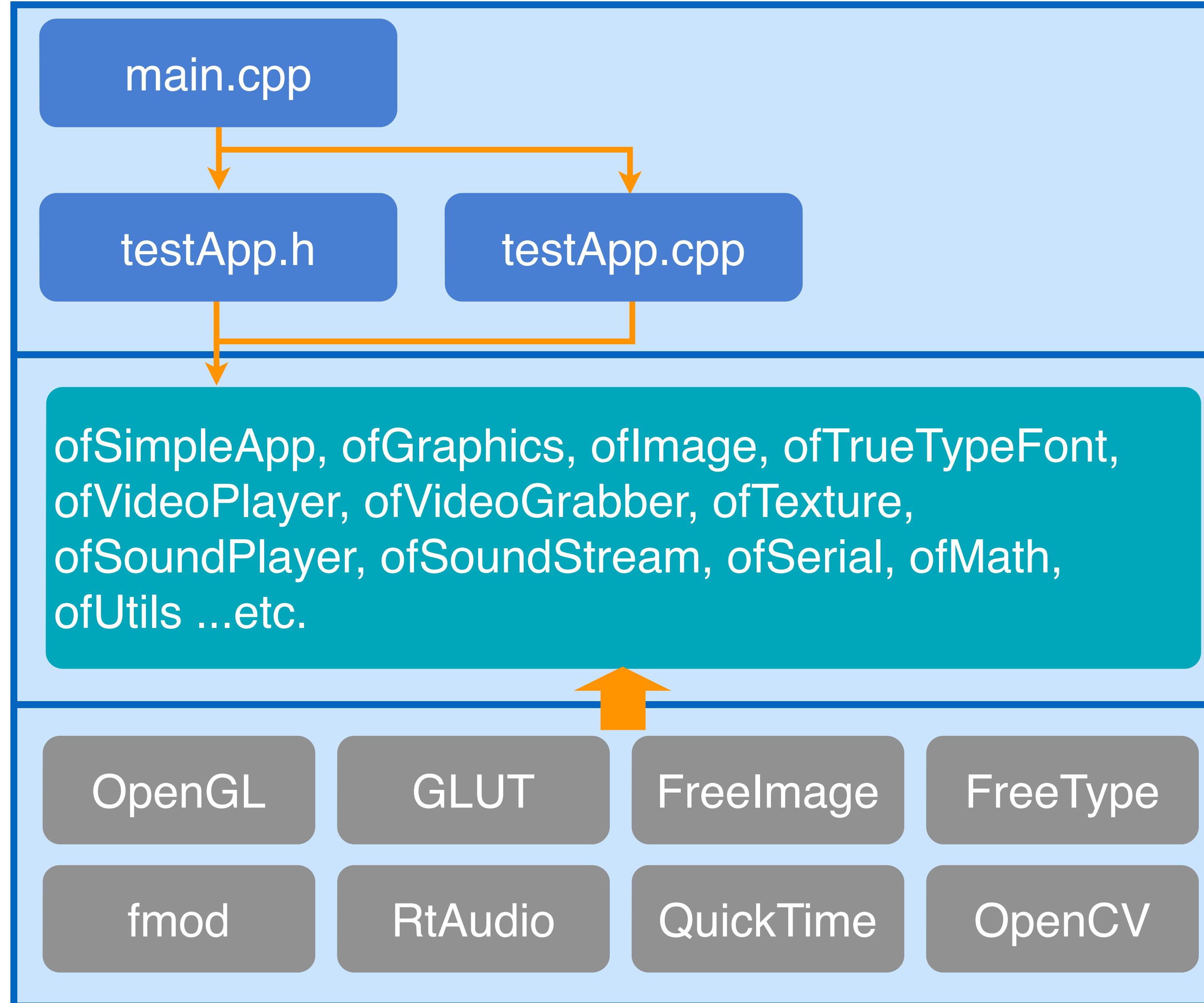
開発のための「糊」



openFrameworksの様々な機能

既存のフリーなライブラリ群

開発のための「糊」



実際に編集する部分

openFrameworksの様々な機能

既存のフリーなライブラリ群

openFrameworksに使用されているライブラリ

- ▶ できるだけオープンなライセンスのライブラリを採用
- ▶ グラフィクス：[OpenGL](#), [GLEW](#), [GLUT](#), [libtess2](#), [cairo](#)
- ▶ オーディオの入出力と分析：[rtAudio](#), [PortAudio](#), [OpenAL](#), [Kiss FFT](#), [FMOD](#)
- ▶ フォント：[FreeType](#)
- ▶ イメージの読み込みと保存：[Freelimage](#)
- ▶ 動画の再生と取込：[Quicktime](#), [GStreamer](#), [videoInput](#)
- ▶ 様々なユーティリティー：[Poco](#)
- ▶ コンピュータービジョン：[OpenCV](#)
- ▶ 3Dモデルの読み込み：[Assimp](#)

復習1：openFrameworks開発環境の構築

openFrameworksをダウンロード

- ▶ openFrameworksのダウンロードページより
- ▶ <http://www.openframeworks.cc/download>

The screenshot shows the 'download' section of the openFrameworks website. At the top, there's a navigation bar with links for 'about', 'download' (which is highlighted in pink), 'documentation', 'tutorials', 'gallery', 'community', 'development', 'forum', 'addons', 'github', 'mailing list', 'IRC', and 'blog'. Below the navigation, the word 'download' is centered in a large, bold, dark font. Underneath it, the version '0.8.3' is mentioned as the most recent release, noting new features, bugs, and compatibility issues. A link to the 'changelog' is provided. Further down, there are sections for different platforms: 'OSX', 'linux', 'windows', 'mobile', 'ios', and 'android'. Each platform section contains links for 'download', 'IDE setup guide' (with specific IDEs like 'xcode' or 'code::blocks'), and sometimes additional guides like 'IDE setup guides' or 'IDE setup guide ADT'.

about [download](#) documentation tutorials gallery community development
> forum > addons > github > mailing list > IRC > blog
english / [japanese](#)

download

0.8.3 is the most recent release. It has a lot of new features, new interfaces, and probably some new bugs too. 0.8.3 is not 100% compatible with older projects. Please see the [changelog](#) to get an overview of the differences between versions.

To use openFrameworks you will need an IDE, and the setup guide for your platform can walk you through this. Please post any bugs on the [issues](#) page, and post to the [forum](#) if you have any other questions. openFrameworks is distributed under the [MIT License](#).

OSX	linux	windows
download openFrameworks for xcode IDE setup guide xcode	download openFrameworks for code::blocks code::blocks (64 bit) IDE setup guide code::blocks IDE setup guide eclipse	download openFrameworks for code::blocks visual studio IDE setup guides code::blocks visual studio

mobile	ios	android
<i>openFrameworks for mobile platforms supports the same features as the desktop versions plus mobile specific features like accelerometer, compass, gps...</i>	osx only download openFrameworks for xcode IDE setup guide xcode	download openFrameworks for eclipse IDE setup guide ADT

openFrameworksをダウンロード

- ▶ Mac OS Xの方
- ▶ osx - xcode版をダウンロード

- ▶ Windowsの方
- ▶ windows - code::blocks版をダウンロード

openFrameworksの開発環境

- ▶ openFrameworksには、ProcessingやFlashなどのように専用の開発環境があるわけではない
- ▶ それぞれのOSに応じた、C++の開発環境を使用する
- ▶ Mac OSXの場合 - XCode
- ▶ Windowsの場合 - Code::Blocks もしくは Visual Studio 2010
- ▶ Linuxの場合 - Code::Blocks

openFrameworksの開発環境

- ▶ Xcodeを入手するには → App Store.app を利用する
- ▶ App Storeで「Xcode」で検索



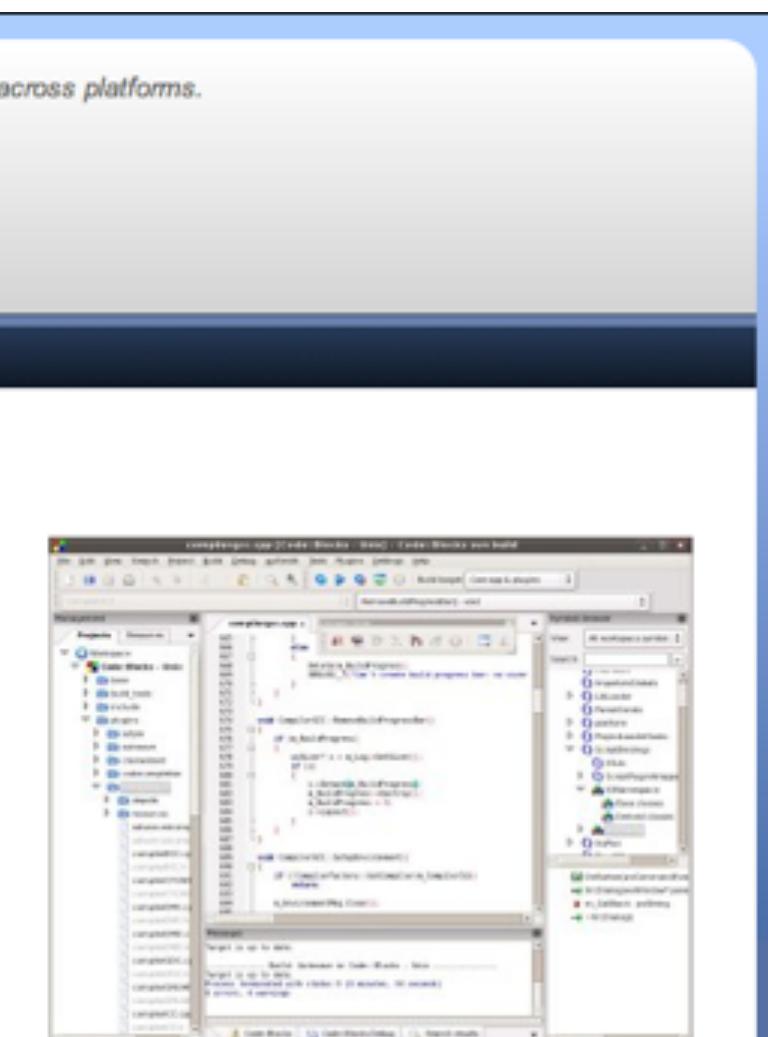
openFrameworksの開発環境

- ▶ XCodeのバージョン
- ▶ OS X 10.7 Lion → Xcode 4.x
- ▶ OS X 10.8 Mountain Lion、 10.9 Mavericks → Xcode 5.x
- ▶ OS X 10.10 Yosemite → Xcode 6.x
- ▶ 参考: [XcodeとMac OS XのVersionの組み合わせごとのSDKとSimulatorの比較 - Qiita](#)



openFrameworksの開発環境

- ▶ Windowsの方には、code::blocks がおススメ!
- ▶ フリーウェア
- ▶ <http://www.codeblocks.org/>



The screenshot shows the official website for Code::Blocks. At the top, there's a navigation bar with links for Home, Features, Downloads, Forums, and Wiki. The main content area features a large image of the Code::Blocks IDE interface, which includes a project tree on the left, code editors in the center, and toolbars at the bottom. Below the image, there's a section titled "The open source, cross platform, free C++ IDE." with descriptive text and links to forums and documentation. A sidebar on the left contains links for Home, Features, Screenshots, Downloads, Plugins, User manual, Licensing, and Donations. Another sidebar lists "Quick links" such as FAQ, Wiki, Forums, and BugTracker. At the bottom, there are logos for GPL, wxWidgets, W3C CSS, and GetFirefox.

Code::Blocks

Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms.

Main

- Home
- Features
- Downloads
- Forums
- Wiki

The open source, cross platform, free C++ IDE.

Code::Blocks is a *free C++ IDE* built to meet the most demanding needs of its users. It is designed to be very extensible and fully configurable.

Finally, an IDE with all the features you need, having a consistent look, feel and operation across platforms.

Built around a plugin framework, Code::Blocks can be extended with *plugins*. Any kind of functionality can be added by installing/coding a plugin. For instance, compiling and debugging functionality is already provided by plugins!

We hope you enjoy using Code::Blocks!

The Code::Blocks Team

Repository has moved to SF.NET

Written by MortenMacFly
Thursday, 28 March 2013 04:23

Our SVN repository has moved to SourceForge.NET. Please update the URL of your working copy to <svn://svn.code.sf.net/p/codeblocks/code/trunk>.

Last Updated on Thursday, 28 March 2013 04:25

12.11 mirror is up!

Written by MortenMacFly
Thursday, 13 December 2012 17:36

We have mirrored all file releases at our webpage at SourceForge, too. Hopefully this will resolve some download issues reported. Sorry for the inconvenience.

Last Updated on Thursday, 13 December 2012 17:39

復習2：サンプルを実行してみよう!!

サンプルを実行してみよう!!

- ▶ openFrameworksには、大量のサンプルが付属している
- ▶ どれも素晴らしいサンプル!



3d



addons



communication



empty



events



gl



graphics



gui



math



sound



utils



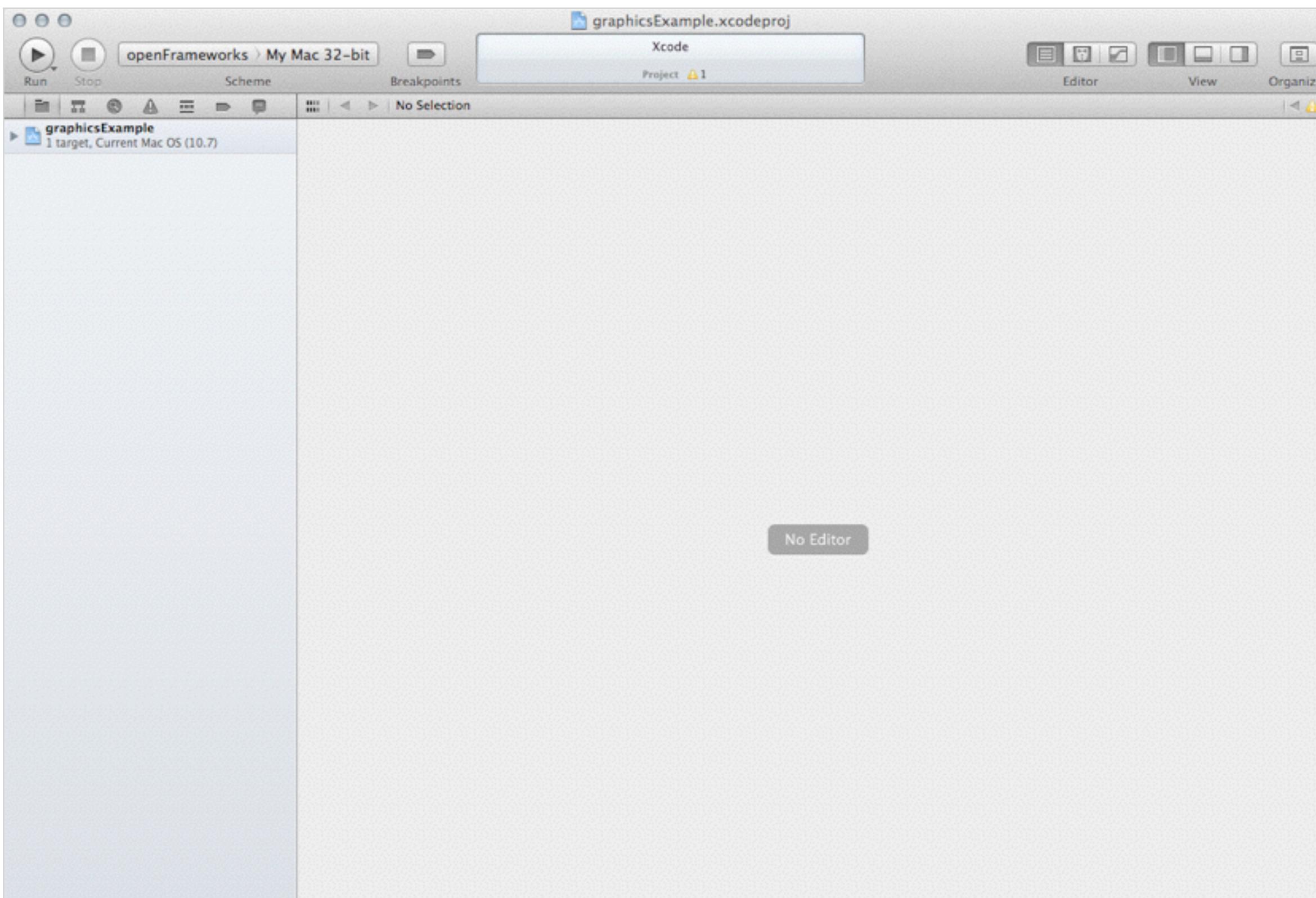
video

サンプルを実行してみよう!!

- ▶ 3d - 3次元表現いろいろ
- ▶ addons - 拡張機能のサンプル
- ▶ communication - 外部デバイスとの通信(シリアル通信)
- ▶ empty - 制作のテンプレートとなる「空」サンプル
- ▶ events - イベント(プログラムへの外部からのアクション)処理
- ▶ gl - OpenGLの活用(Shader、VBO、カメラなど)
- ▶ graphics - グラフィクスプログラミング
- ▶ math - 数式による表現、ノイズ、周期など
- ▶ sound - 音響生成、サウンドファイルの再生
- ▶ utils - 補助的な機能の例
- ▶ video - 動画の再生、カメラからの入力

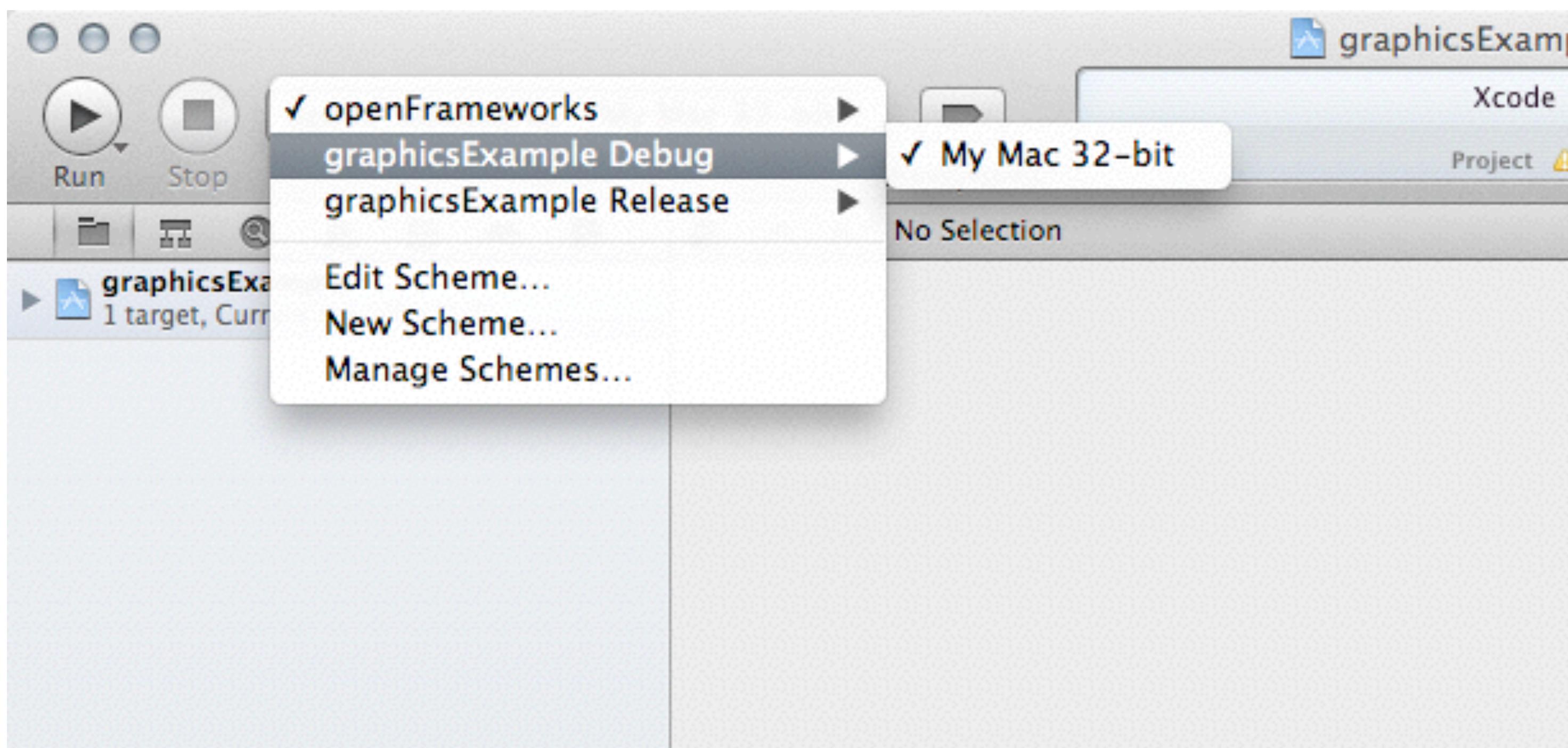
サンプルを実行してみよう!!

- ▶ 「of_v0.8.3_osx_release/examples/」以下のフォルダ内
- ▶ プロジェクトファイル「.xcodeproj」を開く
- ▶ 例えば、graphicsExample.xcodeproj
- ▶ プロジェクトファイルを開くと、自動的にXcodeが起動する



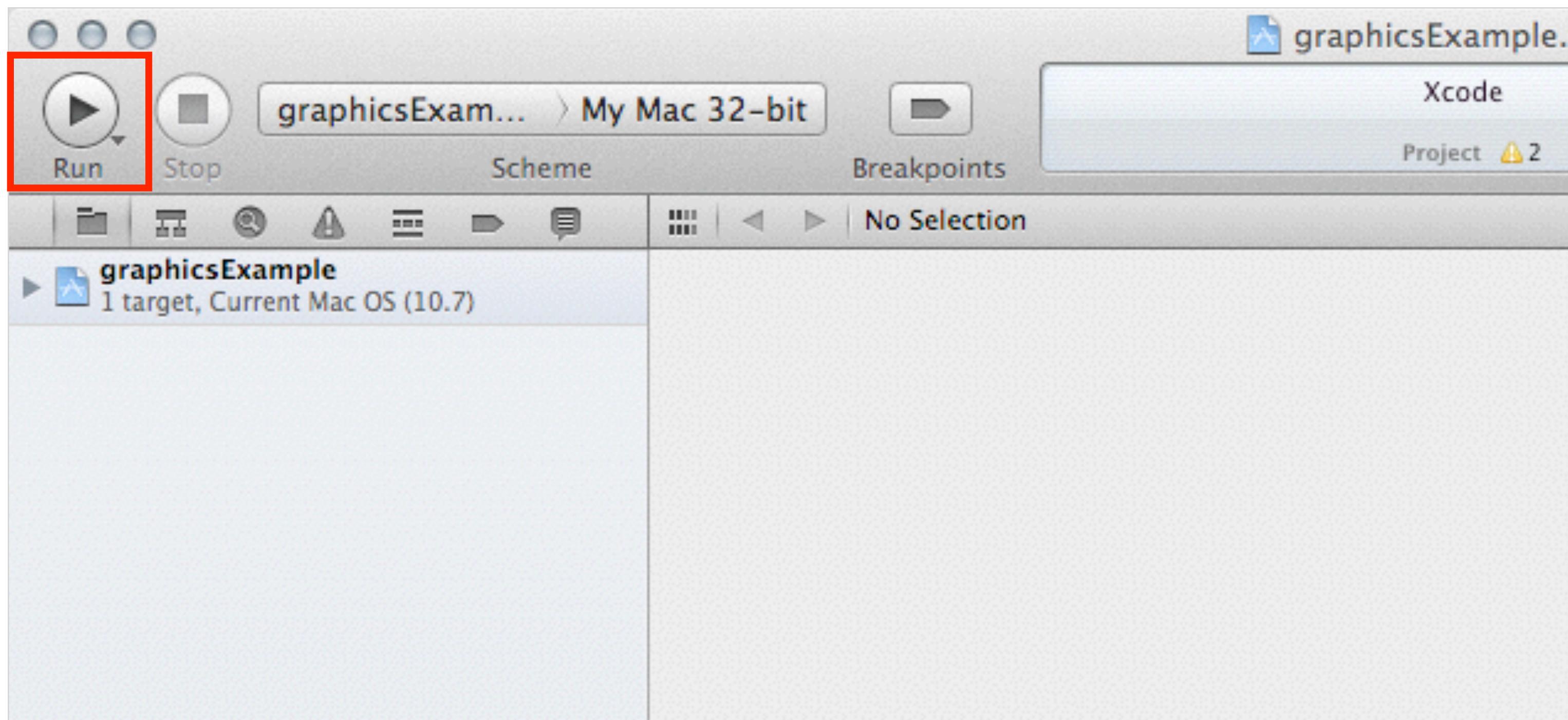
サンプルを実行してみよう!!

- ▶ プログラムを実行するには
- ▶ まず、Schemeのプルダウンより「サンプル名 Debug」を選択する



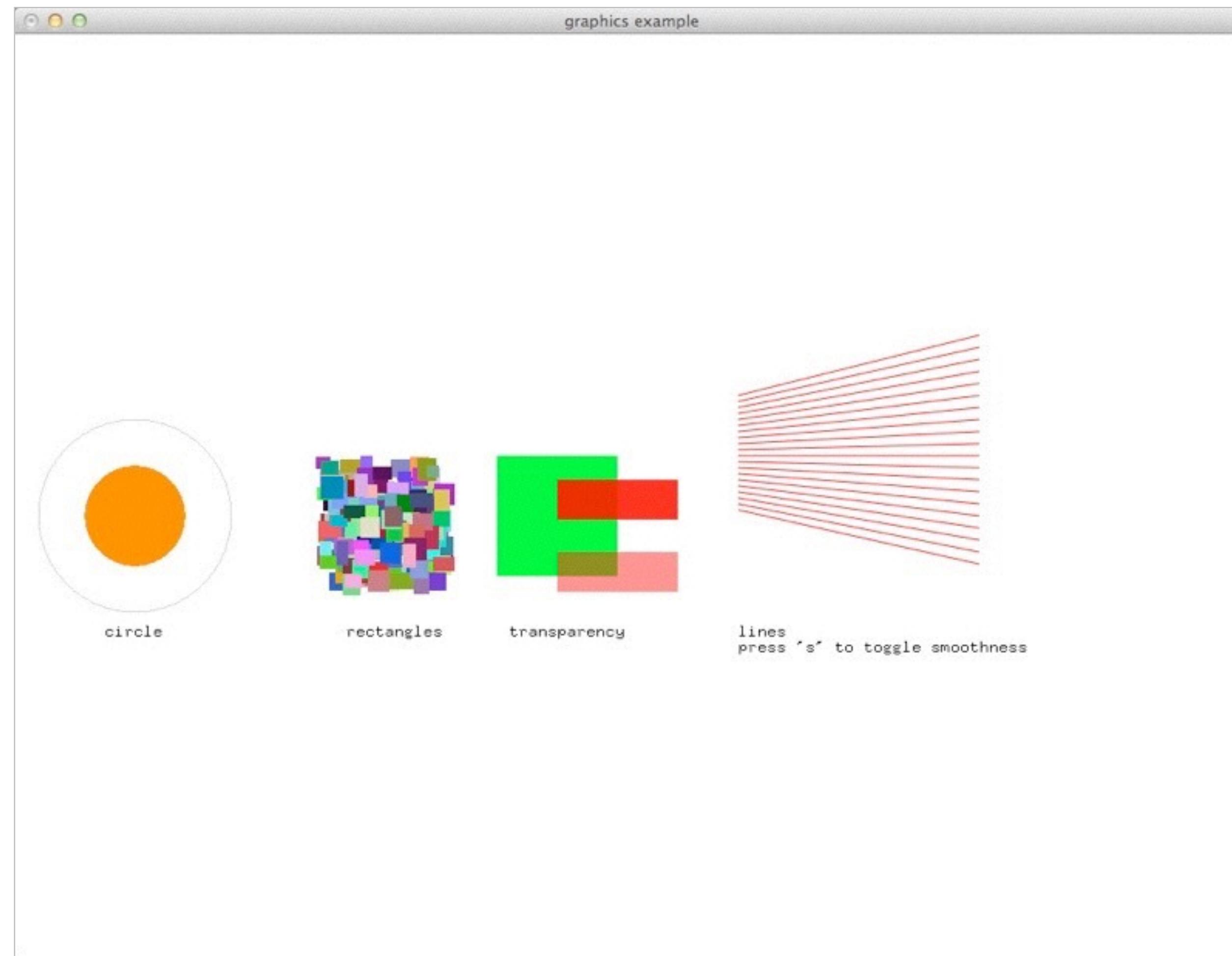
サンプルを実行してみよう!!

- ▶ 左上の「Run」ボタンを押す



サンプルを実行してみよう!!

▶ 実行例 : graphics example



実習：いろいろなサンプルを実行してみる

- ▶ まずは、examples以下のサンプルをいろいろ実行してみましょう!
- ▶ いったい何をやっているのかを類推しつつ
- ▶ うまく実行できない場合は、質問してください



openFrameworks

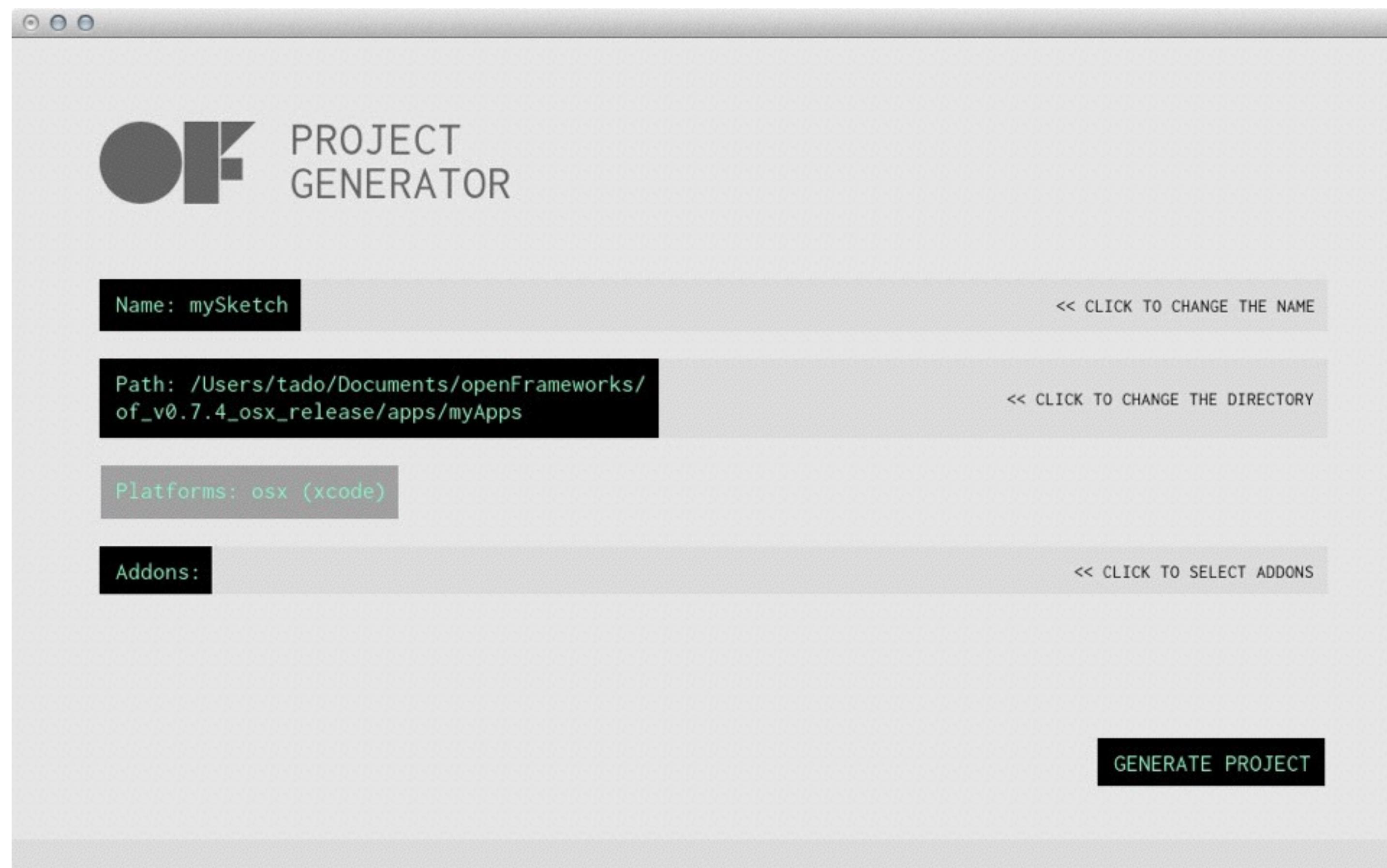
プログラミング、はじめの一歩

新規にプロジェクトを作成する

- ▶ 新規にプロジェクトを作成する方法は2つ
- ▶ 方法1: ProjectGeneratorを利用する
- ▶ 方法2: 空プロジェクト(EmptyProject)をコピーする

新規にプロジェクトを作成する

- ▶ 方法1: ProjectGeneratorを利用する方法 – とても簡単!
- ▶ projectGeneratorフォルダ内のprojectGenerator.appを起動



新規にプロジェクトを作成する

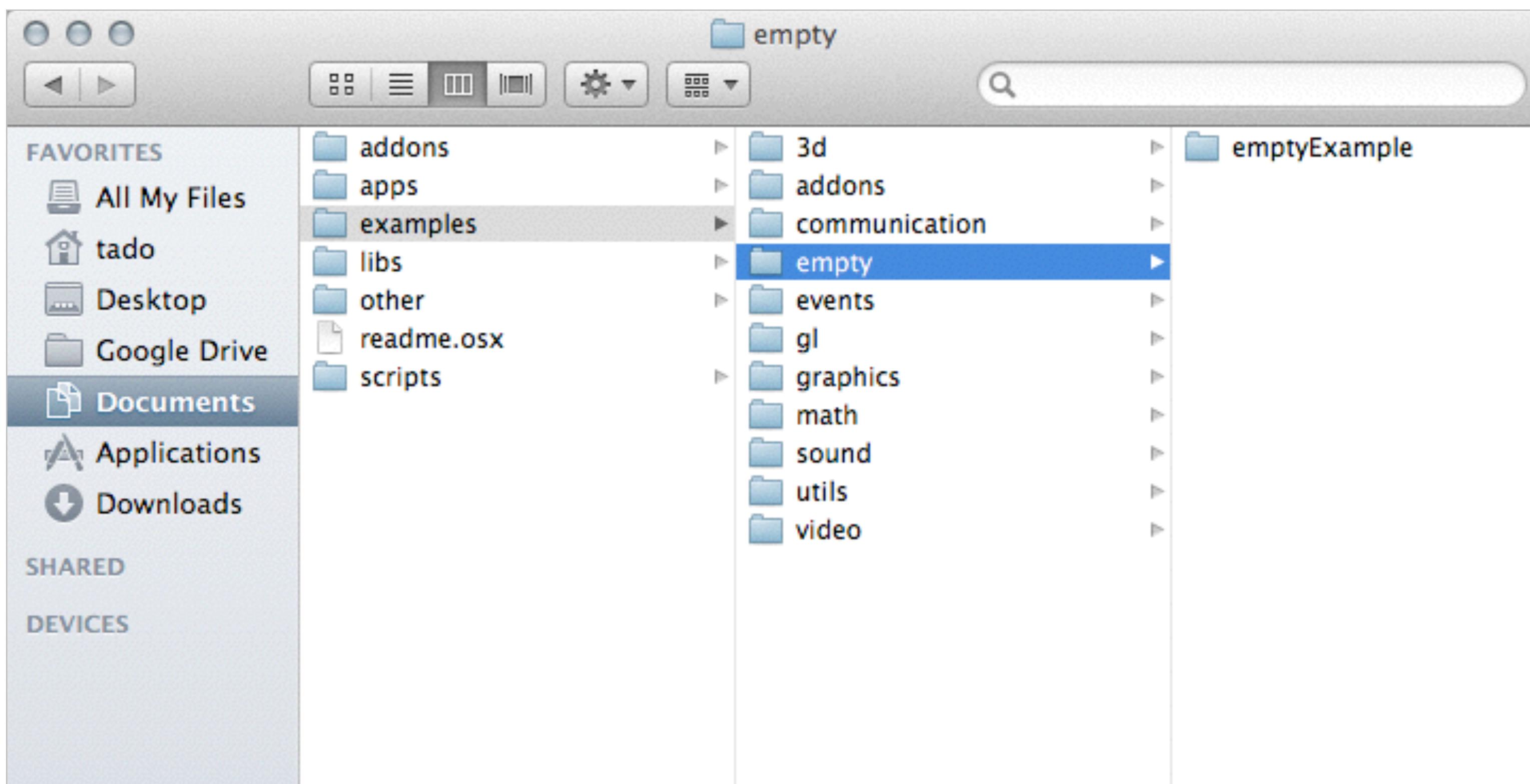
- ▶ 方法1: ProjectGeneratorを利用する方法

- ▶ 新規プロジェクトの作成手順
- ▶ 「CLICK TO CHANGE THE NAME」を選択して名前を設定
- ▶ (もし必要なら)「CLICK TO CHANGE THE NAME」で場所を変更
- ▶ 「GENERATE PROJECT」で生成

- ▶ これで新規プロジェクトに必要なファイル一式が生成される

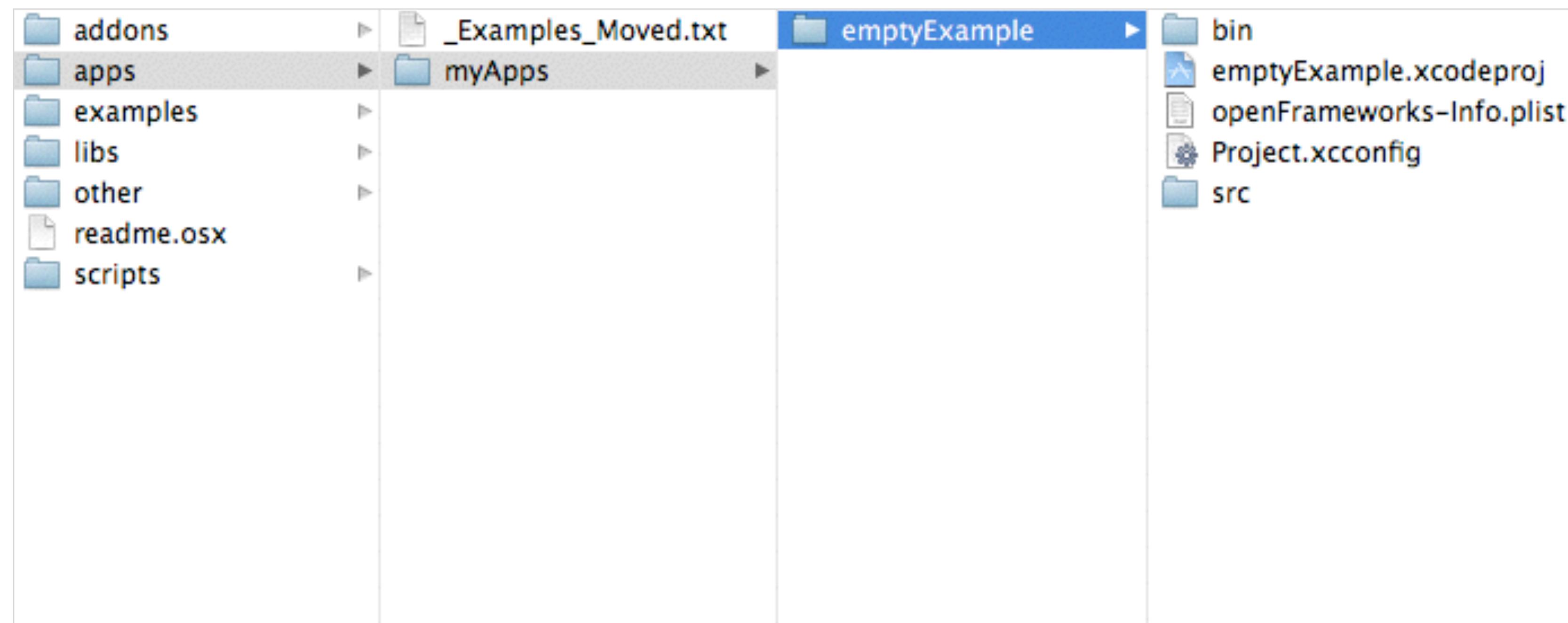
新規にプロジェクトを作成する

- ▶ 方法2: 空プロジェクトをコピーする方法
- ▶ 新規にプロジェクトを作成するには、空プロジェクトをコピー
- ▶ 空プロジェクトは、下記のものをコピーしてつかう
- ▶ examples > empty > emptyExample



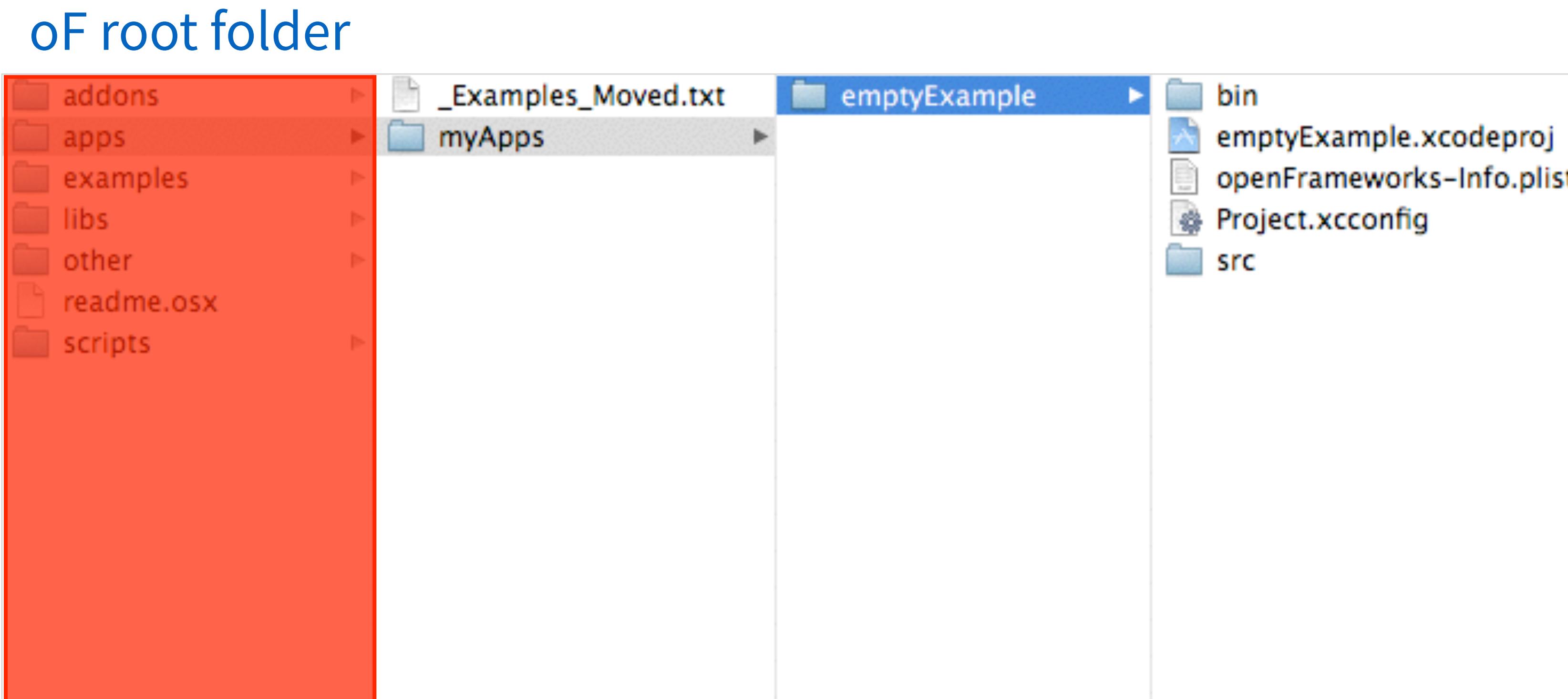
openFrameworks、フォルダの階層構造

- ▶ openFrameworksは、フォルダの階層構造がとても大事!!
- ▶ ここを間違えるとBuildできなくなってしまう



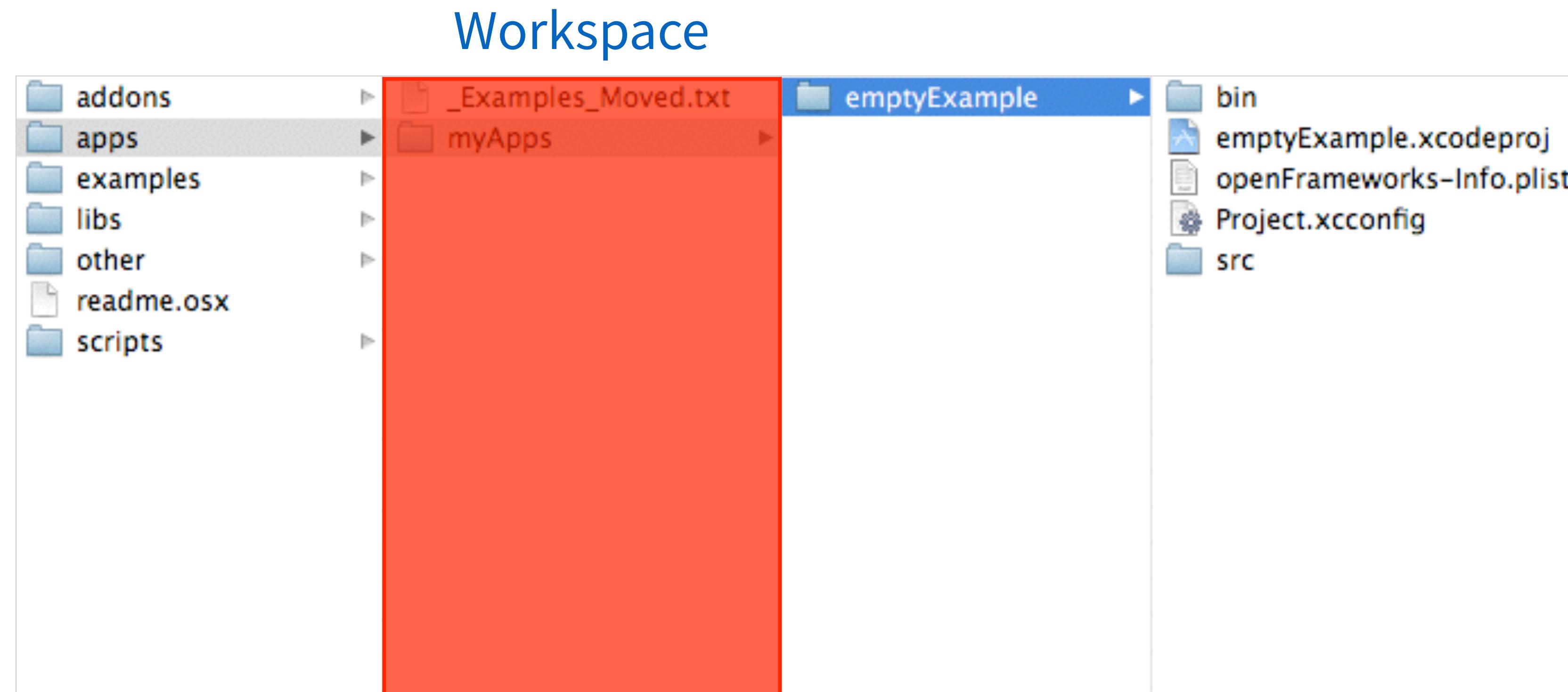
openFrameworks、フォルダの階層構造

- ▶ oFルートフォルダ (oF root folder)
- ▶ openFrameworksの一番先頭(root = 根っこ)の階層



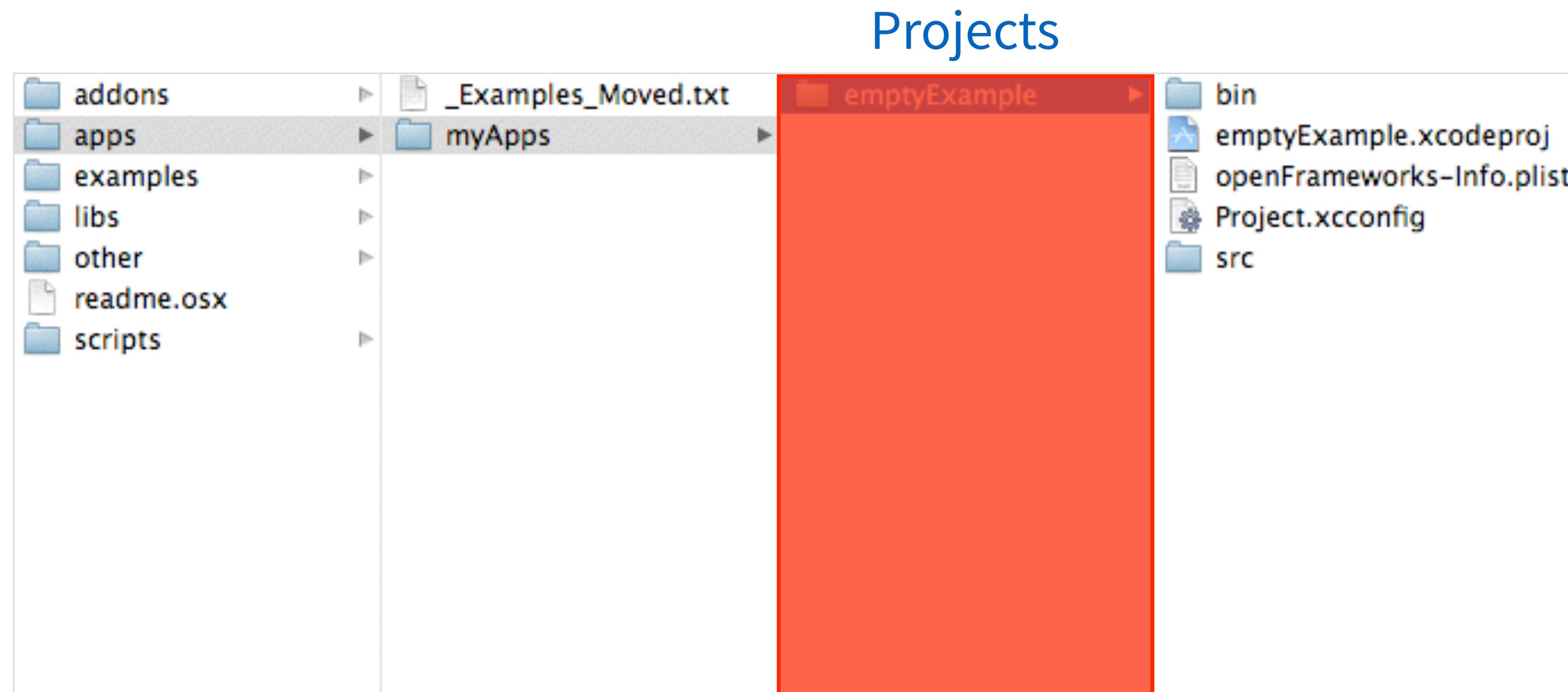
openFrameworks、フォルダの階層構造

- ▶ ワークスペース (Workspace)
- ▶ プロジェクトのまとめを分類して整理 (examples など)



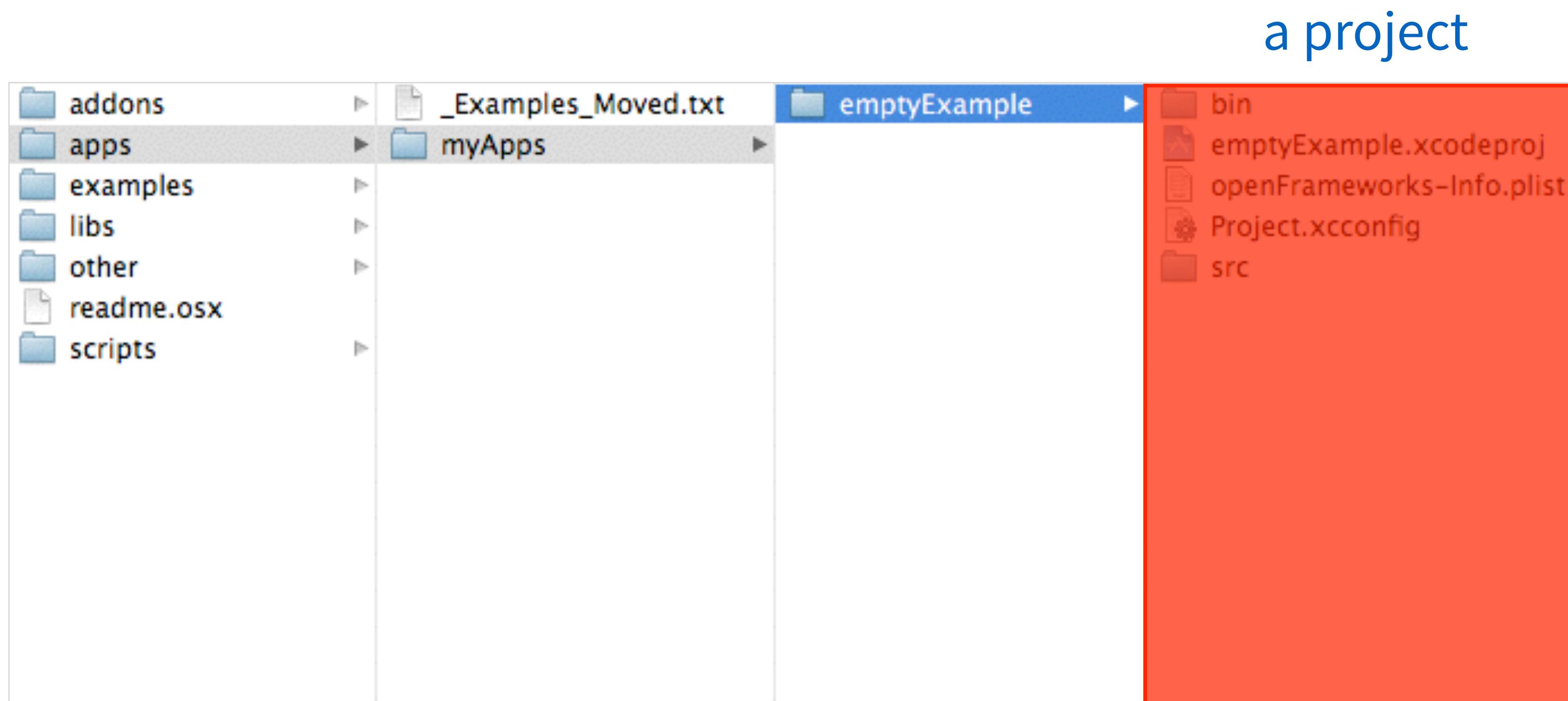
openFrameworks、フォルダの階層構造

- ▶ プロジェクト群 (Projects)
- ▶ この階層に一つ一つのプロジェクトのフォルダが格納



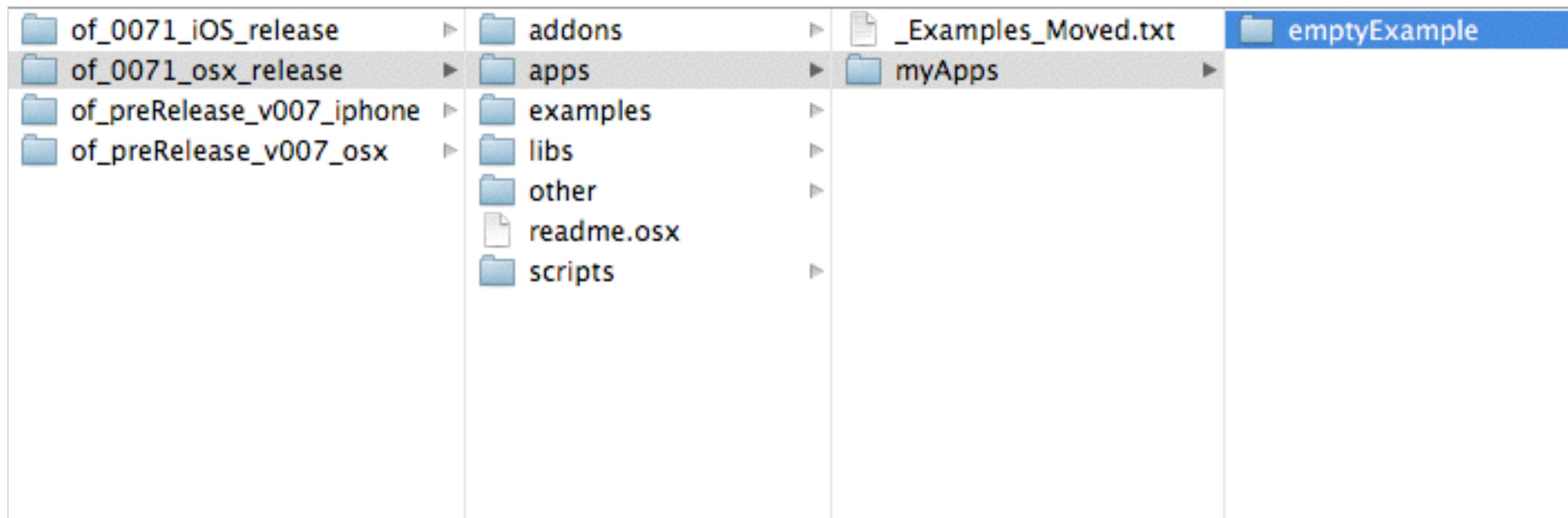
openFrameworks、フォルダの階層構造

- ▶ プロジェクト単体 (a project)
- ▶ Xcodeのプロジェクトファイルを含んだ单一のプロジェクトの中身



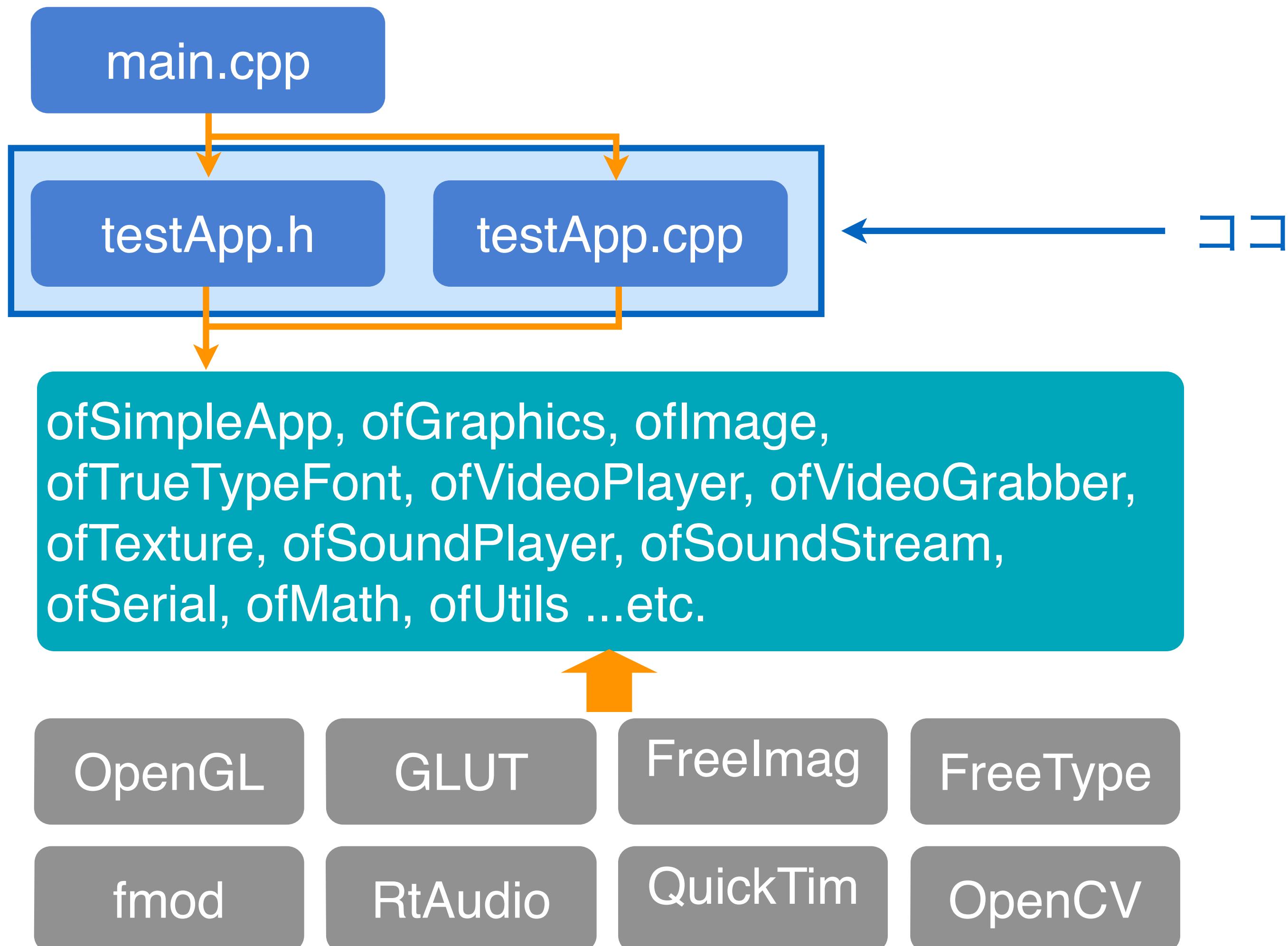
openFrameworks、フォルダの階層構造

- ▶ 今回のワークショップ用に、ワークスペースに一つ専用のフォルダを用意しておきましょう
- ▶ 例えば、myAppsというフォルダを作成した場合



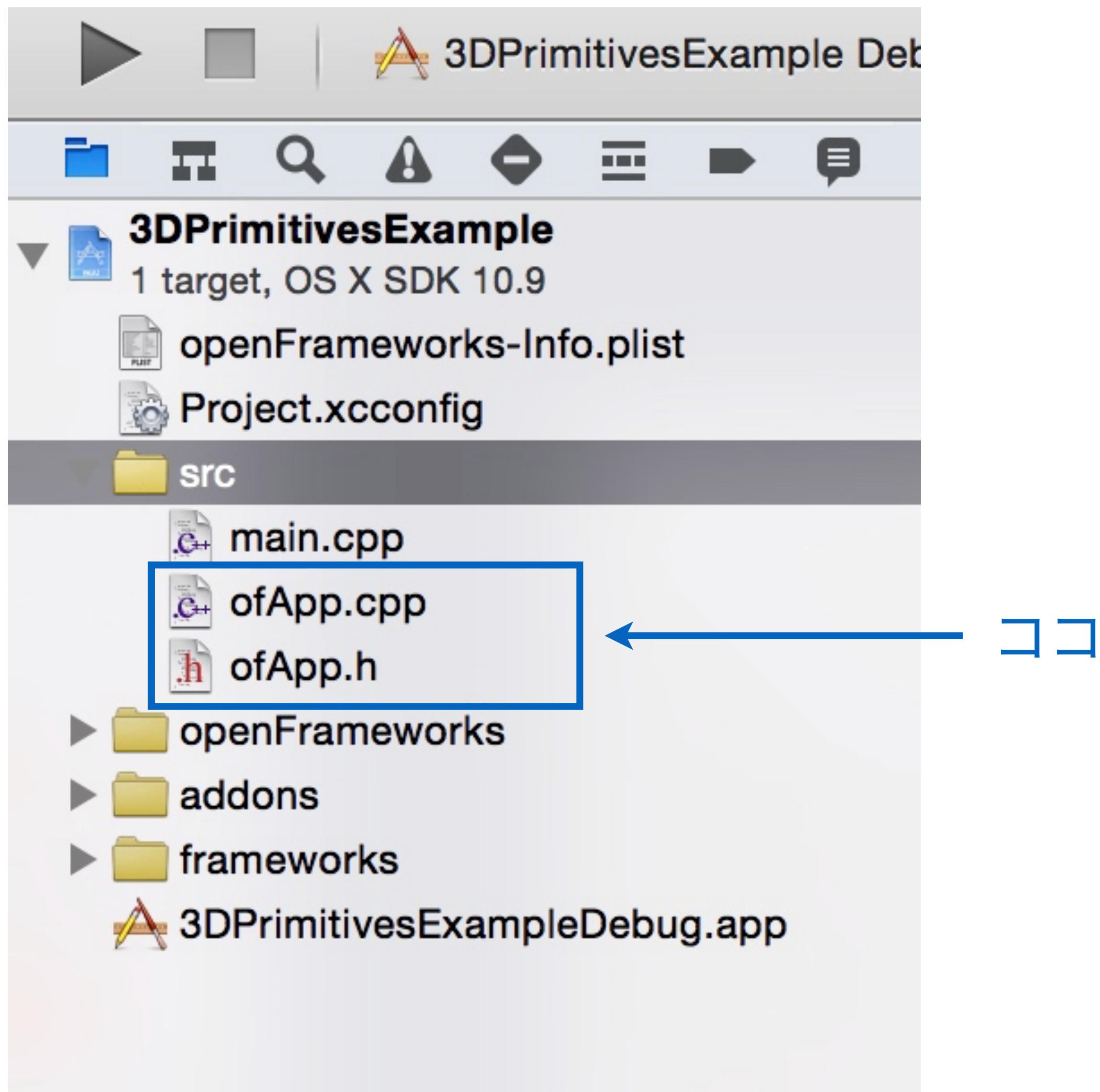
openFrameworksのコード構造

- さしあたって編集するのは、testApp.hとtestApp.cpp



openFrameworksのコード構造

- ▶ testApp.cppとtestApp.h Xcode内の場所



プロジェクトの中身を開いてみよう!

- ▶ testApp.cpp を開いてみる!

The screenshot shows the Xcode IDE interface with the following details:

- Project Navigator:** Shows the project structure under "emptyExample". It includes files like "openFrameworks-Info.plist", "Project.xcconfig", and source files "main.cpp", "ofApp.cpp", and "ofApp.h" located in the "src" folder.
- Editor:** Displays the content of the "ofApp.cpp" file. The code defines an "ofApp" class with methods for setup, update, draw, keypressed, keyreleased, mousemoved, and mousedragged.
- Utilities Navigator:** On the right, the "Identity and Type" tab is selected for "ofApp.cpp".
 - Name:** ofApp.cpp
 - Type:** C++ Source
 - Location:** Relative to Project
 - Full Path:** /Users/tado/Documents/of_v0.8.3_osx_release/apps/myApps/emptyExample/src/ofApp.cpp
- Target Membership:** The checkbox for "emptyExample" is checked.
- Text Settings:** Text Encoding is set to Western (Mac OS Roman).
- Objectives-C Class:** A preview and description of an Objective-C class are shown.
- Objectives-C Protocol:** A preview and description of an Objective-C protocol are shown.
- Objectives-C Test Case Class:** A preview and description of an Objective-C test case class are shown.

2つのファイル

- ▶ ofApp.h - ヘッダファイル
 - ▶ プログラム全体で使用される部品 (変数、関数) を宣言
- ▶ ofApp.cpp - 実装ファイル
 - ▶ 実際に何をするかを記述

2つのファイル

- ▶ ヘッダファイル(.h)と実装ファイル(.cpp)を料理にたとえると…

ヘッダファイル = レシピ



必要な材料の一覧
料理の手順を書き出す

2つのファイル

- ▶ ヘッダファイル(.h)と実装ファイル(.cpp)を料理にたとえると…

ヘッダファイル = レシピ



必要な材料の一覧
料理の手順を書き出す

実装ファイル = 料理



料理完成までの過程を
具体的に全て記述

重要な3つのブロック

- ▶ とりあえず今の段階で重要なのは、下記の3つ処理のブロック (関数, function)
 - ▶ setup - 準備
 - ▶ update - 更新
 - ▶ draw - 描画
- ▶ つまり...
- ▶ 絵を描く準備をしたら継続的に更新しながら描画する

testApp.h では

▶ testApp.h

```
#pragma once

#include "ofMain.h"

class ofApp : public ofBaseApp{
public:
    void setup(); ← 準備
    void update();
    void draw();

    void keyPressed(int key);
    void keyReleased(int key);
    void mouseMoved(int x, int y);
    void mouseDragged(int x, int y, int button);
    void mousePressed(int x, int y, int button);
    void mouseReleased(int x, int y, int button);
    void windowResized(int w, int h);
    void dragEvent(ofDragInfo dragInfo);
    void gotMessage(ofMessage msg);
};
```

testApp.h では

▶ testApp.h

```
#pragma once

#include "ofMain.h"

class ofApp : public ofBaseApp{
public:
    void setup();
    void update(); ← 更新
    void draw();

    void keyPressed(int key);
    void keyReleased(int key);
    void mouseMoved(int x, int y);
    void mouseDragged(int x, int y, int button);
    void mousePressed(int x, int y, int button);
    void mouseReleased(int x, int y, int button);
    void windowResized(int w, int h);
    void dragEvent(ofDragInfo dragInfo);
    void gotMessage(ofMessage msg);
};
```

testApp.h では

▶ testApp.h

```
#pragma once

#include "ofMain.h"

class ofApp : public ofBaseApp{
public:
    void setup();
    void update();
    void draw(); ← 描画

    void keyPressed(int key);
    void keyReleased(int key);
    void mouseMoved(int x, int y);
    void mouseDragged(int x, int y, int button);
    void mousePressed(int x, int y, int button);
    void mouseReleased(int x, int y, int button);
    void windowResized(int w, int h);
    void dragEvent(ofDragInfo dragInfo);
    void gotMessage(ofMessage msg);
};
```

testApp.cpp では

- ▶ testApp.h

```
#include "ofApp.h"

//-----
void ofApp::setup(){

}

//-----
void ofApp::update(){

}

//-----
void ofApp::draw(){

}
```

← 準備

testApp.cpp では

- ▶ testApp.h

```
#include "ofApp.h"

//-----
void ofApp::setup(){

}

//-----
void ofApp::update(){

}

//-----
void ofApp::draw(){

}
```

← 更新

testApp.cpp では

- ▶ testApp.h

```
#include "ofApp.h"

//-----
void ofApp::setup(){

}

//-----
void ofApp::update(){

}

//-----
void ofApp::draw(){

}
```

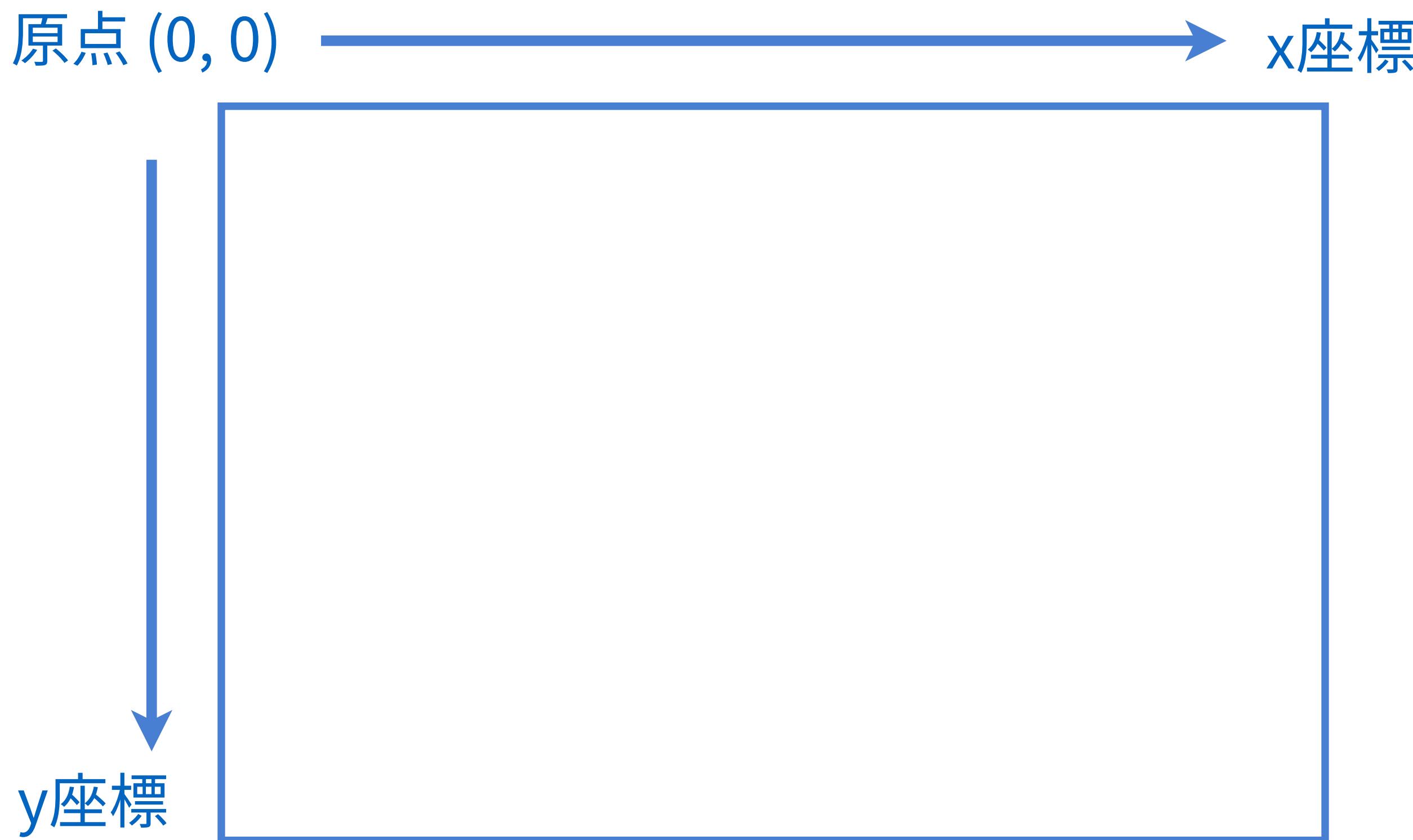
← 描画

図形を描いてみる!

- ▶ まず円を描いてみましょう
- ▶ 何を指定したら円を描けるのか、を考える

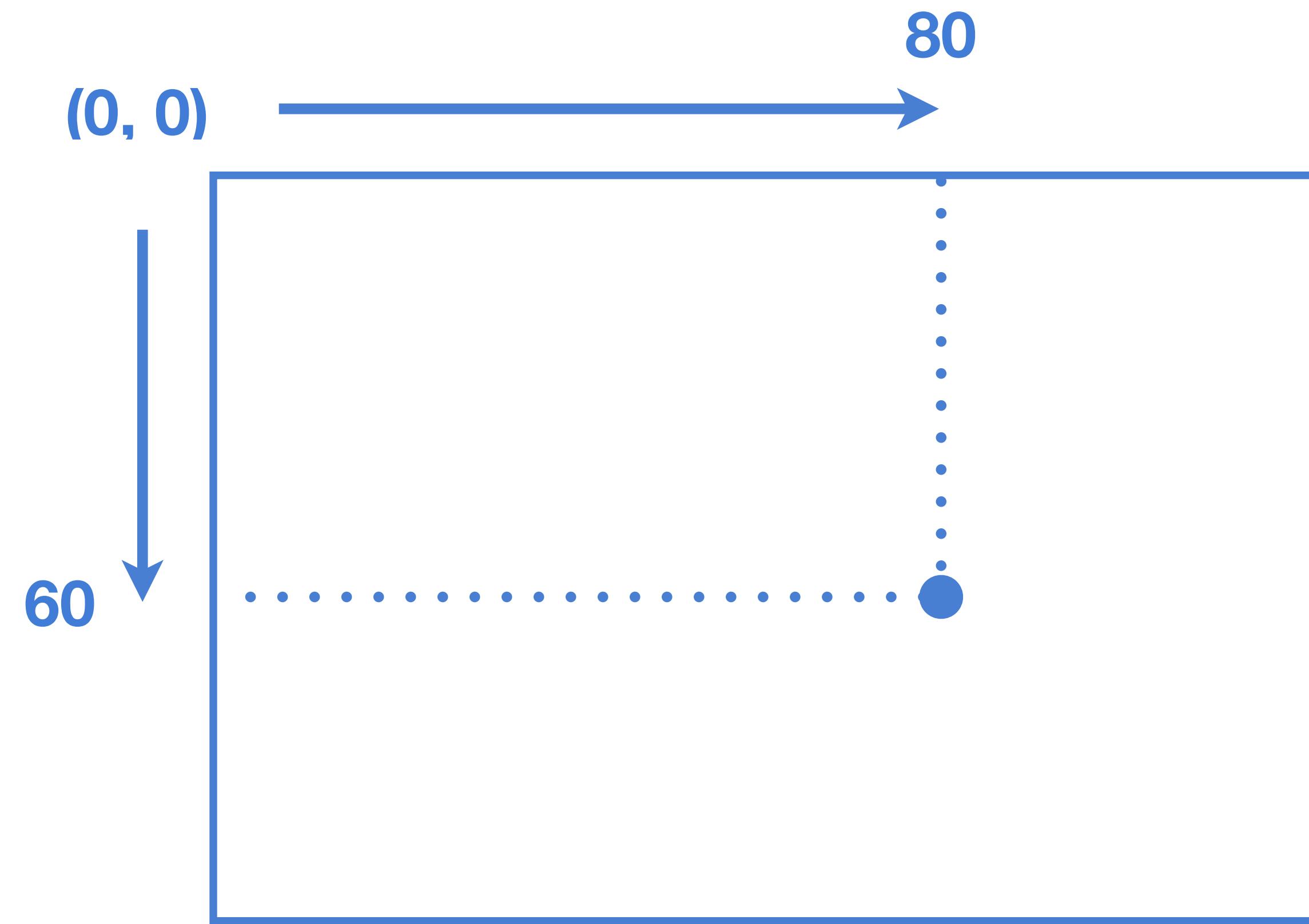
図形を描いてみる!

- ▶ 座標系：画面の中の場所(点)を指定するしくみ
- ▶ 横(x座標)と縦(y座標)で考える
- ▶ openFrameworksの場合、原点(0, 0)は左上



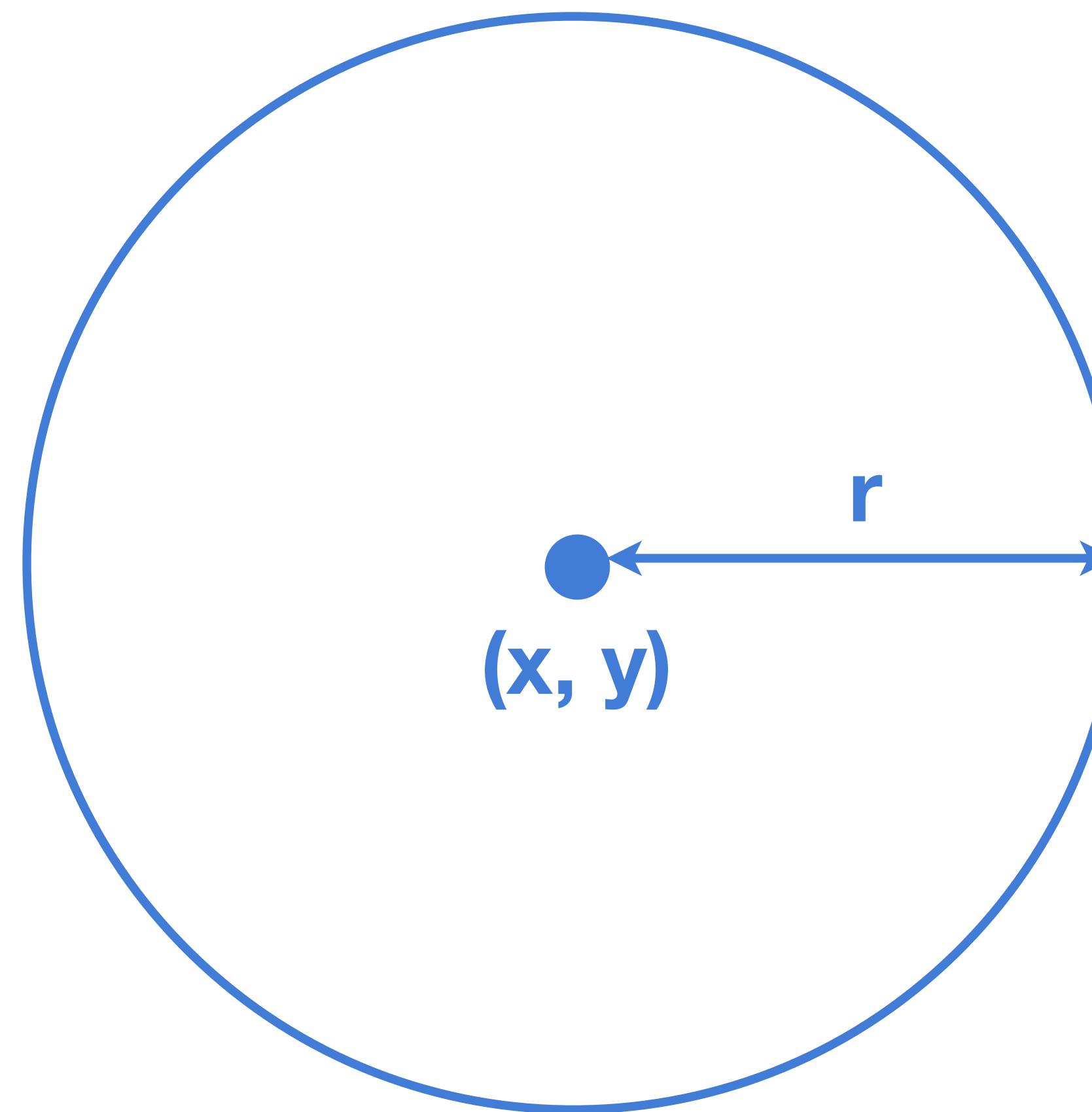
図形を描いてみる!

- ▶ 例えば、 $(80, 60)$ の点($x = 80, y = 60$)だったら
- ▶ 左上の点から、80pixel右、上から60pixel下にいったところ



図形を描いてみる!

- ▶ 中心の位置(座標 = x, y)と半径(r)の長さがわかれば円は描くことができる!



図形を描いてみる!

- ▶ openFrameworksでは、下記のように指定する

ofCircle (中心のx座標, 中心のy座標, 半径の長さ);

- ▶ 例：

ofCircle (100, 200, 50);

- ▶ 座標(100, 200)を中心、半径50の円を描く

やってみよう!!

```
#include "ofApp.h"

//-----
void ofApp::setup(){

}

//-----
void ofApp::update(){

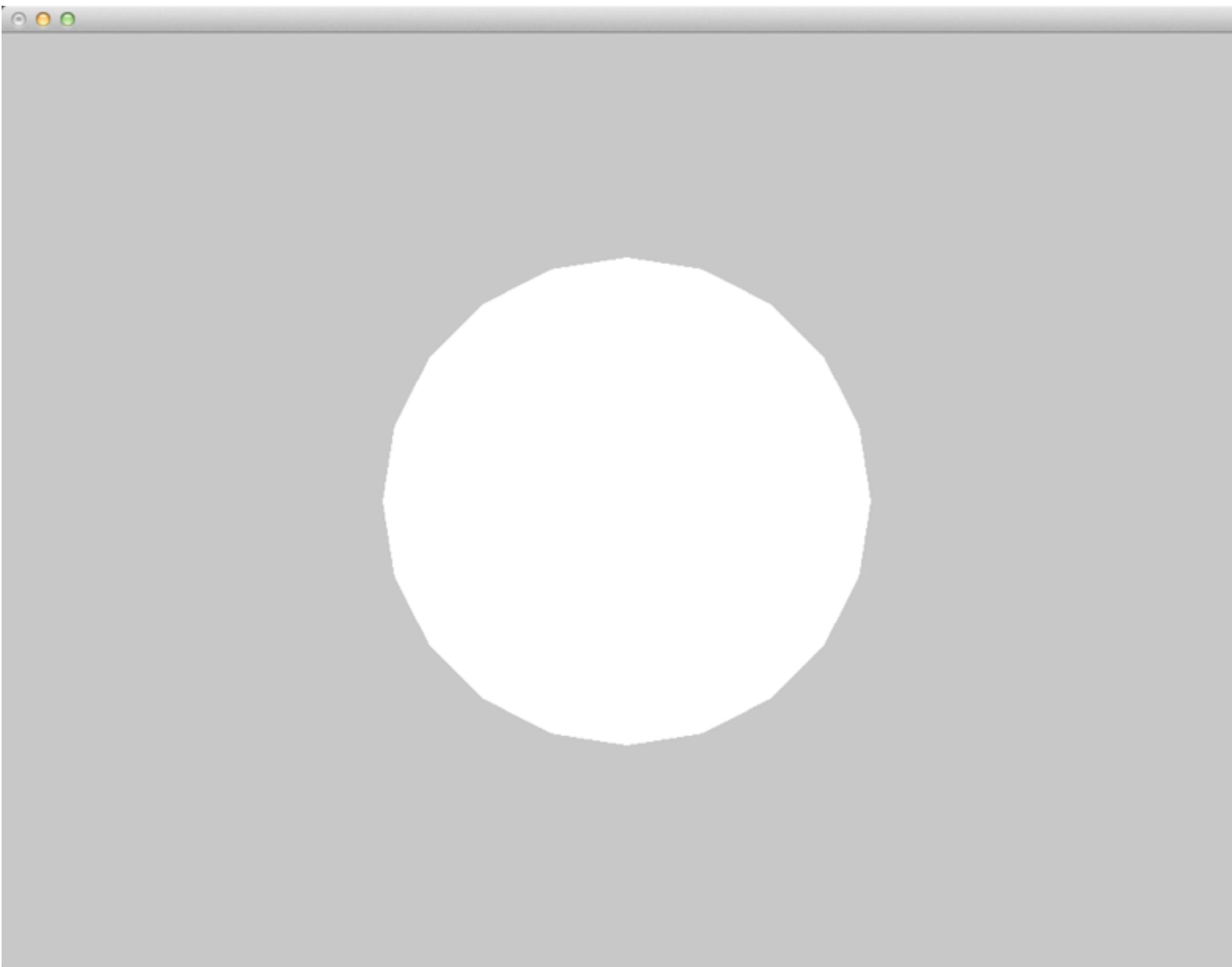
}

//-----
void ofApp::draw(){
    ofCircle(512, 384, 200);
}

. . .
```

やってみよう!!

▶ 円が描けた!



参考：oFの命令を調べる

- ▶ 円以外の形を描きたくなったとき、どうやって調べる？
- ▶ リファレンスを参考にすると良い
- ▶ <http://www.openframeworks.cc/documentation>

The screenshot shows the official OpenFrameworks documentation website. At the top, there's a navigation bar with links to 'about / setup / download / addons / **docs** / gallery / community / forum / wiki / rss feeds'. Below the navigation is a search bar with a 'Search' button and a 'Register / Login' link. The main content area is titled 'documentation'. It contains a message about the start documentation for the main OF api, mentioning additional libraries available on the addons page. It also includes a 'Show advanced?' toggle with 'yes / no' options. A note states that the document refers to version 0.06. The page is organized into four main sections: 'application', 'graphics', 'video', and 'serial'. The 'application' section lists methods like 'ofBackground()', 'ofSetColor()', etc. The 'graphics' section lists 'ofRect()', 'ofTriangle()', etc. The 'video' section lists 'listDevices()', 'isFrameNew()', etc. The 'serial' section lists 'enumerateDevices()', 'close()', etc.

This page the start documentation for the main OF api. There are also additional libraries that you can find on the addons page. This api is a work in progress and any corrections, additions or comments are very welcome!

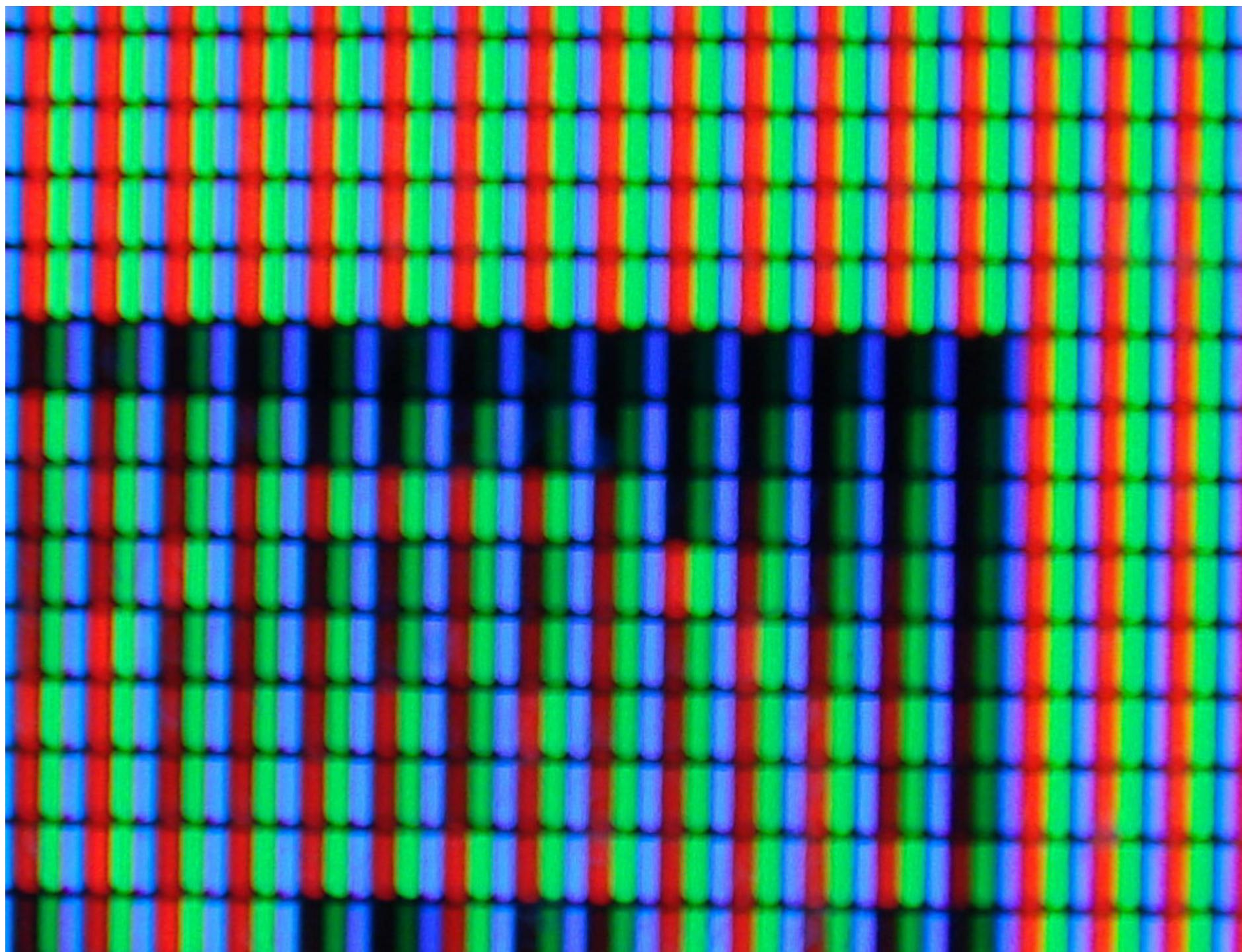
Show advanced?
yes / [no](#)

This document refers to version 0.06

application	graphics	video	serial
class ofBaseApp setup() update() draw() exit() windowResized(...) keyPressed(...) keyReleased(...) mouseMoved(...) mouseDragged(...)	ofBackground(...) ofSetColor() ofSetBackgroundAuto(...) ofbClearBg() ofSetCircleResolution(...) ofSetRectMode(...) ofGetRectMode() ofRect(...) ofTriangle(...) ofCircle(...) ofEllipse(...)	class ofVideoGrabber listDevices() isFrameNew() grabFrame() close() initGrabber(...) videoSettings() getPixels() getTextureReference() setVerbose(...)	class ofSerial enumerateDevices() close() setup() setup(...) setup(...) readBytes(...) writeBytes(...) writeByte(...) readByte()

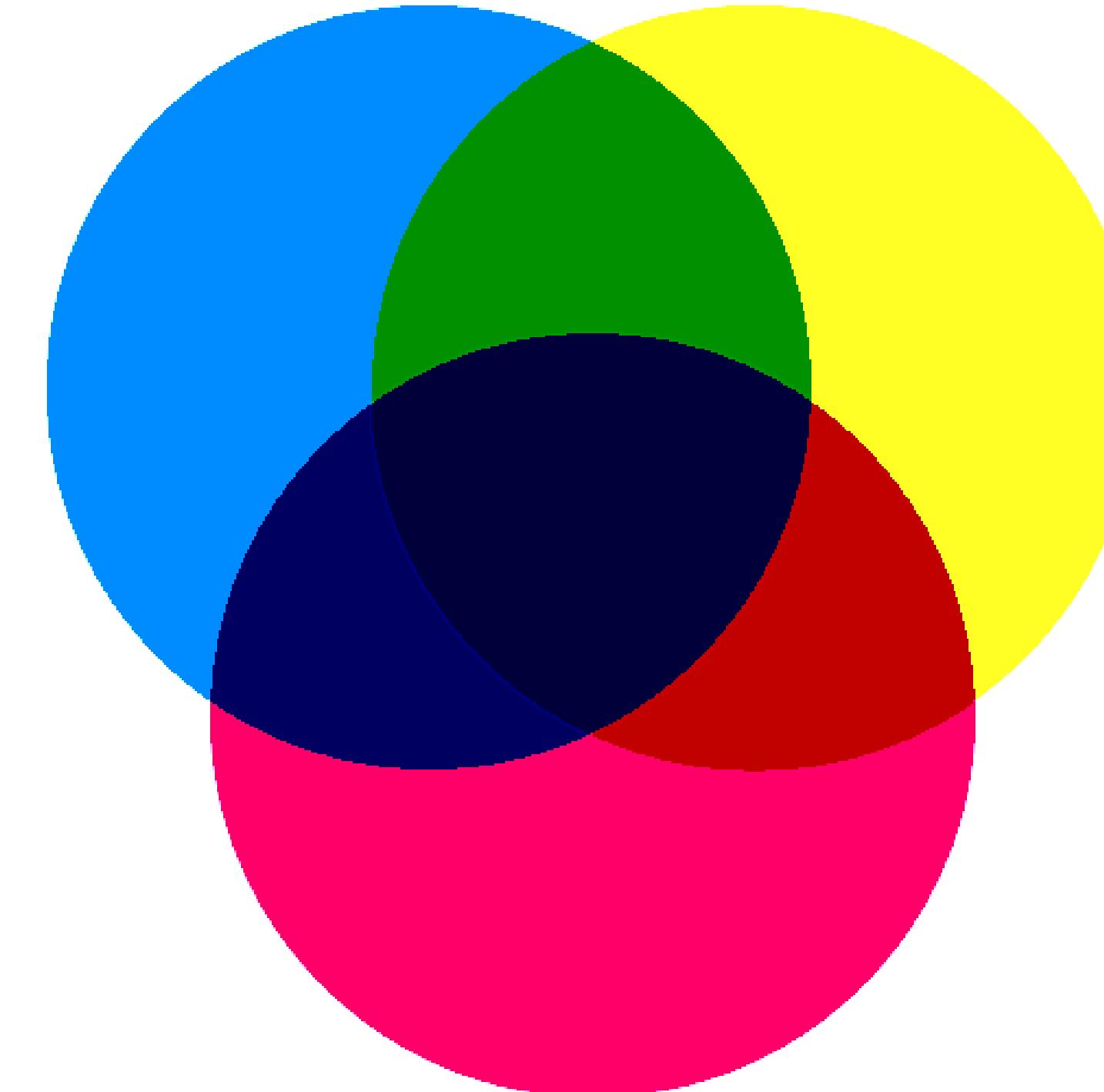
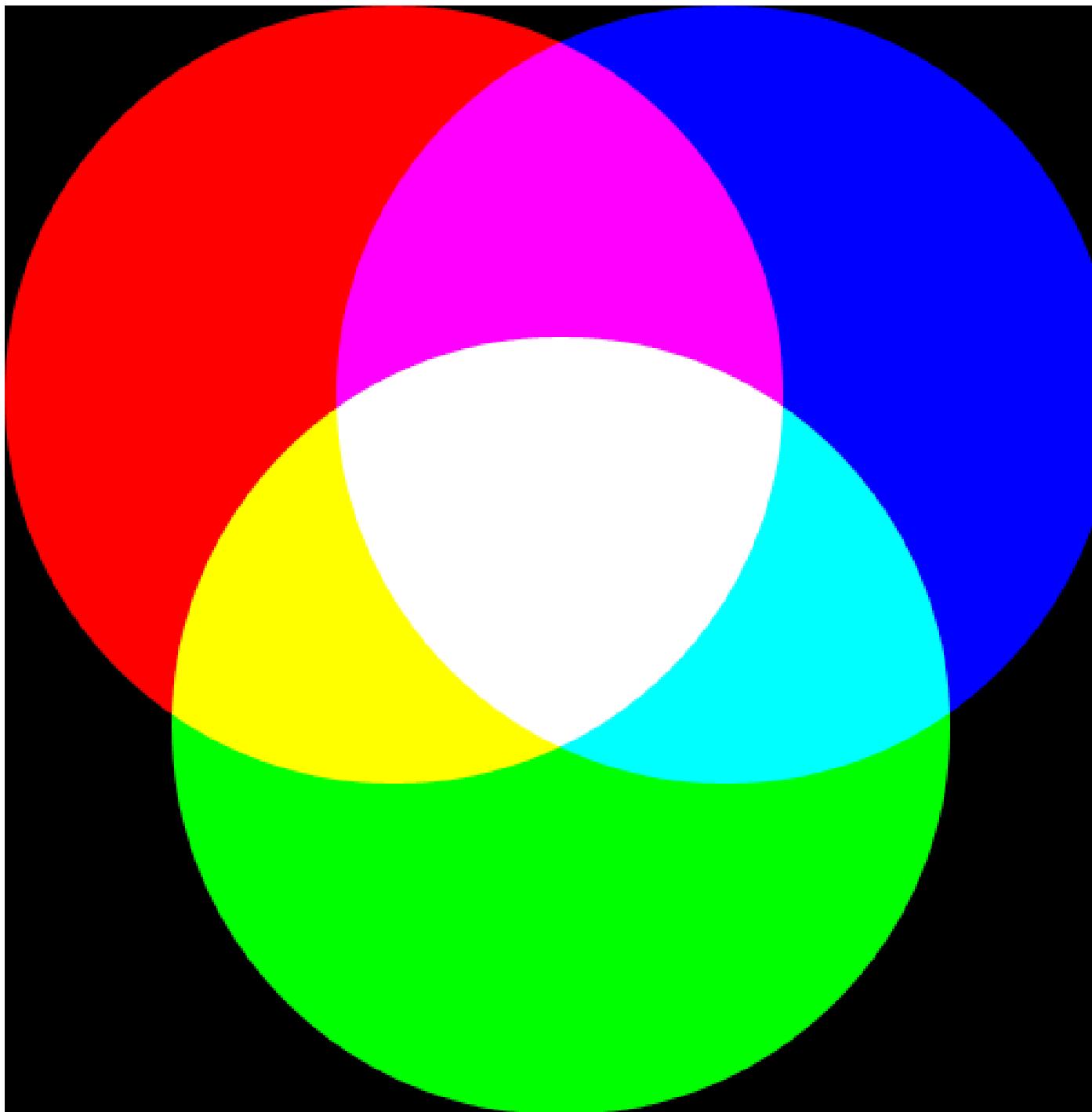
色を塗ってみる

- ▶ コンピュータの画面はどうなっているのか?
- ▶ コンピュータの画面を拡大していくと...
- ▶ 縦横に並んだ点の集合 → ピクセル (Pixel)
- ▶ 一つのピクセルは赤、緑、青の三原色から成り立っている



色を塗ってみる

- ▶ 色を指定するには?
- ▶ R(赤) G(緑) B(青)の三原色で指定する
- ▶ 加法混色 (光の三原色であることに注意) ←→ 色料の三原色



色を指定するには?

- ▶ openFrameworks で色を指定するには?
- ▶ ofSetColor を使用する

`ofSetColor(Red, Green, Blue, Alpha);`

- ▶ それぞれの色の範囲は 0 ~ 255
 - ▶ Alphaは透明度をあらわす
 - ▶ 色を指定した以降の図形に適用される
-
- ▶ 例 :

`ofSetColor(0, 127, 255, 127);`

色を指定するには?

```
#include "ofApp.h"

//-----
void ofApp::setup(){

}

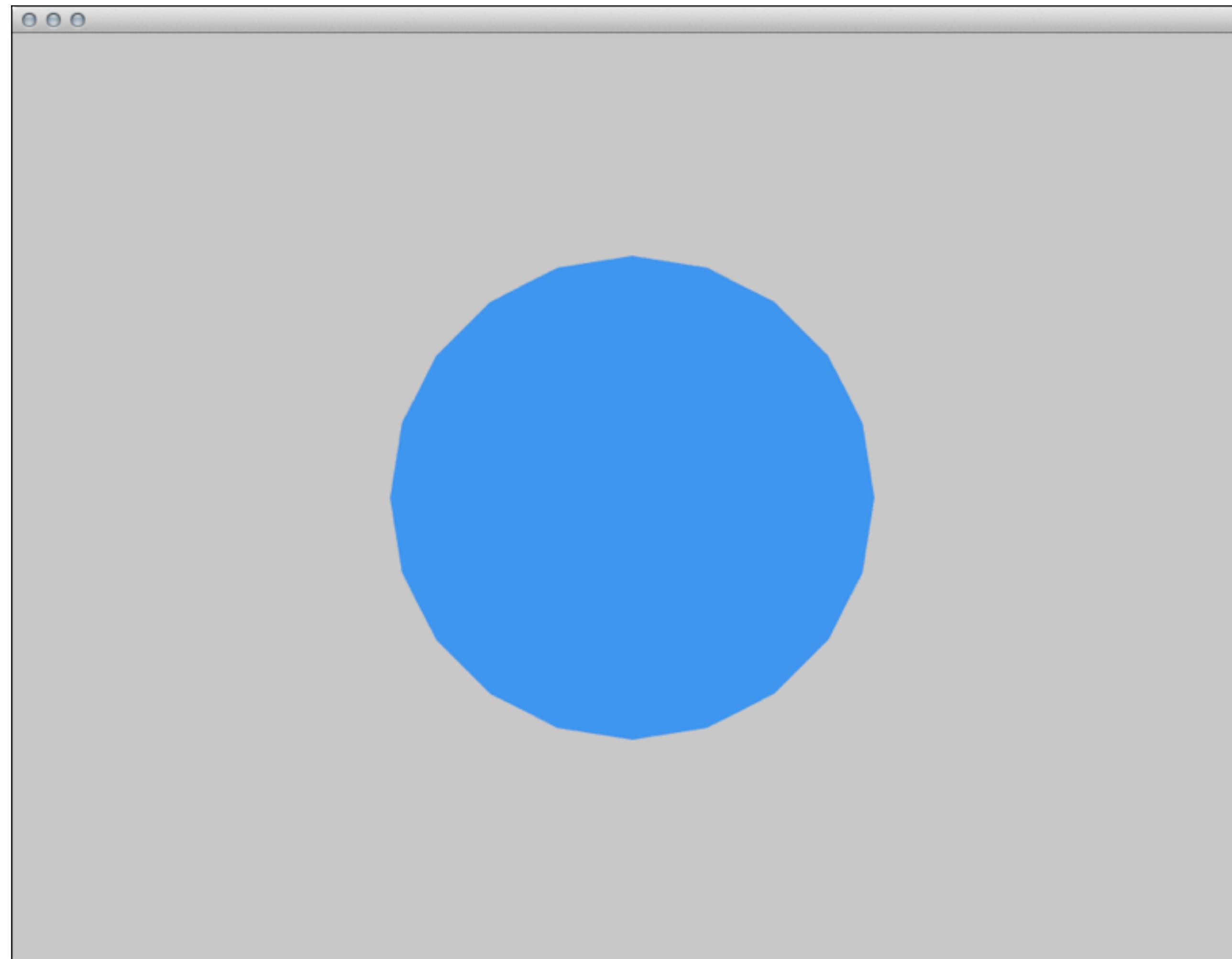
//-----
void ofApp::update(){

}

//-----
void ofApp::draw(){
    ofSetColor(0, 127, 255, 200);
    ofCircle(512, 384, 200);
}
```

色を指定するには?

- ▶ 色がついた!



背景色や描画方法の初期設定

- ▶ `setup()` に様々な初期設定を行う
- ▶ 透明度を有効に - `ofEnableAlphaBlending();`
- ▶ 円を描画する精度 - `ofSetCircleResolution(分割数);`
- ▶ 背景色 - `ofBackground(R, G, B);`

背景色や描画方法の初期設定

```
#include "ofApp.h"

//-----
void ofApp::setup(){
    ofEnableAlphaBlending();
    ofSetCircleResolution(64);
    ofBackground(0, 0, 0);
}

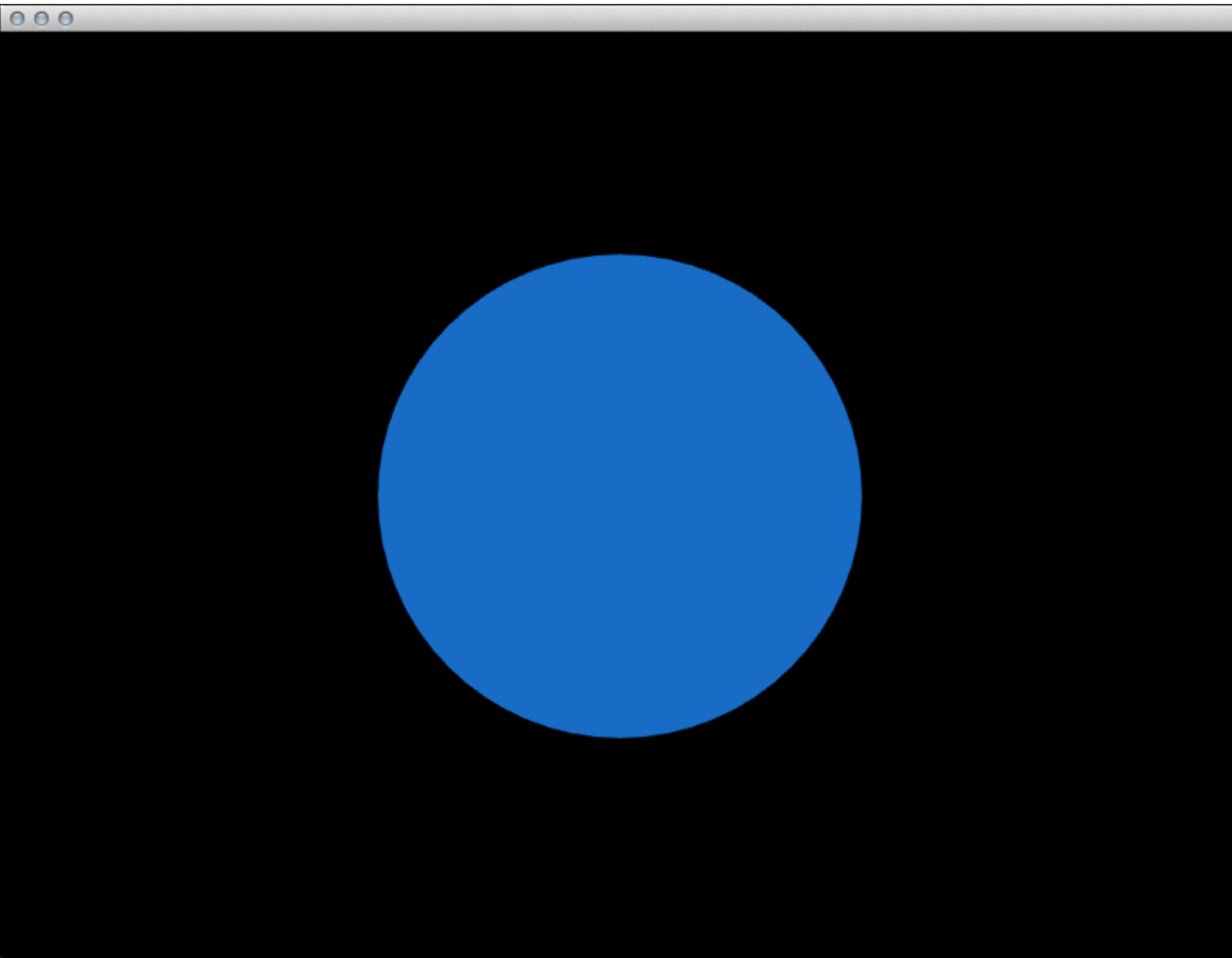
//-----
void ofApp::update(){

}

//-----
void ofApp::draw(){
    ofSetColor(0, 127, 255, 200);
    ofCircle(512, 384, 200);
}
```

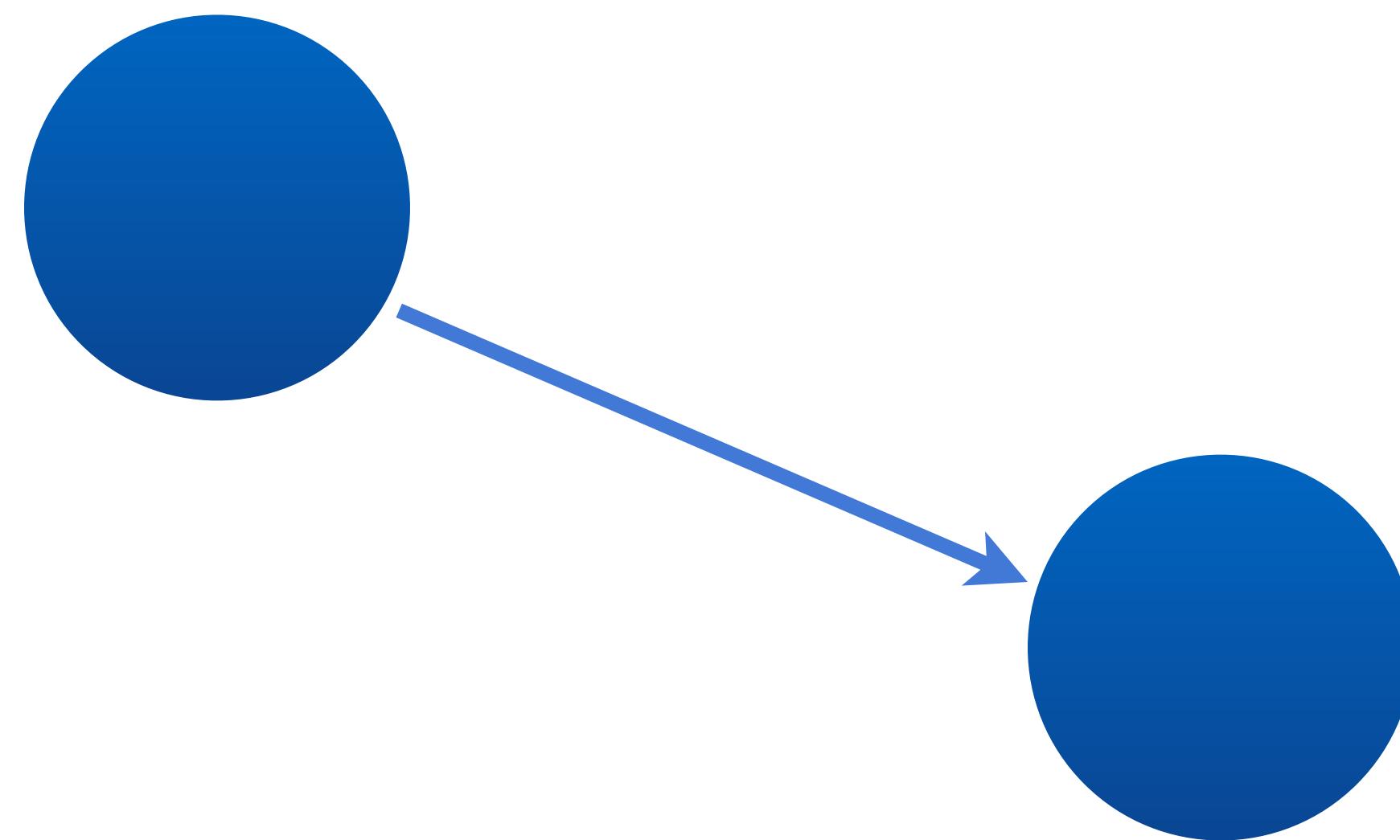
背景色や描画方法の初期設定

- ▶ 背景色 + 円がきれいに!



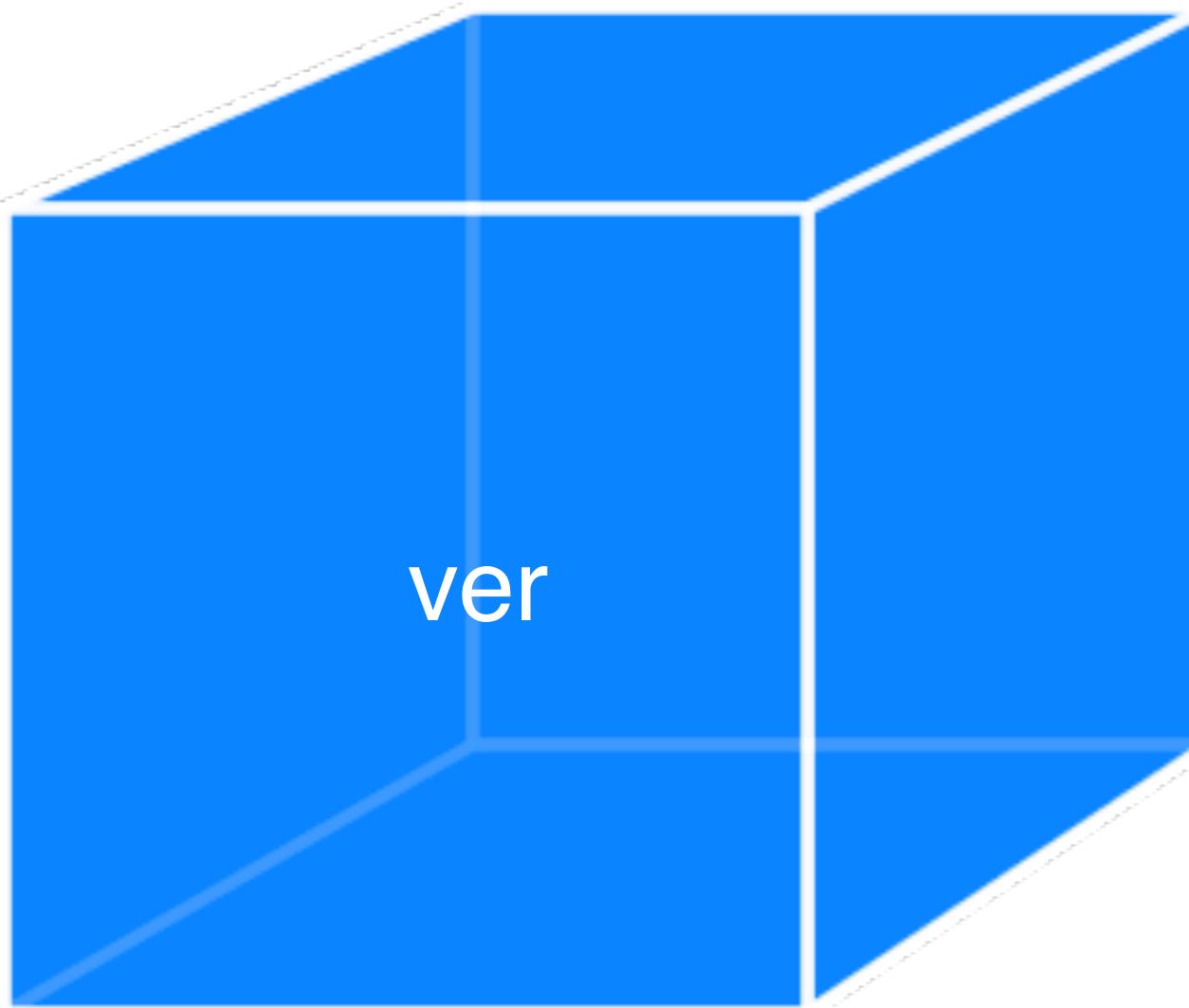
図形を動かしてみよう!

- ▶ いよいよ図形を動かしてみましょう!
- ▶ 円を直線運動させてみる!
- ▶ 円を動かすには、現在の位置からの変化を記述していく
- ▶ そのためには、前回の位置を記憶しておく必要がある!!



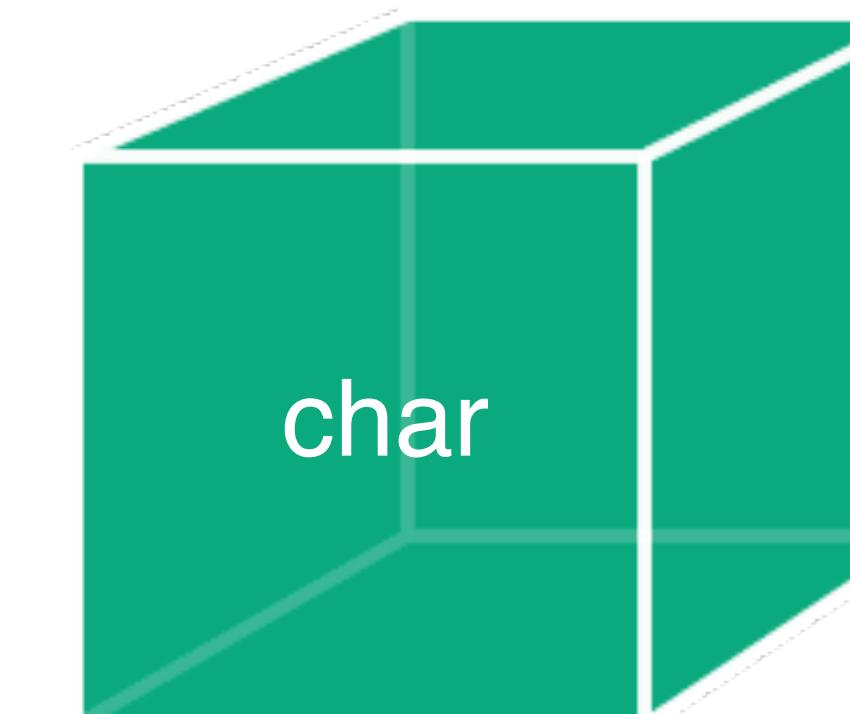
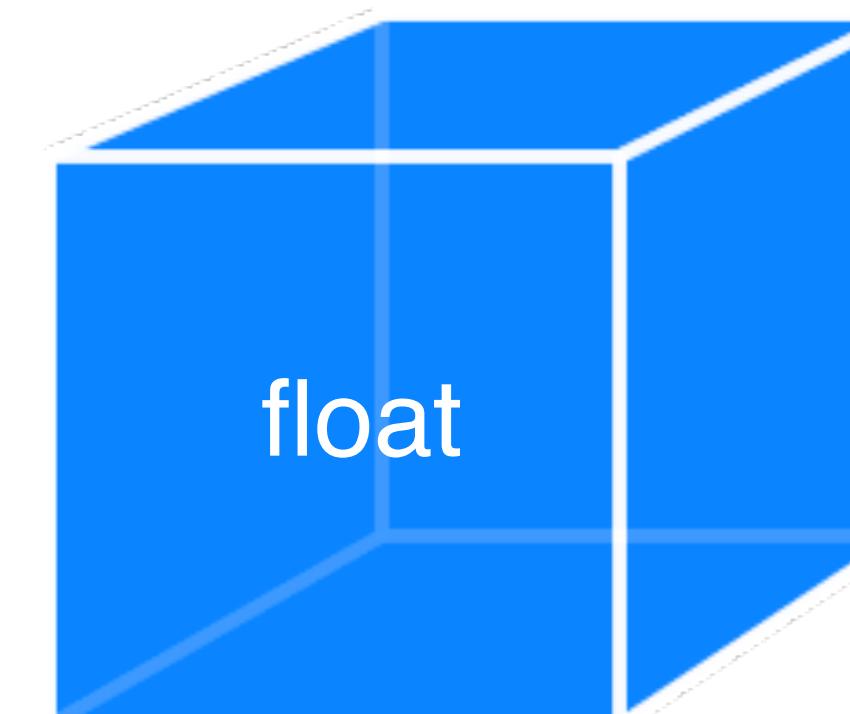
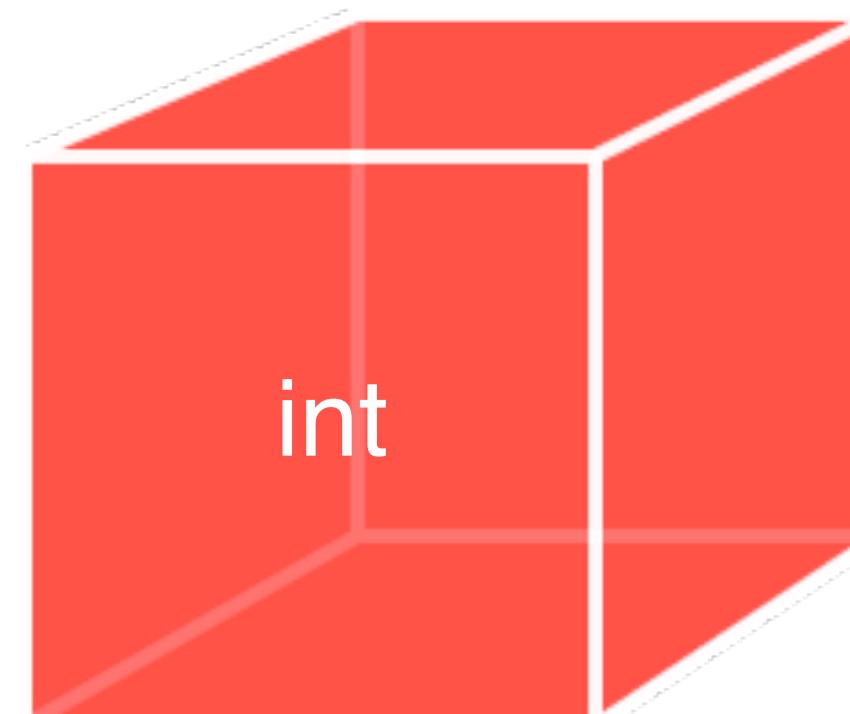
変数

- ▶ 変数とは?
- ▶ 一時的に値（文字、文字列、数字など）を記憶しておく場所
- ▶ データを入れておく「箱」のようなもの



変数

- ▶ データ型 - 値の種類
- ▶ よく用いられるデータ型
 - ▶ int : 整数 (-1, 0, 1, 2, 3....)
 - ▶ float : 少数 (-0.01, 3.14, 21.314)
 - ▶ bool : ブール値、真か偽か、(true, false)
 - ▶ char : 1文字分のデータ(a, b, c, d...)
 - ▶ string : 文字列 “Hello World!”



変数

- ▶ 宣言：使用する変数の名前の箱を準備する

```
int hoo;
```

- ▶ 代入：変数の箱に値を入れる

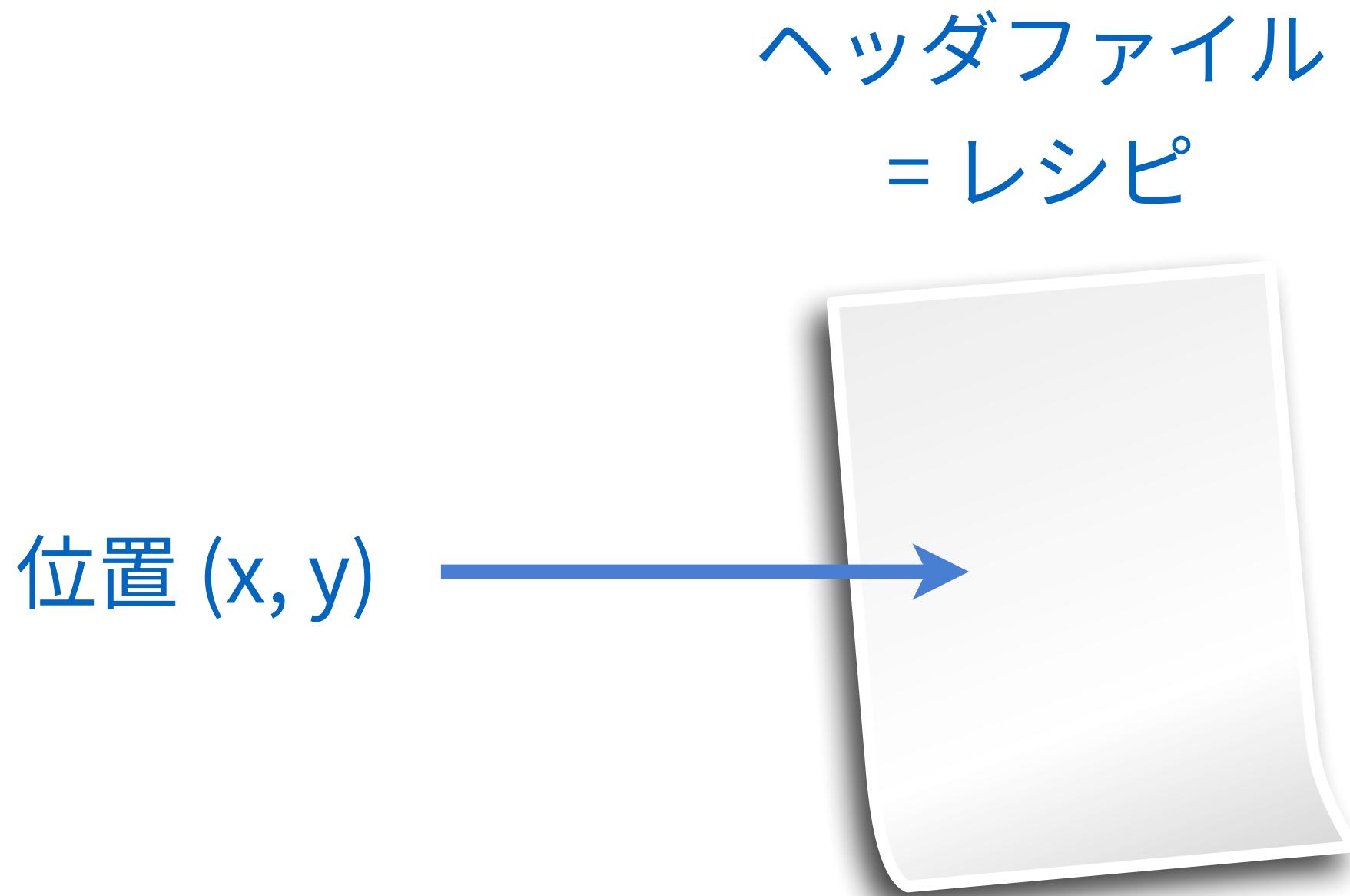
```
hoo = 0;
```

- ▶ 演算：変数の値を計算する

```
hoo = hoo + 1;
```

図形を動かしてみよう!

- ▶ 位置記録する箱
- ▶ X座標とY座標の2つ : positionX , positionY
- ▶ testApp全体で使用する変数は、ヘッダファイル(=レシピ!)に記述する



図形を動かしてみよう!

▶ testApp.h を編集

```
#pragma once
#include "ofMain.h"

class ofApp : public ofBaseApp{
public:
    void setup();
    void update();
    void draw();

    void keyPressed(int key);

    【中略】

    void gotMessage(ofMessage msg);

    float positionX;
    float positionY; ← 追加
};
```

図形を動かしてみよう!

- ▶ 実装ファイル testApp.cpp にも変更を加える
- ▶ setup():
- ▶ フレームレート(1秒間に更新する回数)を設定
- ▶ update():
- ▶ 前のフレームからの(x, y)の変化量を指定
- ▶ draw():
- ▶ 更新した位置で円を描く

図形を動かしてみよう!

- ▶ testApp.cpp を編集

```
void ofApp::setup(){
    ofEnableAlphaBlending();
    ofSetCircleResolution(64);
    ofBackground(0, 0, 0);
    ofSetFrameRate(60);
    positionX = 0;
    positionY = 0;
}
```

]← 追加

図形を動かしてみよう!

- ▶ testApp.cpp を編集

```
//-----
void ofApp::update(){
    positionX = positionX + 3;
    positionY = positionY + 2; ] ← 追加
}

//-----
void ofApp::draw(){
    ofSetColor(0, 127, 255, 200);
    ofCircle(positionX, positionY, 20); ← 変更
}
```

図形を動かしてみよう!

- ▶ 円が、左上から右下にむかってアニメーションするはず!!

