

Coding and debugging with Xcode

2016/03/16

東京大学大学院 情報学環学際情報学府 学際情報学専攻

暦本研究室 修士課程1年

松田 晓



<https://github.com/0x0c/XcodeStudy>

Xcodeって？

- **Apple製のIDE**
 - 主にiOS/OS Xのアプリ開発に使える
- **実はいろいろな言語で使える**
 - Obj-C, C++, C, Java

Xcodeのいいところ

- ・ ネイティブアプリである
 - ・ さくさく動く
 - ・ UIがOS標準
- ・ 基本的にいい感じに補完が効く
- ・ デバッガ(lldb)をGUIで操作できる
- ・ Cmakeから簡単に乗り換えられる

- ・次のコマンドで変換可能
 - ・ cmake -G Xcode .
 - ・例)
 - ・ cd /Library/Caches/Homebrew/pcl—git/doc/tutorials/content/sources/matrix_transform
 - ・ cmake -G Xcode .
 - ・ CMakeCache.txtを削除する必要あり

インストール方法

- App Storeからインストール
 - らくちん
- dmgからインストール
 - beta版がインストール可能
 - 最新のObj-C, Swiftが利用可能
 - Apple Developer Accountが必要



インストール方法

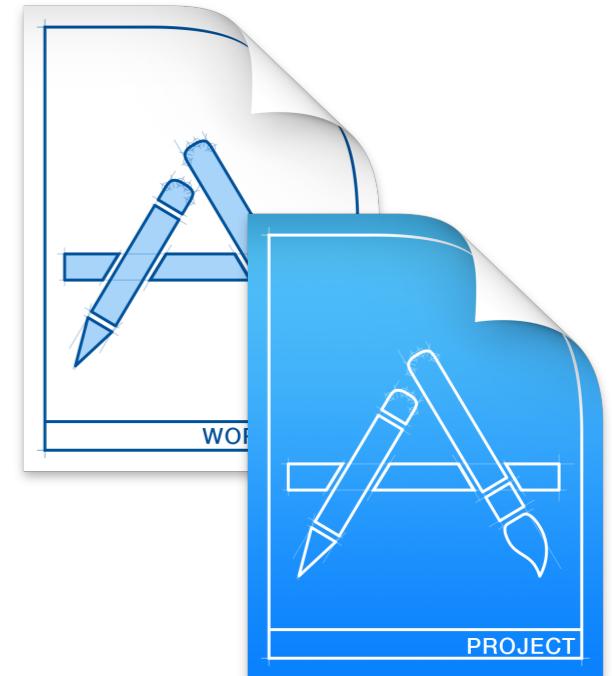
- App Storeからインストール
 - らくちん
- dmgからインストール
 - beta版
 - 最新のObj-C, Swiftが利用可能
- Apple Developer Accountが必要



入門編

Xcodeの基本 – プロジェクト

- **単一のproject or
複数のproject and workspace**

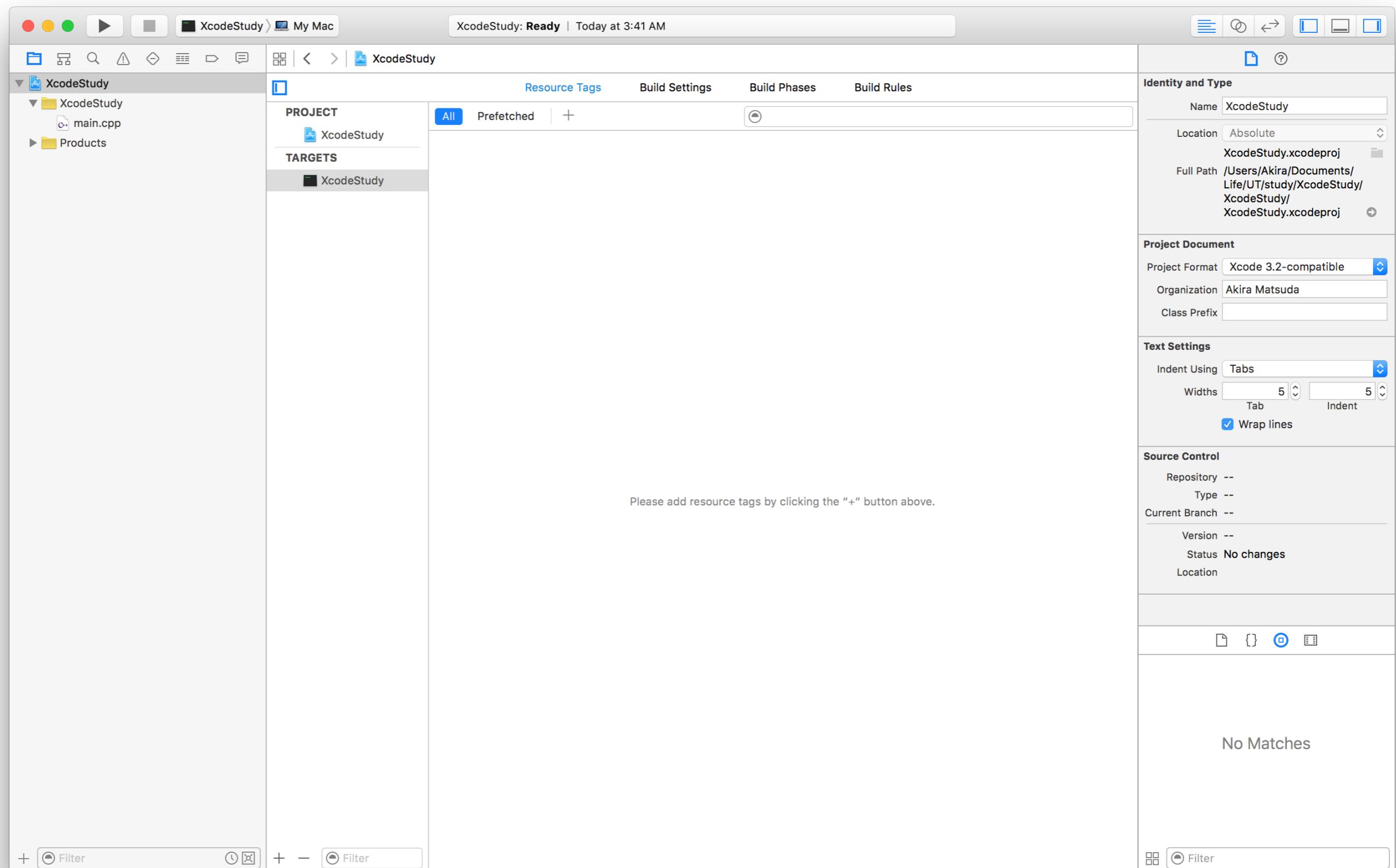


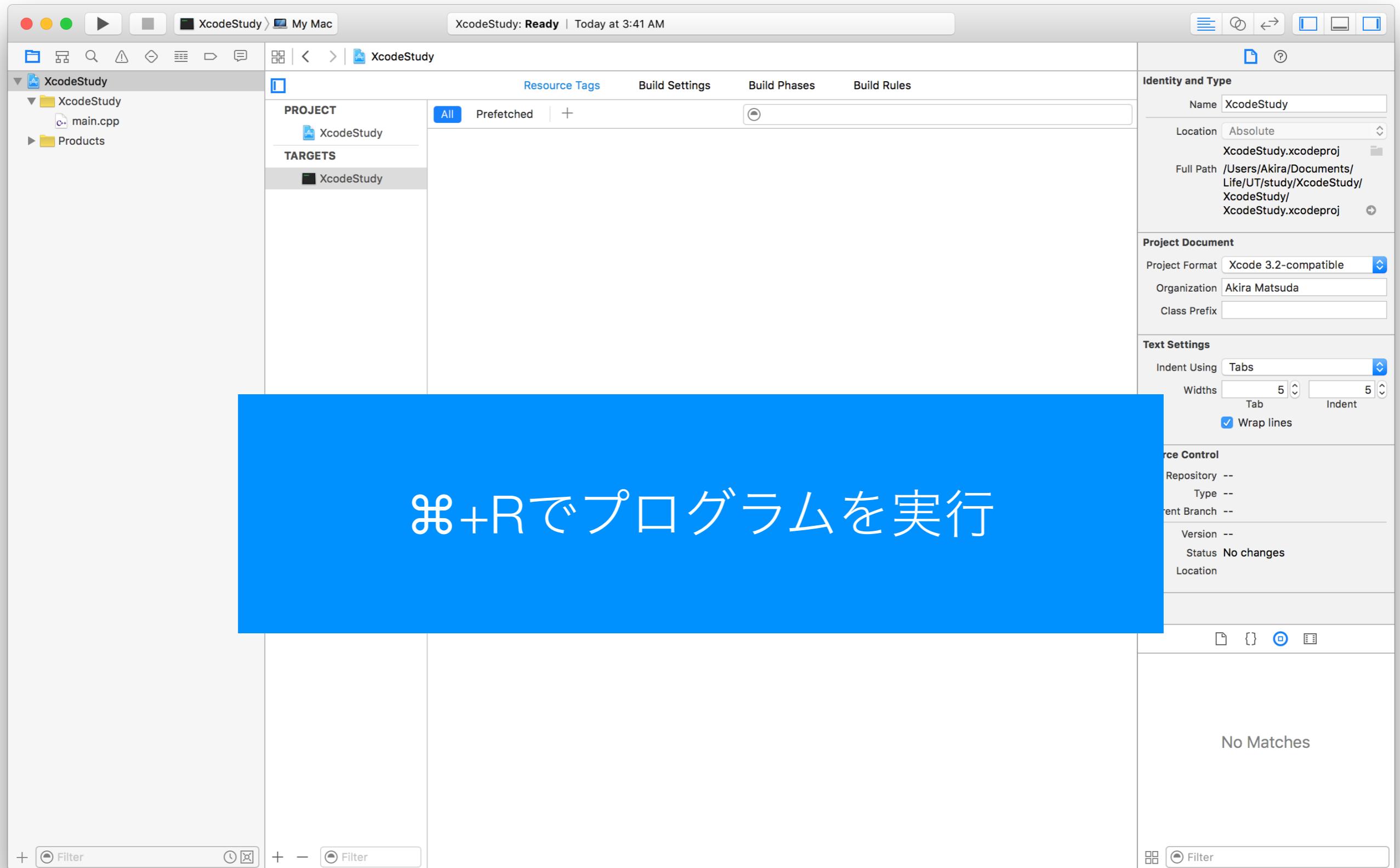
- .xcodeprojがプロジェクトファイル
- 複数のプロジェクトファイルを束ねる.xcworkspace

プロジェクトファイルに
ソースファイルやリソースを追加していく

プロジェクトを作ってみよう～～

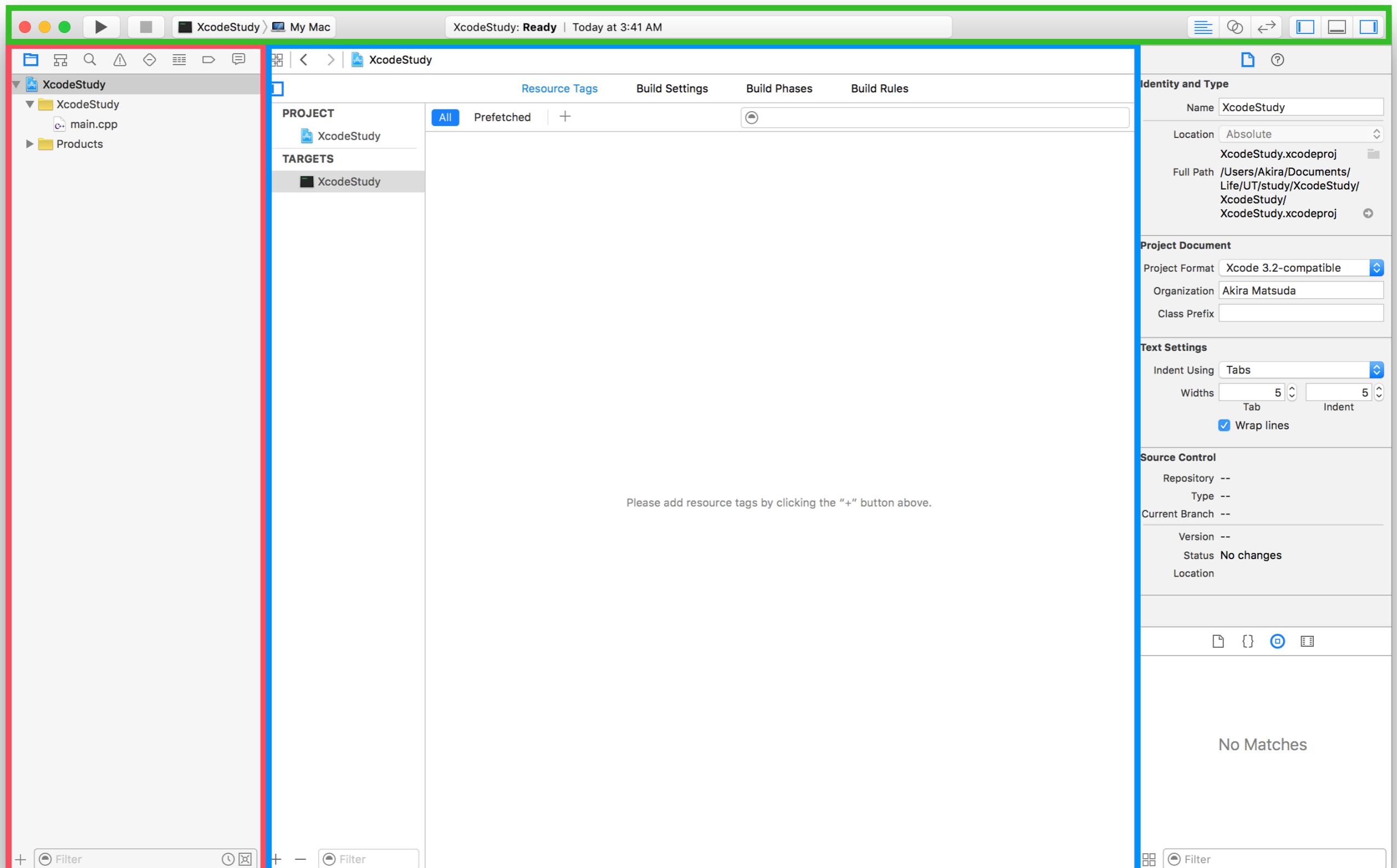
- Xcodeを起動
- ⌘+Shift+Nで新規プロジェクトを作る
- OS X > Application > Command line tool
- Project Nameに「XcodeStudy」
- Languageは「C++」





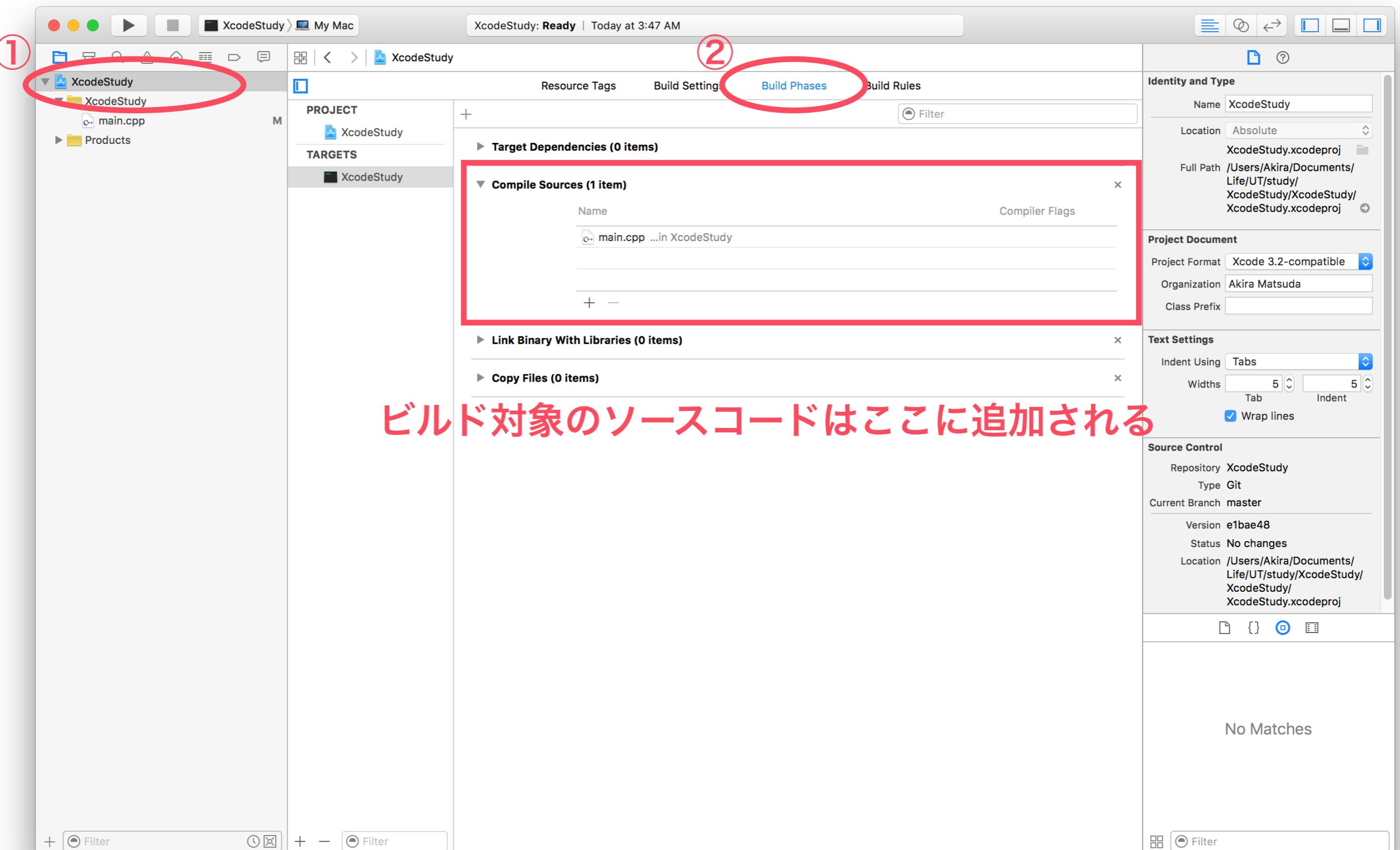
Hello, World!

Toolbar Area



Navigator Area

Editor Area



デバッグ編

作ってみよう～～

- FizzBuzzをC++で実装
- FizzBuzzクラスを作る
 - 値を引数に渡すとenumの値がreturnされる
 - 結果だけ返すクラスメソッド版と今までの履歴を保持するインスタンスメソッド版を作る

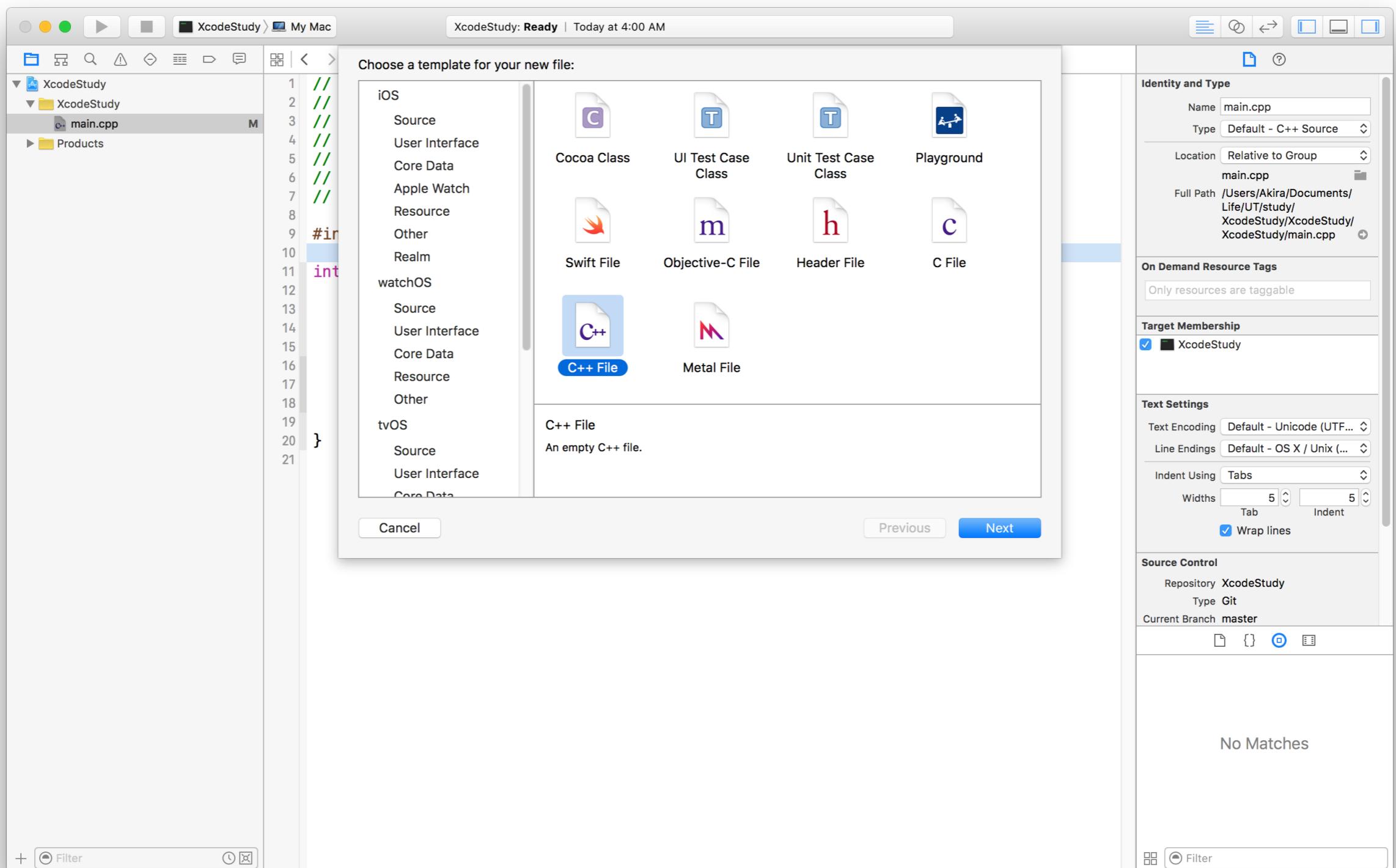
(復習) FizzBuzzってなんだっけ

- ある値が...
 - 3の倍数ならFizz
 - 5の倍数ならBuzz
 - 15の倍数ならFizzBuzz
- と出力するプログラム

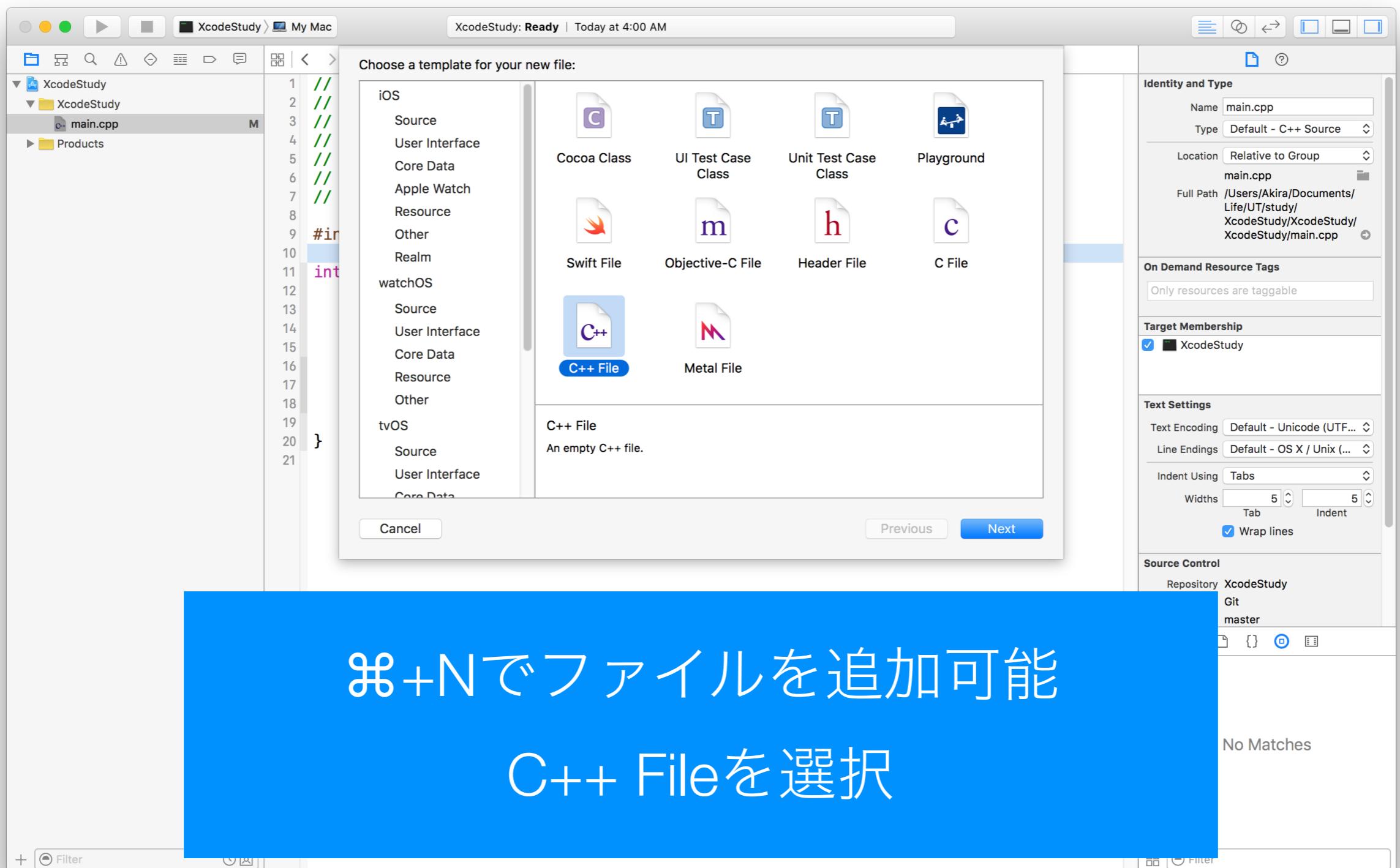
判定部分の実装

- namespaceはFizzBuzz
- Judgementクラスの中に実際の処理を実装

クラスの追加



クラスの追加



```
namespace FizzBuzz {
    typedef enum {
        None,
        Fizz,
        Buzz,
        FizzBuzz
    } Result;

    class Judgement {
        public:
            static Result judge(int num) {
                Result r = None;
                if (num % 3 == 0) {
                    r = Fizz;
                }
                else if (num % 5 == 0) {
                    r = Buzz;
                }
                else if (num % 3 == 0 && num % 5 == 0) {
                    r = FizzBuzz;
                }

                return r;
            }
    };
}
```

```
int main(int argc, const char * argv[]) {
    // insert code here...
    std::cout << ">> ";
    int num = 0;
    std::cin >> num;
    for (int i = 1; i <= num; i++) {
        std::cout << i << " : ";
        FizzBuzz::Result r = FizzBuzz::Judgement::judge(i);
        switch (r) {
            case FizzBuzz::Result::Fizz :
                std::cout << "Fizz";
                break;
            case FizzBuzz::Result::Buzz :
                std::cout << "Buzz";
                break;
            case FizzBuzz::Result::FizzBuzz :
                std::cout << "FizzBuzz";
                break;
            default:
                break;
        }
        std::cout << std::endl;
    }

    return 0;
}
```

```
>> 15
1 :
2 :
3 : Fizz
4 :
5 : Buzz
6 : Fizz
7 :
8 :
9 : Fizz
10 : Buzz
11 :
12 : Fizz
13 :
14 :
15 : Fizz
Program ended with exit code: 0
```

```
>> 15
1 :
2 :
3 : Fizz
4 :
5 : Buzz
6 : Fizz
7 :
8 :
9 : Fizz
10 : Buzz
11 :
12 : Fizz
13 :
14 :
→ 15 : Fizz
Program ended with exit code: 0
```

(↗ ↘)

何がおかしいのか調べてみよう～

- printfデバッグはいちいちコンパイルし直す必要ある
 - (ノ^)
- 脳内でプログラムの挙動を完全に再現
 - (ノ^)
- llDbを使う

行番号をクリックしてbreakpointを追加



```
27     static Result judge(int num) {  
28         Result r = None;  
29         if (num % 3 == 0) {  
30             r = Fizz;  
31         }  
32         else if (num % 5 == 0) {  
33             r = Buzz;  
34         }  
35         else if (num % 3 == 0 && num % 5 == 0) {  
36             r = FizzBuzz;  
37         }  
38  
39         return r;  
40     }
```

XcodeStudy > My Mac Running XcodeStudy : XcodeStudy

XcodeStudy PID 48908

CPU 0% Memory 568 KB Energy Impact Zero Disk Zero KB/s Network Zero KB/s

Thread 1 Queue: com....-thread (serial)

0 FizzBuzz::Judgement::judge(int)

1 main

2 start

3 start

```
#include <iomanip>
namespace FizzBuzz {
    typedef enum {
        None,
        Fizz,
        Buzz,
        FizzBuzz
    } Result;

    class Judgement {
public:
        static Result judge(int num) {
            Result r = None;
            if (num % 3 == 0) {
                r = Fizz;
            }
            else if (num % 5 == 0) {
                r = Buzz;
            }
            else if (num % 3 == 0 && num % 5 == 0) {
                r = FizzBuzz;
            }

            return r;
        }
    };
}
```

No Selection

No Matches

A num = (int) 1
L r = (FizzBuzz::Result) 1982411512
>> 15
(lldb) |

All Output

The screenshot shows the Xcode interface during a debugging session. The top navigation bar indicates "Running XcodeStudy : XcodeStudy". The left sidebar displays system resource usage for "XcodeStudy PID 48908" and a stack trace for Thread 1. The main editor area shows C++ code for the FizzBuzz problem. A breakpoint is set at line 26, which is highlighted in light blue. The bottom-left panel, framed in red, shows the memory dump for variable "num" (value 1) and variable "r" (value 1982411512). The bottom-right panel, framed in blue, shows the lldb command-line interface with the command ">> 15".

```
#include <map>
namespace FizzBuzz {
    typedef enum {
        None,
        Fizz,
        Buzz,
        FizzBuzz
    } Result;

    class Judgement {
public:
    static Result judge(int num) {
        Result r = None;
        if (num % 3 == 0) {
            r = Fizz;
        }
        else if (num % 5 == 0) {
            r = Buzz;
        }
        else if (num % 3 == 0 && num % 5 == 0) {
            r = FizzBuzz;
        }

        return r;
    }
}
```

Scopes:

- num = (int) 1
- r = (FizzBuzz::Result) 1982411512

lldb:

```
>> 15
```

スコープ内の変数情報

コンソール

The screenshot shows the Xcode interface during a debugging session. The top navigation bar indicates "Running XcodeStudy : XcodeStudy". The left sidebar displays system resource usage (CPU, Memory, Energy Impact, Disk, Network) and a thread list for Thread 1. The main editor area shows the FizzBuzz.cpp code with a breakpoint at line 26. The bottom section contains two panes: the "Scopes" pane (red border) showing variable information for "num" and "r", and the "Console" pane (blue border) showing the lldb command prompt with an arrow pointing to the input field.

Scopes

- A num = (int) 1
- L r = (FizzBuzz::Result) 1982411512

Console

```
>> 15  
(lldb)
```

lldbの入力待ち

スコープ内の変数情報

コンソール

XcodeStudy > My Mac Running XcodeStudy : XcodeStudy

XcodeStudy PID 48908 CPU 0% Memory 568 KB Energy Impact Zero Disk Zero KB/s Network Zero KB/s Thread 1 Queue: com....-thread (serial) 0 FizzBuzz::Judgement::judge(int)

```
#include <iomanip>
namespace FizzBuzz {
    typedef enum {
        None,
        Fizz,
        Buzz,
        FizzBuzz
    } Result;

    class Judgement {
public:
        static Result judge(int num) {
            Result r = None;
            if (num % 3 == 0) {
                r = Fizz;
            }
            else if (num % 5 == 0) {
                r = Buzz;
            }
            else if (num % 3 == 0 && num % 5 == 0) {
                r = FizzBuzz;
            }

            return r;
        }
    };
}
```

デバッガの操作ボタン

XcodeStudy > Thread 1 > 0 FizzBuzz::Judgement::judge(int)

A num = (int) 1
L r = (FizzBuzz::Result) 1982411512
>> 15
(lldb) |

No Matches

```
33         else if (num % 3 == 0 && num %  
34             r = FizzBuzz;  
35     }  
36  
37     return r;  
38 }  
39
```

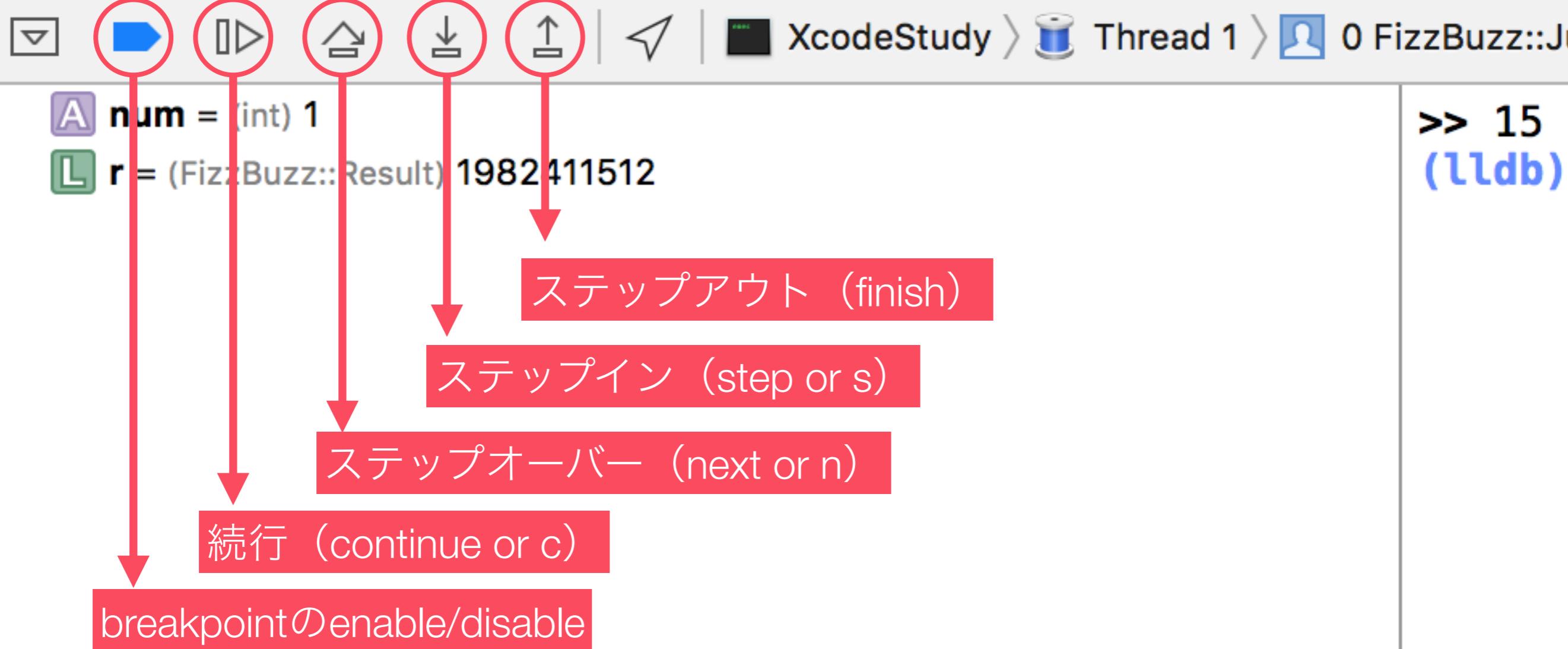
▽ ▶ ▷ ⌂ ⌄ ⌅ ⌆ | ↵ | XcodeStudy > Thread 1 > 0 FizzBuzz::J

A num = (int) 1

L r = (FizzBuzz::Result) 1982411512

>> 15
(lldb)

```
33     else if (num % 3 == 0 && num %  
34         r = FizzBuzz;  
35     }  
36  
37     return r;  
38 }  
39
```



続行 or ステップオーバーの連打は (,)

breakpointの操作

- ・ 次の条件が指定可能
 - ・ 止まる条件
 - ・ 例) `n == 5` の時のみ止まる
 - ・ 止まった時の処理
 - ・ 例) 止まった時にlldbのコマンドを実行
 - ・ 止まらずに続行
 - ・ 例) ↑でコマンドだけ実行し処理を続ける

```
public static Result judge(int num) {
    Result r = None;
    if (num % 3 == 0)
        r = Fizz;
    if (num % 5 == 0)
        r = Buzz;
    else if (num % 3 == 0)
        r = FizzBuzz;
    return r;
}

int main() {
    num = 3;
    judge();
}
```

A **num** = (int) 3

```
static Result j
Result r =
if (num %
r = F
if (n
r = B
if (n
```

FizzBuzz.hpp:26

Condition num == 15

Ignore 0 times before stopping

Action

Options Automatically continue after evaluating actions

       XcodeS

A num = (int) 3

XcodeStudy > My Mac Running XcodeStudy : XcodeStudy

XcodeStudy PID 48908

CPU 0%
Memory 568 KB
Energy Impact Zero
Disk Zero KB/s
Network Zero KB/s

Thread 1 Queue: com....-thread (serial)

0 FizzBuzz::Judgement::judge(int)

1 main
2 start
3 start

```
14
15 namespace FizzBuzz {
16     typedef enum {
17         None,
18         Fizz,
19         Buzz,
20         FizzBuzz
21     } Result;
22
23     class Judgement {
24     public:
25         static Result judge(int num) {
26             Result r = None;
27             if (num % 3 == 0) {
28                 r = Fizz;
29             }
29             else if (num % 5 == 0) {
30                 r = Buzz;
31             }
31             else if (num % 3 == 0 && num % 5 == 0) {
32                 r = FizzBuzz;
33             }
33
34             return r;
35         }
36
37     };
38
39 // Result judge_and_record(int num)

```

Thread 1: step over

num = (int) 15
r = (FizzBuzz::Result) None

>> 15
1 :
2 :
3 : Fizz
4 :
5 : Buzz
6 : Fizz
7 :
8 :
9 : Fizz
10 : Buzz
11 :
12 : Fizz
13 :
14 :
(lldb)

All Output

No Matches

Identity and Type

Name FizzBuzz.hpp
Type Default - C++ Header
Location Relative to Group
FizzBuzz.hpp
Full Path /Users/Akira/Documents/Life/UT/study/XcodeStudy/XcodeStudy/XcodeStudy/FizzBuzz.hpp

On Demand Resource Tags

Add to a target to enable tagging

Target Membership

XcodeStudy

Text Settings

Text Encoding Unicode (UTF-8)
Line Endings Default - OS X / Unix (...)
Indent Using Tabs
Widths 5 Tab 5 Indent
Wrap lines

Source Control

Repository XcodeStudy
Type Git

15は3の倍数でもある



$$15 \% 3 == 0$$

バグの原因

- 15は3の倍数であった
 - つまり3で割った時のおまりは0
 - 最初のif文に引っかかり他のelse ifに入らず終了

正解

```
static Result judge(int num) {  
    Result r = None;  
    if (num % 3 == 0 && num % 5 == 0) {  
        r = FizzBuzz;  
    }  
    else if (num % 3 == 0) {  
        r = Fizz;  
    }  
    else if (num % 5 == 0) {  
        r = Buzz;  
    }  
  
    return r;  
}
```

履歴の保持

- 方針
 - 入力と結果の配列を作る
 - 配列はstd::vector、入力と結果の対はstd::tupleで実装

```
15 namespace FizzBuzz {
16     typedef enum {
17         None,
18         Fizz,
19         Buzz,
20         FizzBuzz
21     } Result;
22
23     class Judgement {
24     private:
25         std::vector<std::tuple<int, Result>> history;
26     public:
27         static Result judge(int num) { ... }
28
29         Result judge_and_record(int num) {
30             Result r = FizzBuzz::Judgement::judge(num);
31             auto t = std::make_tuple(num, r);
32             this->history.push_back(t);
33
34             return r;
35         }
36
37         std::vector<std::tuple<int, Result>> get_history() {
38             return this->history;
39         }
40     };
41 }
```

```
15 namespace FizzBuzz {
16     typedef enum {
17         None,
18         Fizz,
19         Buzz,
20         FizzBuzz
21     } Result;
22
23     private:
24         std::vector<std::tuple<int, Result>> history;
25
26     public:
27         static Result judge(int num) { ... }
28
29         Result judge_and_record(int num) {
30             Result r = FizzBuzz::Judgement::judge(num);
31             auto t = std::make_tuple(num, r);
32             this->history.push_back(t);
33
34             return r;
35         }
36
37         std::vector<std::tuple<int, Result>> get_history() {
38             return this->history;
39         }
40     };
41
42 }
```

灰色の部分をクリックするとfolding可能

```
15 namespace FizzBuzz {  
16     typedef enum {  
17         None,  
18         Fizz,  
19         Buzz,  
20         FizzBuzz  
21     } Result;
```

autoでいい感じに変数宣言を省略可能 (C++11)

```
24     private:  
25         std::vector<std::tuple<int, Result>> history;  
26     public:  
27         static Result judge(int num) { ... }  
41  
42         Result judge_and_record(int num) {  
43             Result r = FizzBuzz::Judgement::judge(num);  
44             auto t = std::make_tuple(num, r);  
45             this->history.push_back(t);  
46  
47             return r;  
48         }  
49  
50         std::vector<std::tuple<int, Result>> get_history() {  
51             return this->history;  
52         }  
53     };  
54 }
```



結果の出力

```
for (std::tuple<int, FizzBuzz::Result> t : j.get_history()) {
    int num = std::get<0>(t);
    FizzBuzz::Result r = std::get<1>(t);
    std::string result = "";
    switch (r) {
        case FizzBuzz::Result::Fizz :
            result = "Fizz";
            break;
        case FizzBuzz::Result::Buzz :
            result = "Buzz";
            break;
        case FizzBuzz::Result::FizzBuzz :
            result = "FizzBuzz";
            break;
        default:
            break;
    }
    std::cout << num << " : " << result << std::endl;
}
```

長い... ('ʌ')

```
28         case FizzBuzz::Result::FizzBuzz :
29             std::cout << "FizzBuzz";
30             break;
31         default:
32             break;
33     }
34     std::cout << std::endl;
35 }
36
37 FizzBuzz::Judgement j;
38 for (int i = 1; i <= num; i++) {
39     j.judge_and_record(i);
40 }
41
42 >>> return 0; Thread 1: breakpoint 1.1
43 }
```

□ ▶ ⏪ ⏴ ⏵ ⏵ | ↵ | XcodeStudy > Thread 1 > 0 main

```
>> 15
1 :
2 :
3 : Fizz
4 :
5 : Buzz
6 : Fizz
7 :
8 :
9 : Fizz
10 : Buzz
11 :
12 : Fizz
13 :
14 :
15 : FizzBuzz
(lldb)
```

おもむろに

(lldb) po j

```
(lldb) po j
{
    size=15
{
    {
        {
            (value = 1)
            (value = None)
        }
    }
    {
        {
            (value = 2)
            (value = None)
        }
    }
    {
        {
            (value = 3)
            (value = Fizz)
        }
    }
    {
        {
    }
All Output ◊
```



XcodeStudy > Thread 1 > 0 main



よく使うlldbのコマンド

- **p, po**
 - 変数の中身を見る
 - pはプリミティブ, poはオブジェクトの中身を見れる
- **bt**
 - バックトレースを見る
 - -n <数字>で幾つ前まで見るか指定可能

その他 – よく使う設定項目

Build Settings

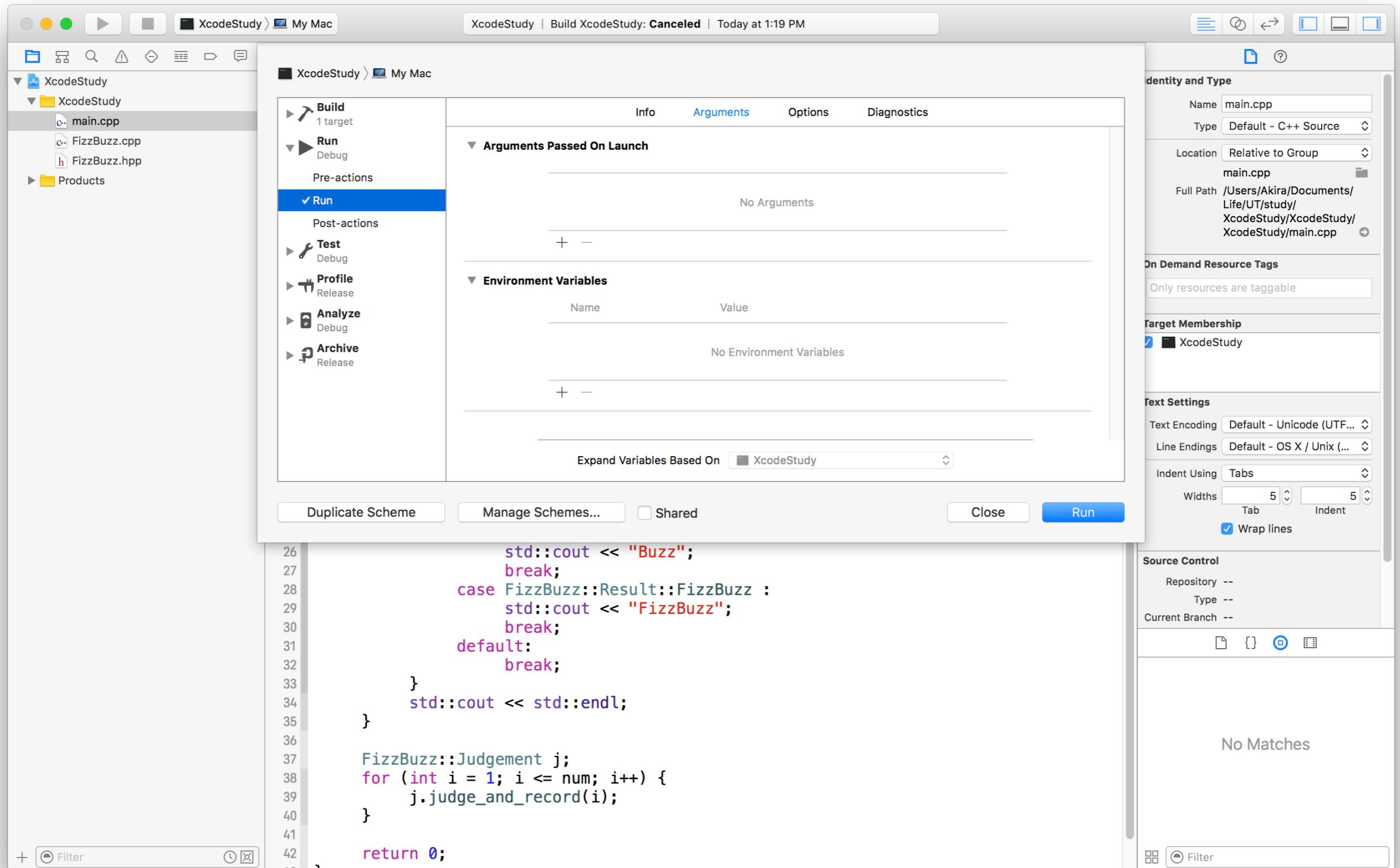
- Other Linker Flag
 - コンパイル時オプション
- Header Search Path
 - 含めるヘッダのパス。indexingの対象になる。
- Library Search Path
 - 含めるライブラリのパス

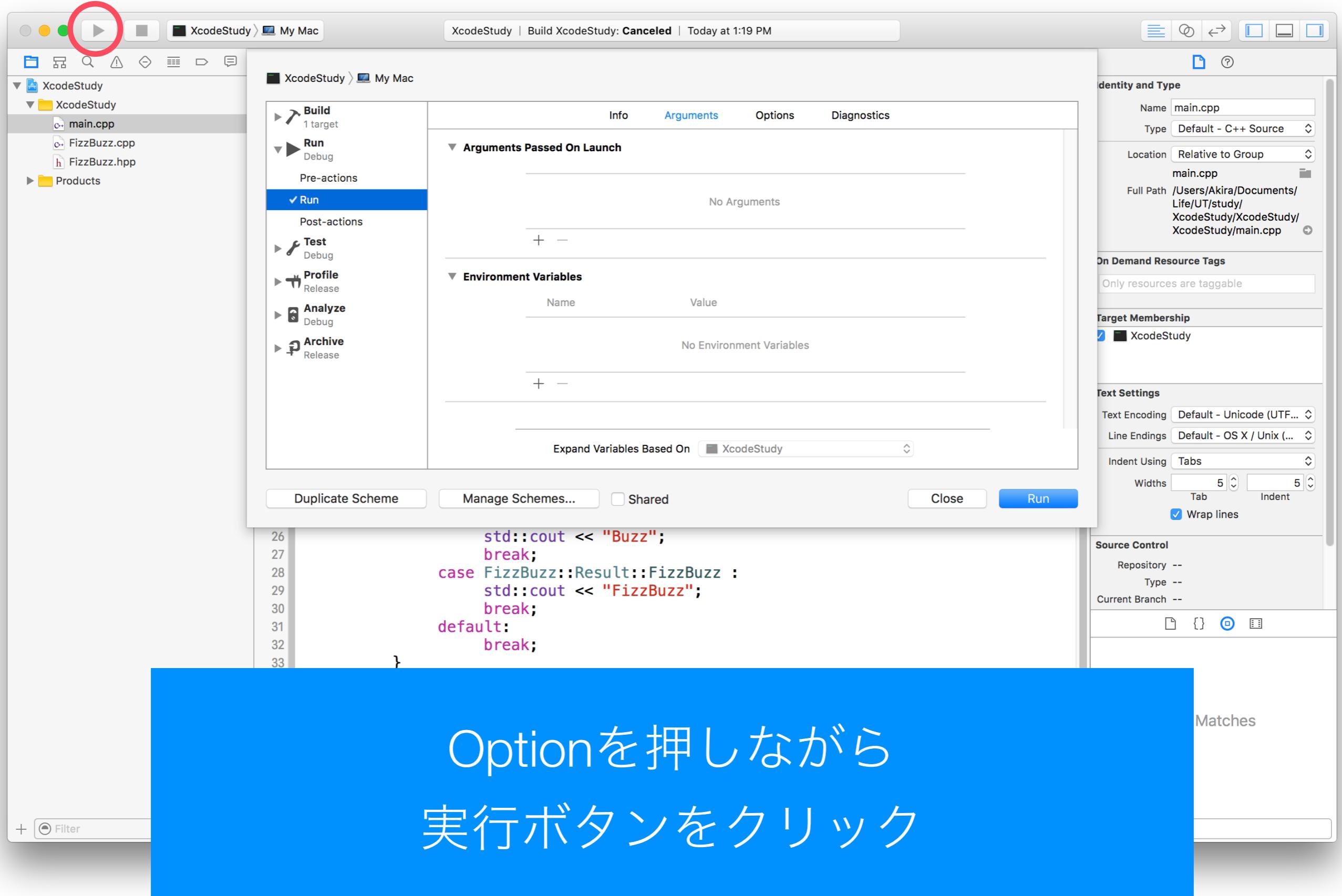
Build Phases

- Compile Sources
 - コンパイル対象となるソースコード
- Run Script
 - ビルド後に走るスクリプト

Arguments

- Arguments Passed On Launch
 - 実行時の引数の指定
- Environment Variables
 - 実行時の環境変数





Optionを押しながら
実行ボタンをクリック

開発を楽にしてくれるツール

Alcatraz

- Xcode用のパッケージマネージャ

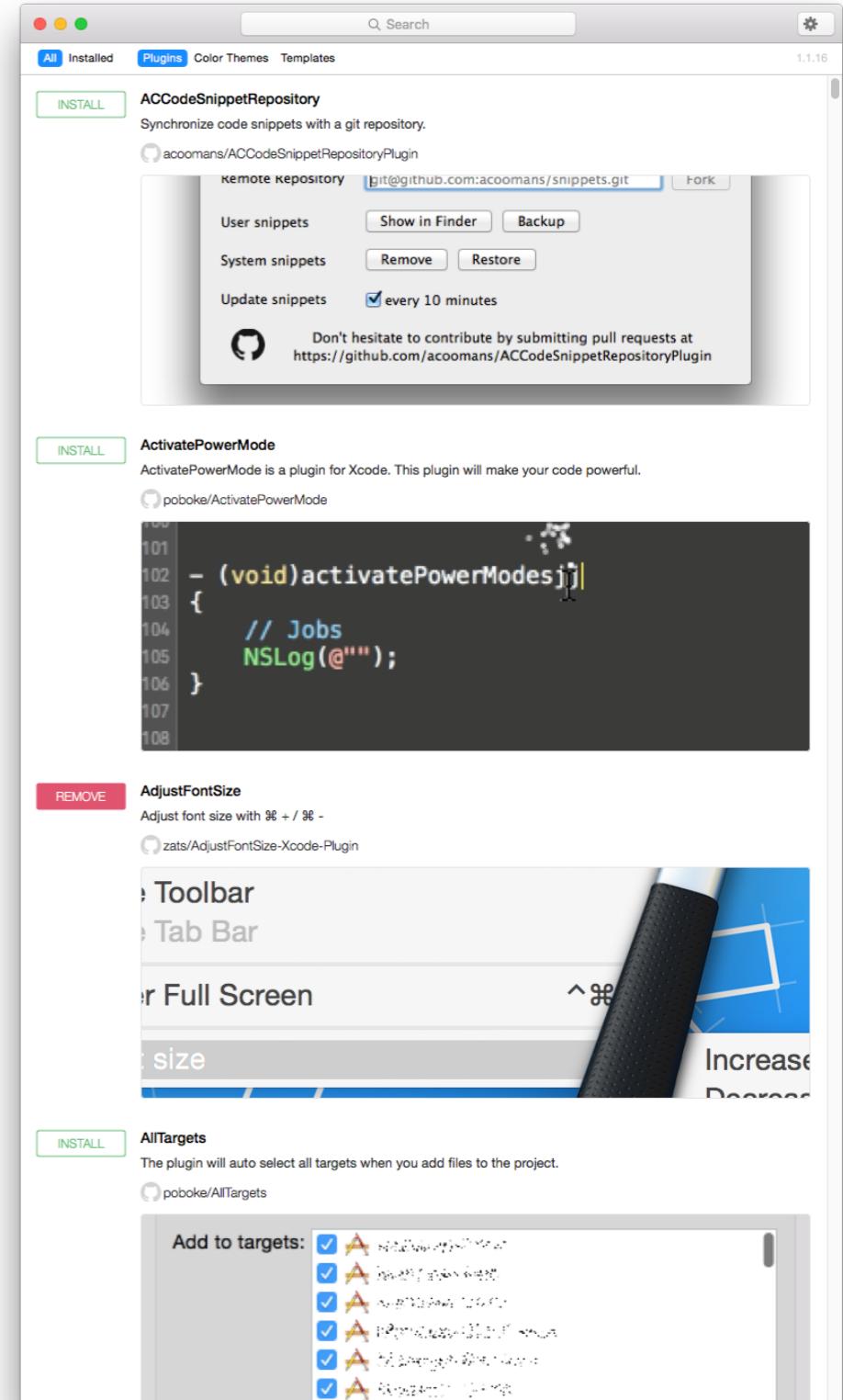
- 便利なプラグインがワンクリックでインストール可能

- iOS, OS X向けのプラグインが多い

- Xcodeそのものの挙動を変えるものも

- Xcodeのテーマも配布されている

- <http://alcatraz.io>



Alcatraz

- ・ おすすめプラグイン
 - ・ AdjustFontSize — フォントサイズを⌘+, ⌘-で変更
 - ・ Backlight — カーソルのいる行をハイライト
 - ・ ClangFormat — ClangFormatを実行
 - ・ FuzzyAutocomplete — 曖昧な入力補完が可能
 - ・ SCXcodeSwitchExpander — switch文をいい感じに展開
 - ・ VVDocumenter-Xcode — ///でドキュメント用のスニペットを入力

appledoc

- **Objective-Cのコードからドキュメントを生成**
 - Apple標準ドキュメントっぽいデザイン
 - 何かライブラリを配布するときに使うと◎
- <https://github.com/tomaz/appledoc>

Dash

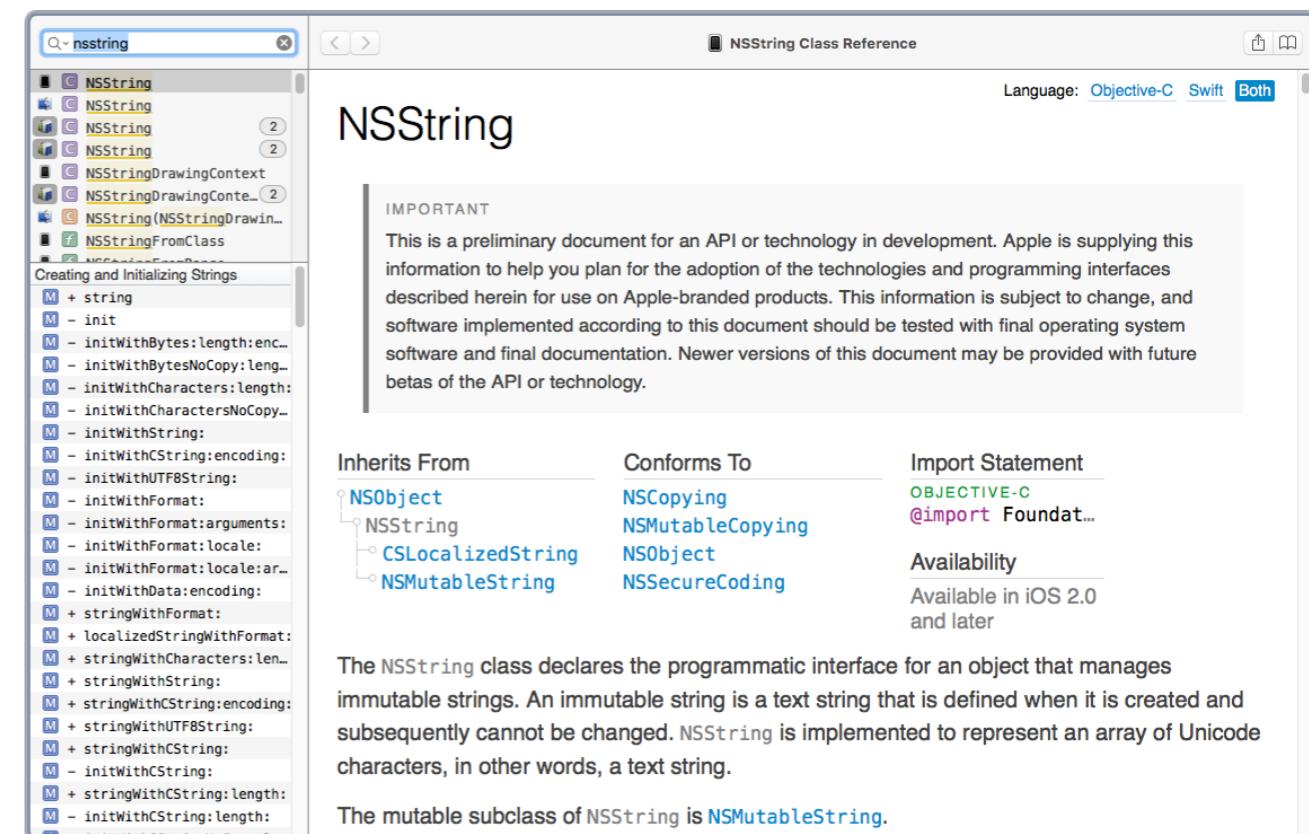
- 使い勝手の良いドキュメントビューアー

- ショートカットキーで
すぐ呼び出せる

- ドキュメントが豊富

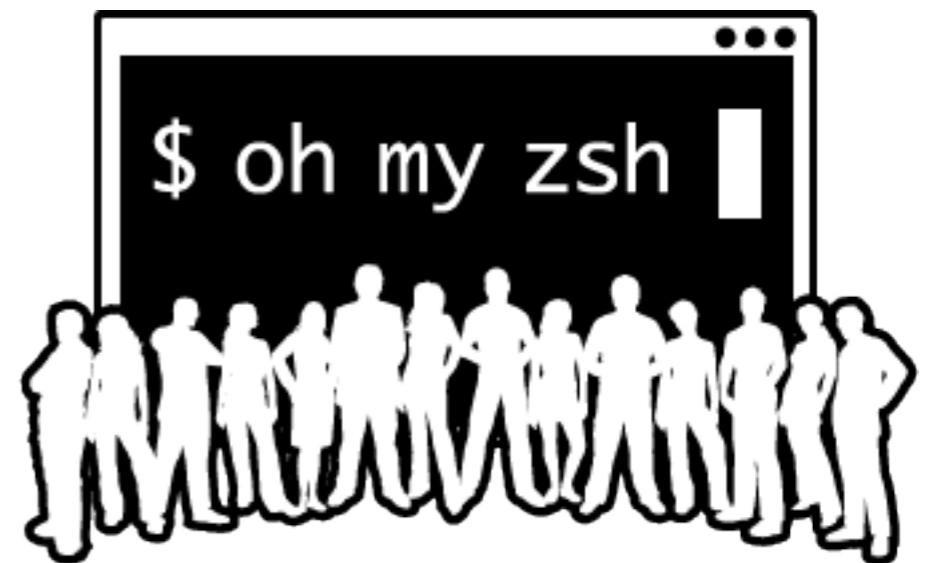
- Obj-Cだけではない
- 自分でインストール可能

- <https://goo.gl/LrEAGT>



oh-my-zshとかPrezto

- zshを便利にしてくれるフレームワーク
 - ほぼコマンド一発でインストールできるので入れておくと良い
 - とりあえず入れておけば便利機能が使える
 - gitのブランチ表示
 - cdのディレクトリを矢印で移動
 - oh-my-zshは設計が古く重いらしい
 - <https://github.com/robbyrussell/oh-my-zsh>
 - <https://github.com/sorin-ionescu/prezto>
 - terminalを制すものは開発を制す

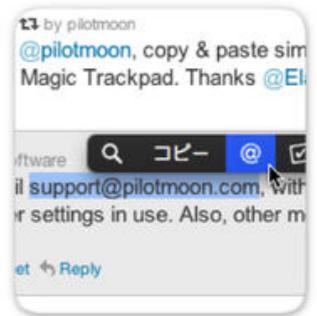
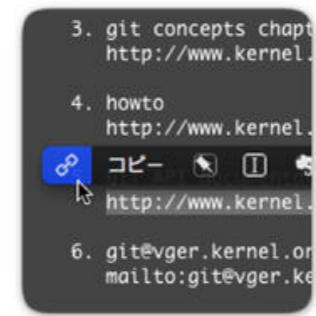
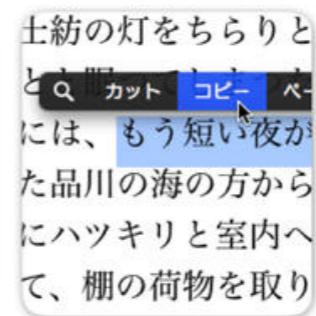
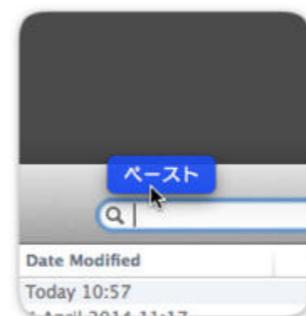


```
~ ➤ cd testproject
~/testproject ➤ ↵ master ➤ gco detached-head-state -q
~/testproject ➤ ↵ fdffaf6 ➤ touch dirty-working-directory
~/testproject ➤ ↵ fdffaf6± ➤ cd
~ ➤ ssh milly
Welcome to Ubuntu 11.04 (GNU/Linux 2.6.18-308.8.2.el5.028stab101.1 x86_64)
Last login: Wed Sep 26 03:42:49 2012 from 71-215-222-90.mpls.qwest.net
agnoster@milly: ~
Connection to milly.agnoster.net closed.
~ ➤ sudo -s
Password:
⚡ root@Arya: ~ ➤ top &
[1] 34523
[1] + 34523 suspended (tty output) top
⚡ ⚡ root@Arya: ~ ➤ rm no-such-file
rm: no-such-file: No such file or directory
✖ ⚡ ⚡ root@Arya: ~ ➤ kill %
[1] + 34523 terminated top
⚡ root@Arya: ~
~ ➤
```

その他便利ツール

PopClip

- 選択したテキストをいい感じに処理してくれるツール
 - コピペ, 検索, 文字数カウント, 辞書, 翻訳
- <https://goo.gl/wLoMw>



BetterSnapTool

- ・ ウィンドウをいい感じに
リサイズしてくれるツール
- ・ Windows7の画面端にウィンドウを
持っていくとリサイズされる機能と同じ
- ・ ショートカットキーで操作も可能
- ・ <https://goo.gl/k2uLbI>



XtraFinder

- **Finderに便利機能を追加**
 - タブ, Split View, パスのコピー
 - rootlessのせいで開発止まったっぽい
 - 無効にすれば使えるようになる
- <https://www.trankynam.com/xtrafinder/>



Go2Shell

- Finderで開いている場所から
ターミナルを開く
 - いちいちcdして移動しなくて良い
 - <https://goo.gl/MukxSo>



enjoy ;)