たった２日でマスターできる

iPhone アプリ開発 集中講座

Xcode 11 | Swift 5 | 対応

藤治仁・小林加奈子・小林由憲［著］

ソシム

●商標等について

・Apple、iCloud、iPad、iPad Air、iPad mini、iPhone、iPod touch、Mac、Macintosh、macOS、Objective-C、Swift、Xcode は、米国およびその他の国々で登録された Apple Inc. の商標です。

・その他、本書に記載されている社名、製品名、ブランド名、システム名などは、一般に商標または登録商標で、それぞれ帰属者の所有物です。

・本文中では、©、®、™ は表示していません。

●諸注意

・本書はソシム株式会社が出版したもので、本書に関する権利、責任はソシム株式会社が保有します。

・本書に記載されている情報は、2019 年 10 月現在のものであり、URL などの各種の情報や内容は、ご利用時には変更されている可能性があります。

・本書の内容は参照用としてのみ使用されるべきものであり、予告なしに変更されることがあります。また、ソシム株式会社がその内容を保証するものではありません。本書の内容に誤りや不正確な記述がある場合も、ソシム株式会社は一切の責任を負いません。

・本書に記載されている内容の運用によっていかなる損害が生じても、ソシム株式会社および著者は責任を負いかねますので、あらかじめご了承ください。

・本書のいかなる部分についても、ソシム株式会社との書面による事前の同意なしに、電気、機械、複写、録音、その他のいかなる形式や手段によっても、複製、および検索システムへの保存や転送は禁止されています。

はじめに

この書籍は、**「iOS アプリを作ってみたい、すべての初心者が、体験から学べる入門書」**です。

iOS とは、Apple が macOS をベースに開発した、iPhone、iPad、iPod touch 向けのモバイル OS です。本書籍では、プログラム言語の Swift（スウィフト）を用いて開発を行います。

執筆陣は、2014 年 11 月 1 日から**「Swift ビギナーズ倶楽部」**というコミュニティを開催していました。プログラミング自体がはじめての方、iOS アプリを作ってみたい他分野のエンジニアの方、そして高校生から、定年退職された方まで、たくさんの方々にご参加いただいています。最初は試行錯誤しながらも、勉強会に参加をして参加者の方どうしで教え合い、アプリ開発を楽しまれている方々が多くいらっしゃいます。

その経験を通して執筆陣が理解したことは、はじめての分野を学習する際には、経験者にとってはどんな些細なことも、初学者にとってはつまずくポイントになりえるということです。ただ、どんな分野もそうですが、最初は基礎的なことを、まねごとでよいので丁寧に繰り返し練習すれば、少しずつ自分だけのオリジナルの発想が出てきます。

本書では、構成段階からハンズオンセミナーを開催し、たくさんの参加者の方々の声をまとめ、各レッスンを構成しました。また、継続してセミナーを開催しながら、どの操作でつまずくのか、どのコードの説明が理解しがたいのかをフィードバックを受けて、教材を改善し続けています。

そうした調査に基づき、プログラミングを通して、**「モノづくり」の楽しさを体験**していただけるように、少しずつ階段を上っていく体験を重視した構成にしました。初心者が最初の一歩を踏み出す書籍を目指しています。

本書のコンセプトは**「まずは体験してみて、その経験を生かして学んでいく」**です。難しいプログラミング文法の説明は極力最小限にまとめ、多くのサンプルアプリの開発を体験してもらうことで、最短距離でアプリ開発の「勘所」がつかめるように工夫しました。

これから iOS アプリを制作される方々の一助となれば幸いです。

ようこそ、iOS アプリ開発の世界へ。

目次 CONTENTS

はじめに 3
この本の読み方と使い方 10
ご利用の前に必ずお読みください 14

Day 1 Day 2

Lesson 1 はじめてのアプリを開発する前に知っておこう 15

iOS アプリの開発を始める前に、知っておいた方がよいことを学びます。
書籍を読み進めていく上で、最初の心構えを知っておくことで、心理的なハードルが下がり、学習が行いやすくなります。

1-1 プログラミングを体験から学んでいこう 16
1 この本の使い方と前提知識 16
1-2 あらかじめ挫折しそうなポイントを押さえておこう 18
1 学習ポイントを押さえよう 18
1-3 アプリ開発をするなら知っておこう！ ～WWDC、手数料、課金方法～ 20
1 WWDC での公開情報をもとに考察してみよう 20
1-4 Swift（スウィフト）を知ろう 22
1 Swift の特徴を押さえよう 22

Lesson 2 アプリ開発の環境を整えて、Xcode の使い方を学ぼう 25

iOS アプリを開発するために、必要なものを学びましょう。
最初に、まだ Apple ID を取得していない方のために、Apple ID の基礎知識と取得方法についても解説します。

2-1 開発するために必要な準備をしよう 26
1 アプリ開発に必要な 3 つのものを準備しよう 26
2-2 Apple ID を取得しよう 28
1 Apple ID をすでに取得されている方 28
2 Apple ID をまだ取得されていない方 29
2-3 Xcode をインストールしよう 31
1 Xcode をダウンロードしよう 31
2-4 Xcode を起動して、プロジェクトを作成しよう 33
1 Xcode を起動しよう 33

2 プロジェクトを作成しよう 34
3 Xcode 画面のナビゲーションを理解しよう 38
2-5 Xcode をより使いやすくするための設定をしよう 43
1 Xcode の環境を設定しよう 43
2-6 「Hello Swift!」と表示してみよう 46
1 Label を配置して、AutoLayout を設定してみよう 48
2 プログラムを書いてみよう 51
3 シミュレータを起動してみよう 56
4 実機転送を行い、iPhone で確認してみよう 59

Lesson 3 じゃんけんアプリを作ろう —Swift の基本を学ぶ— 67

はじめての iPhone アプリ「じゃんけんアプリ」を作ります。じゃんけんアプリを作りながら、アプリ開発の基本を学びます。
最後に「ステップアップ」があります。もう少し深く学習してみたい方はチャレンジしてみてください！

3-1 完成をイメージしよう 68
3-2 プロジェクトを作成しよう 70
1 プロジェクトを作成しよう 70
3-3 必要な部品を配置しよう 72
1 Storyboard にパーツを配置してみよう 72
2 パーツを装飾してみよう 75
3 Image View パーツで使う画像ファイルを設定しよう 84
3-4 部品の表示位置、幅、高さを設定しよう 86
1 AutoLayout（オートレイアウト）の概要を理解しよう 86
2 作業しやすいように画面を整えよう 86
3 Button の位置と幅・高さを設定しよう 87
4 Label の位置と幅・高さを設定しよう 93
5 Image View の位置と幅・高さを設定しよう 94
3-5 部品とプログラムを関連付けしよう 97
1 コードを書きやすいように Xcode 画面を整理しよう 97
2 Image View をコードと関連付けしよう 99
3 Label をコードと関連付けしよう 102
4 Button をコードと関連付けしよう 103
3-6 じゃんけん画像を切り替えよう 108
1 プログラムを書く前に、大きく作り方を把握しよう 108
3-7 ステップアップ アイコンを設定しよう 124
1 アイコンのサイズや使用用途を確認する 125
2 アイコンの作成と設定をしよう 126

Lesson 4 音楽アプリを作ろう
—音の扱い方を学ぶ—

131

この章では、iOS アプリ開発で音を扱う方法を学びます。また、AutoLayout の設定を繰り返し行い、習得をすることを目的としています。

4-1 完成をイメージしよう 132
1 プロジェクトを作成しよう 133
4-2 シンバルとギターを配置しよう 134
1 Storyboard にパーツを配置しよう 134
4-3 タップで音を鳴らそう 150
1 音を扱う準備をしよう 150
2 音を便利に扱うことができる「AVFoundation」を読み込もう 152
3 シンバルの画像とプログラムコードをつなごう 153
4 シンバルを鳴らそう 154
5 ギターの画像とプログラムコードをつなごう 159
6 「Play」「Stop」とプログラムをつなごう 160
4-4 ステップアップ リファクタリングで見通しを改善しよう 165

Lesson 5 マップ検索アプリを作ろう
—UI パーツと delegate—

171

マップを表示させることができる「MapKit」を利用して、マップアプリを開発します。テキストエリアにキーワードを入力すると、該当する場所を検索し、ピンを立てます。クロージャ、delegate の概念についても説明します。

5-1 完成をイメージしよう 172
5-2 画面を作成しよう 174
1 プロジェクトを作成しよう 174
2 使用する UI パーツを配置しよう 175
3 AutoLayout で、レイアウトを整えよう 178
4 関連付けをしよう 179
5-3 マップ検索アプリの動作をプログラミングしよう 182
1 検索キーワードを入力するために Text Field を使う 182
2 キーワードから緯度経度を検索する 189
5-4 ステップアップ マップの種別（航空写真など）を切り替えできるようにしよう 198
1 マップの種別を切り替えるボタンを配置しよう 199
2 マップの種別切り替え処理を実装しよう 202

Day 1 Day 2

Lesson 1 タイマーアプリを作ろう
―画面遷移とデータの永続化―

209

いままで単一画面で完結するアプリを作ってきましたが、この章では画面遷移の方法を学びます。
また、設定画面で選択した秒数をタイマー画面で利用できるように、設定画面でデータを保持しておく必要があるので、その方法も学びましょう。

1-1 完成をイメージしよう 210
1 プロジェクトを作成しよう 212
1-2 Storyboard でレイアウトを作ろう 213
1 ［Main.storyboard］を、Navigation Controller に変更しよう 213
2 Navigation Controller の仕組みを知ろう 215
3 タイマー画面の UI パーツを配置しよう 216
4 タイマー画面の AutoLayout を指定しよう 219
5 秒数の設定画面を追加しよう 227
6 画面遷移を行う Segue を追加しよう 228
7 設定画面の UI パーツを配置しよう 230
8 設定画面の各パーツに AutoLayout を設定しよう 233
9 設定画面の SettingViewController を追加しよう 234
10 ［Assistant editor］で確認しよう 237
11 タイマー画面の UI パーツとプログラムを関連付けよう 239
12 設定画面の UI パーツとプログラムを関連付けよう 243
1-3 タイマー画面を作成しよう 245
1 コードを書きやすいように準備しよう 245
2 変数の宣言を追加しよう 246
3 アプリ起動時に初期値を設定しよう 247
4 画面を更新するメソッドを作成しよう 249
5 経過時間を処理するメソッドを作成しよう 251
6 カウントダウンの処理を実装しよう 253
7 タイマーを停止する処理を実装しよう 256
8 秒数の設定画面に、画面遷移をしよう 257
9 秒数設定画面から戻ってきたら、設定した秒数で画面を更新しよう 259
10 シミュレータでカウントダウンを確認しよう 260
1-4 秒数設定画面を作成しよう 261
1 PickerView を利用できるようにしよう 261
2 変数を宣言しよう 262
3 設定画面起動時に、PickerView に保存している秒数をセットしよう 264
4 for 文とは？ 266
5 PickerView にデータを渡すメソッドを作成しよう 267
6 PickerView の内容を表示する delegate メソッドを作成しよう 268
7 画面で秒数を選択したときの delegate メソッドを作成しよう 269

8 「決定」ボタンがタップされたら、タイマー画面に遷移しよう 270
1-5 ステップアップ タイマー終了後にアラートを表示しよう 275
1 UIAlertController を追加しよう 276

Lesson 2 カメラアプリを作ろう［前半］
—カメラと SNS 投稿— 279

カメラを利用することができる「UIImagePickerController」と、カメラロールに保存したり SNS に投稿したりできる「UIActivityViewController」を利用して、カメラアプリを開発します。

2-1 完成をイメージしよう 280
2-2 画面を作成しよう 282
1 プロジェクトを作成しよう 282
2 使用する UI パーツを配置しよう 283
3 AutoLayout で、レイアウトを整えよう 286
4 UI パーツの詳細を設定しよう 288
5 UI パーツをプログラムと関連付けしよう 289
2-3 カメラを起動しよう 293
1 カメラの使用をユーザーに許可してもらおう 293
2 カメラが使用できる機種、できない機種を判定しよう 296
3 カメラを起動するコードを追加しよう 298
4 カメラで撮影した写真を表示させせよう 299
2-4 SNS 投稿機能を作成しよう 304
1 写真の保存をユーザーに許可してもらおう 304
2 SNS シェア機能を追加しよう 305
2-5 ステップアップ フォトライブラリーから写真を取り込めるようにしよう 309
1 カスタマイズしてみよう 310

Lesson 3 カメラアプリを作ろう［後半］
—エフェクト機能の追加— 319

カメラアプリの定番であるエフェクト機能を利用できる「Core Image」を利用して開発をします。
また、新しい画面遷移方法である「モーダルビュー」の指定方法と、画面遷移を実現する「Segue」（セグエ）の応用として、画面遷移時に情報を渡すことを学びます。

3-1 完成をイメージしよう 320
3-2 不要になったボタンを削除して、新しく画面を追加しよう 322
1 「SNS に投稿する」ボタンを削除しよう 322
2 新しい View Controller を追加しよう 324
3 View Controller とコードを結びつけよう 325
4 カメラ撮影画面(View Controller) から画面遷移できるようにしよう 329
5 使用する UI パーツを配置しよう 330

3-3 エフェクト画面を作成しよう 339
1 画面遷移をしてみよう 339
2 画面遷移して画像を表示してみよう 344
3 画面を閉じてみよう 345
4 画像にエフェクトをかけてみよう 347
5 画像をシェアしてみよう 353
3-4 ステップアップ エフェクトの種類を増やそう 356
1 カスタマイズしてみよう 357
2 フィルタの効果を確認しよう 360

Lesson 4 お菓子検索アプリを作ろう
—Web API と JSON の使い方を学ぶ— 363

iOS アプリでインターネットを通してデータを取得できると、作りたいアプリの可能性が広がります。お菓子に関するキーワードが入力されたら、インターネットからお菓子の情報を取得し、アプリの画面に表示するお菓子検索アプリを作ります。

4-1 完成をイメージしよう 364
4-2 Web API と JSON について学ぼう 366
1 Web API の基本的な仕組みを学ぼう 366
2 JSON と XML について学ぼう 367
3 ブラウザで Web API を使ってデータを取得してみよう 369
4-3 Search Bar、Table View を配置しよう 373
1 プロジェクトを作成しよう 373
2 Search Bar を配置しよう 373
3 Table View を配置しよう 375
4 各パーツに AutoLayout を設定しよう 377
5 Table View Cell の［Style］と［Identifier］を設定 379
4-4 キーワードを入力してお菓子データを取得しよう 380
1 画面のパーツとプログラムの関連付けをしよう 380
2 Search Bar で入力されたキーワードをデバッグエリアに出力しよう 382
3 Web API のリクエスト URL を組み立てよう 386
4 リクエストを生成して、JSON を取得しよう 390
4-5 取得したお菓子データを Table View で一覧表示してみよう 401
1 お菓子データをタプルに格納してみよう 401
2 お菓子データを Table View で一覧表示してみよう 406
4-6 ステップアップ お菓子の一覧をタップして Webページを表示してみよう ... 413
1 SFSafariViewController を使って Web ページを表示しよう 414

索引 417
Swift ビギナーズ倶楽部について 421
謝辞 422
執筆陣プロフィール 423

この本の読み方と使い方

本書が対象とする方

- プログラムを書いたことはないけれど、iPhone／iPad アプリを作ってみたい方
- iPhone アプリをよく利用していて、自分でも作ってみたいと思った方
- 中高生、大学生で iPhone アプリ開発を学んでみたい方
- シルバー世代や中高年の方で再学習を実施したい方
- 企業で入社前研修や企業導入研修での教材を検討している方

そんな iOS アプリを作ってみたい、すべての初心者が対象です。

アプリを作ることを**「開発」**するともいいます。開発といっても「難しいことをする！」と身構える必要はありません。プログラミングを楽しみながら、リラックスして読み進めてください。

本書でできるようになること

初心者の方もサンプルアプリを作ることにより、動く体験と基本の知識が身につくようになります。

この書籍を終えるころには、他の入門書やプログラミング文法書を読む力もついていると思います。そして、作りたいアプリや学習したい分野も見えてくると思いますので、ぜひ、次の書籍を購入してステップアップを目指してください。

本書の特徴

とにかく**「体験」**すること、そしてあとから**「理解」**することに重点を置いています。

本書では、プログラミングの文法説明は最小限にして、iPhone／iPad アプリを作って動かしていくことを目的として構成しています。

プログラミング文法書のように文法を理解して覚えるのではなく、どんどんアプリを作って体験していくことに比重を置いています。プログラミングがはじめての人でも楽しみながら iOS アプリが作れるという体験ができるように工夫しました。

学習が進めやすいように、学校の授業のように時限制（レッスン）で区切っています。各レッスンごとに独立したサンプルアプリが作れるように配慮していますので、制作したいサンプルアプリがあれば、途中からでも学習できます。

まったくの初心者の方は、読み飛ばさずに最初からじっくりと取り組んでみてください。少しでも経験のある方は、作りたいサンプルアプリのレッスンからはじめるのもよいでしょう。

本書の構成

Day 1（1 日目）はレッスン 5 まであり、iPhone アプリ制作の概論と開発の準備から入ります。そして、ここで「じゃんけんアプリ」「楽器アプリ」「マップ検索アプリ」の 3 つのアプリを作ります。「アプリを作って動かすことができた！」という体験を得てください。

Day 2（2 日目）はレッスン 4 まであります。サンプルアプリは「タイマーアプリ」「カメラアプリ（前半）」「カメラアプリ（後半）」「お菓子検索アプリ」を作ります。

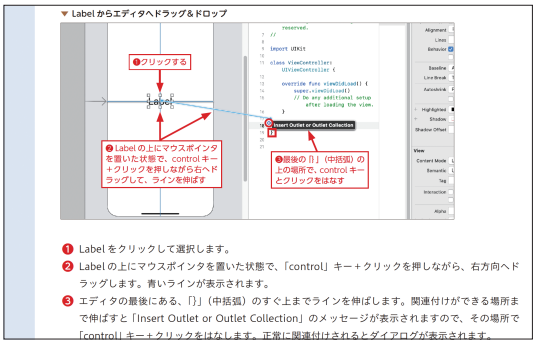
本書の読み方とページ構成

① このレッスンで学ぶこと

レッスンの中で学べることをピックアップして予測できるようにわかりやすくしています。

② Xcode 画面

Xcode（エックスコード）は、iOS アプリを視覚的に開発するための統合開発環境です。Xcode 画面は、操作が理解しやすいようにナンバリングやコメントを入れています。



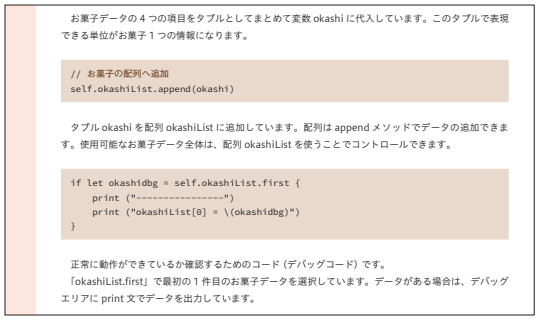
③ プログラムコードの追加

プログラムのコードを追加する場所は、わかりやすいように赤枠で囲っています。また、削除の場合は青枠で囲っています。

④ 説明用のプログラムコード

Swift のプログラムコードを掲載しています。コードは理解しやすいように、1 行から数行の塊で説明します。

※ Swift（スウィフト）は、iOS アプリを作るためのプログラミング言語です。



お菓子データの 4 つの項目をタプルとしてまとめて変数 okashi に代入しています。このタプルで表現できる単位がお菓子 1 つの情報になります。

```
// お菓子の配列へ追加
self.okashiList.append(okashi)
```

タプル okashi を配列 okashiList に追加しています。配列は append メソッドでデータの追加できます。使用可能なお菓子データ全体は、配列 okashiList を使うことでコントロールできます。

```
if let okashidbg = self.okashiList.first {
    print ("---------------")
    print ("okashiList[0] = \(okashidbg)")
}
```

正常に動作ができているか確認するためのコード（デバッグコード）です。
「okashiList.first」で最初の 1 件目のお菓子データを選択しています。データがある場合は、デバッグエリアに print 文でデータを出力しています。

⑤ Point（ポイント）、Tips（ティップス）、Column（技術コラム）

Point

Point（ポイント）は、いまの学習の中で知っておいたほうがよいことや覚えるべき箇所を記載しています。

Tips

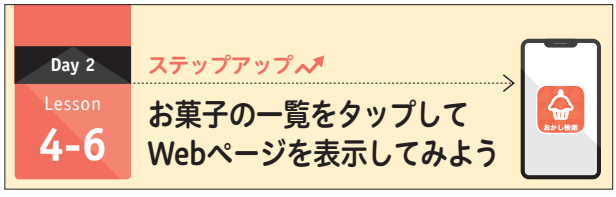
Tips（ティップス）は、Xcode の操作やコードの補足、または技術の事例を紹介します。

COLUMN

Column（技術コラム）は、本文の流れからは少しそれますが、技術の背景や、抽象的な技術の解説を記載しています。

さらに一歩踏み込んで学習できる「ステップアップ」

本書では、もう一歩進んで学習したい方のためにレッスンごとに「ステップアップ」を設けています。



サンプルアプリをさらにカスタマイズして、機能アップを行いながら学びます。「ステップアップ」が難しいと感じられた方は、最初は飛ばして学習を進めてください。2 回目以降からは「ステップアップ」も含めて学習していくことで、効果的に学ぶことができます。

本書の公式サポートサイトの紹介とサンプルアプリダウンロード

公式サポートサイトでは、本書の内容に関するサポートや、本書内で掲載されているサンプルアプリ、プログラムコードなどが提供されています。

完成したサンプルアプリの動きを確認したり、自分で打ち込んだプログラムコードの確認などでご使用ください。

本書の公式サポートサイト
https://ticklecode.com/swiftbook2019/

本書のサンプルアプリダウンロードと使い方 - 公式サポートサイト
https://ticklecode.com/swiftbook2019/sample/

ご利用の前に必ずお読みください

必要なパソコン機器

iOS アプリ開発には、Mac が必要です。

本当にはじめての方は、Windows パソコンでもアプリ開発ができると思いがちですが、iOS アプリ開発には、**Mac が必須**になります。Mac であれば、MacBook、iMac、Mac mini のどれであっても大丈夫です。

macOS を搭載したパソコンがないと、iOS アプリ開発のための環境を作ることができません。ぜひ、自分に合った Mac の購入を検討してみてください。

本書で必要な各ソフトウェアのバージョンについて

アプリ開発を行う前に、**必要なソフトウェアのバージョンを確認**して、それを満たす必要があります。バージョンはそのソフトウェアがいつ提供されたものであるのかを示す番号です。

iOS は Apple が macOS をベースに開発した iPhone、iPad、iPod touch 向けのモバイル OS です。この iOS のバージョンも確認する必要があります。

サンプルプログラムや本書で記載されているプログラムコード、画面掲載は、以下の環境に対応しています。

- macOS Mojave または、macOS Catalina
- Xcode 11.0 以上
- iOS 13.0 以上

上記のバージョンに満たない場合は、バージョンアップを行う必要があります。

バージョンアップに関しては、Apple サポートページをご確認ください。

Apple サポートページ
https://support.apple.com/ja-jp

Day 1 → Day 2

Lesson 1

はじめてのアプリを開発する前に知っておこう

iOS アプリの開発を始める前に、知っておいた方がよいことを学びます。

書籍を読み進めていく上で、最初の心構えを知っておくことで、心理的なハードルが下がり、学習が行いやすくなります。

START　Lesson 1-1 プログラミングを体験から学んでいこう

Lesson 1-2 あらかじめ挫折しそうなポイントを押さえておこう

Lesson 1-3 アプリ開発をするなら知っておこう！

GOAL　Lesson 1-4 Swift（スウィフト）を知ろう

Day 1

Lesson 1-1

プログラミングを体験から学んでいこう

このレッスンで学ぶこと

- プログラミングに対する気持ちを整理します。プログラミングを体験して学んでいくという考え方を理解します。

1 この本の使い方と前提知識

1-1 体験から学習する

あなたの心の中に、「プログラミングは頭で学ぶもの」という気持ちがあるなら、すぐに切り替えましょう！

プログラミングは「体験しながら体で学ぶ」という方法もあります。プログラミングの文法や仕組みの理解は、あとからついてきます。あなたが体験したことが糧となって、徐々に理解できるようになります。

1-2　予習や事前学習について

本書を学習するにあたって、予習のための時間は必要ありません。

はじめてプログラミングを学習する方でも、楽しみながら、段階的に知識を習得していけるように目指して構成しています。

予習で最初に知識を詰め込んでも、その知識が必要になるとは限りません。最初にアプリを作っていき、必要なときに必要な量を学習する方が効率的です。

実際に作ってみて「動いている！」という成功イメージを持つことが大切です。

1-3　基礎知識、前提知識について

本書では、事前の基礎知識、前提知識はとくに必要ありません。「動いた！」という体験をしてから、気になること、疑問に思うことを調べて、「なるほど！」と思えたときに基礎知識が身についたといえるでしょう。

「まずは体験してみる」「基礎知識はあとから学習する」ことを念頭において読み進めていくことが大切です。

まずは、自分で書いたコードが動く楽しさを実感してください。

1-4　反復学習について

最初は、新しい知識の情報量が多いため、一度学んで理解できたと感じても定着していないことが多々あります。学校の学習や、スポーツの練習と同じで、プログラミングも反復学習、反復練習を行うことで、学びが深くなります。

本書では、一度解説した内容を、あとのレッスンでも解説しています。思い出せるように簡略して解説したり、詳細に解説しているページを明記したりと、繰り返し復習していただける構成にしています。

また、最初は本書の通りに設定をしたりコードを書いたりするのも一苦労されると思います。最後までやり遂げられたら、2 周目 3 周目と繰り返していただくことをお薦めします。最初には読み漏らしていたり理解できなかったりした解説も、新たな気づきがあり理解が進みます。

1-5　オリジナルアプリについて

本書のサンプルアプリは、シンプルで理解しやすいように設計しています。これをもとに組み合わせることで本格的なアプリを作ることができます。

本書のサンプルアプリを作れたら、機能を少しずつ変更したり、読者の皆さまが実装してみたい機能を追加したりしてみましょう。

Day 1

Lesson

1-2

あらかじめ挫折しそうなポイントを押さえておこう

このレッスンで学ぶこと

- アプリ開発で挫折しそうなポイントを事前に理解します。本書で集中して学習する、エラーや警告に対する考え方、Xcode について学びます。

1 学習ポイントを押さえよう

ポイント①：まずは一冊の本に取り組む

入門書をたくさん購入することで満足してしまいがちですが、最初は、1 つの書籍を最後まで学習することが大切です。まずは、じっくりと本書だけに取り組んでみてください。あせることはありません。すべてが理解できなくても、気にせず前に進んでください。

そして、本書を最後まで終えたら、また最初から取り組むことをお薦めします。最初はよくわからなかった箇所も、理解が進んでいることが実感できるはずです。

本書を読み終えるころには、次の新しい入門書や文法書も読み進めていくことができるようになります。

ポイント②：アプリ開発をする前の準備

アプリ開発を行う上で、事前の準備が必要になります。事前の設定ではトラブルも多く、ここで諦めてしまう方も多いと思います。

本書では、Day 1 Lesson 2 でアプリ開発の準備にも十分な時間を割いています。事前の準備が終われば、いよいよアプリを作ります！

ポイント③：アプリ開発で表示される警告やエラー

アプリの開発を進めていく過程で、**「警告」**や**「エラー」**と言われるものに遭遇することがあります。

最初は、警告やエラーの意味がよくわからず戸惑うことと思いますが、安心してください。警告やエラーはあなたを責めているのではなく、**よりよい方法を教えてくれている**のです。

警告やエラーに遭遇したときには、「アドバイスしてくれてありがとう！」というぐらいの気持ちを持って、開発に取り組みましょう。エラーメッセージを読んでわからない場合でも、メッセージをそのままインターネットで検索すると対処法がわかる場合も多くあります。調べながら解決をすることを覚えてください。

ポイント④：まずは、Xcode を体験して慣れていこう

アプリ開発には Xcode という統合開発環境を利用します。

最初は、Xcode の操作がよくわからなくてなかなか思うように作業が進みません。でも、Xcode でのメニュー配置が理解できて、目的の作業をするための手順がわかるようになると、効率的に制作できるようになります。

Xcode も理解しようとするよりも体験して、慣れていくことが大切です。繰り返し操作を体験し、どんどん慣れてください。

Day 1

Lesson 1-3

アプリ開発をするなら知っておこう！

～WWDC、手数料、課金方法～

このレッスンで学ぶこと

- **アプリ開発をする上で知っておきたい、WWDC の概要、ダウンロード数などの数値、手数料、課金について概要をつかみます。**

1 WWDC での公開情報をもとに考察してみよう

1-1 WWDC とは

WWDC（Worldwide Developers Conference）は、iPhone の開発元である Apple が毎年開催している、開発者向けのイベントです。

WWDC で Apple の新製品や新機能が発表されたり、市場動向や開発者への支払額が公開されたりします。そのため、アプリ開発者にとっては、とても関心が高いイベントです。

WWDC - Apple Developer
https://developer.apple.com/wwdc19/

WWDC 2019 は 6 月 3 日～7 日にカリフォルニア州サンノゼで開催されました。

WWDC は Apple の製品・サービスが発表されるイベントですが、参加するためのチケットは 1,599 ドル（WWDC 2019 チケット価格）で販売されていて、1 ドル 108 円換算で約 17 万 2 千円と、その人気の高さがわかります。

WWDC では、表のような製品・サービスが発表されています。

▼ WWDC での製品発表

発表年	製品
2010 年	iPhone 4、iPad
2012 年	MacBook Pro（Retina ディスプレイ搭載）
2013 年	Mac Pro（円筒形デザイン）
2014 年	Swift
2015 年	Swift 2.0、Swift オープンソース化
2016 年	Swift 3.0、Sirikit、iMessage Apps
2017 年	Swift 4.0、iPad Pro、HomePod
2018 年	ARKit 2.0、CreateML
2019 年	Sign in with Apple、SwiftUI

1-2　Apple がアプリ開発者に支払った金額

アプリ開発者は平均でどのぐらいの収入を得ているのでしょうか？ WWDC 2019 では開発者への支払い総額や登録アプリ数の公式発表がありませんでした。そのため、昨年の WWDC 2018 での公式発表の数値から試算してみたいと思います。

WWDC 2018 での公式発表によると、2008～2018 年までの 10 年間で Apple が開発者に支払った金額は累計 1000 億ドルを超えています。支払った金額を 1 ドル 110 円換算すると、約 11 兆円にもなります。また、登録アプリ数は約 220 万本（2017 年 1 月現在）であるとされるため、この 11 兆円を 220 万本のアプリ数で割ると、1 つのアプリの平均収入は約 500 万円になります。

2008 年からアプリを 1 つリリースしていたとすると、年間平均で 50 万円、月平均で約 4 万円の収入になります。そして、登録アプリ数の 220 万本には無料アプリも含まれていますので、平均は約 4 万円以上になると思われます。

1-3　Apple の手数料

Apple Store で販売した売上のうちの 30% を Apple が手数料として差し引きます。これが Apple の収益になります。残りの 70% が開発者への支払額です。支払いは、指定した銀行口座に振り込まれます。

たとえば、120 円のアプリであれば、84 円が振り込まれることになります。

1-4　ユーザーへの課金方法

iPhone アプリを App Store に公開してユーザーに課金する方法は 3 種類あります。

▼ ユーザーへの課金方法

課金方法	説明
有料ダウンロード	アプリをダウンロードするときに課金する方法。ユーザーは 1 度ダウンロードすれば、その後のアップデートなどは無料で実行可能。アプリを削除して再びダウンロードするときも無料
広告	アプリの中に広告を表示させて収益を得る方法。いろいろな会社がアプリ向けの広告配信サービスを提供している
In App Purchase（アプリ内課金）	アプリ内で課金する方法。有料もしくは無料でアプリを提供し、そのアプリ内でアイテムや機能追加を販売し、収益を得る。アプリ内課金には、次の 4 つの方法がある ①消耗型：アプリの実行に伴い消費されていく。消費アイテムなど ②非消耗型：1 度購入するとユーザーのすべてのデバイスで使用可。書籍など ③自動更新購読：期間を決めて販売、自動で更新。新聞、雑誌など ④非更新購読：期間を決めて販売、自動で更新されない。 1ヶ月購読など

Day 1

Lesson 1-4

Swift（スウィフト）を知ろう

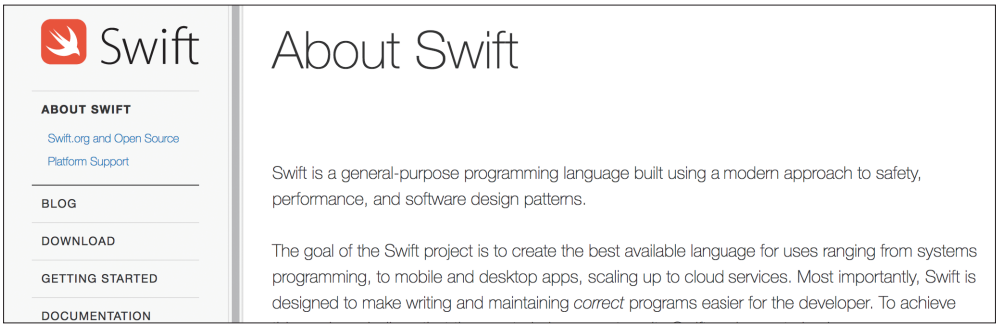
このレッスンで学ぶこと

- iOS アプリ開発で使う、Swift（スウィフト）というプログラミング言語について概要を学びましょう。

1 Swift の特徴を押さえよう

みなさんが、iOS アプリを作るために使うプログラミング言語が Swift（スウィフト）と呼ばれるものです。Swift は 2014 年の WWDC ではじめて発表されました。

▼ Swift.org



Swift.org - Welcome to Swift.org
https://swift.org/

Apple の公式サイトでは、Swift とは「誰もが圧倒的に優れたアプリケーションを作れる、パワフルなオープンソースの言語」として紹介されています。

Swift を使うことでアプリ開発者はより安全で、より信頼性の高いコードを書くことができ、時間を節約しながら、より豊かなアプリを作ることができます。

特徴①：Swift は高速である

Swift は日本語訳で「迅速」という意味で、鳥の「あまつばめ」を示す意味でも使われます。 Swift の迅速さを強調するために、ロゴマークにも「あまつばめ」が採用されています。

Swift は他のプログラミング言語と比較して、検索アルゴリズム（データを探し出す方法）が高速だと言われています。

特徴②：Swift はモダンである

モダン（modern）とは「現代風」という意味です。 Swift では、他のプログラミングでも採用されている新しい機能を、積極的に取り入れています。

特徴③：Swift は安全である

Swift はプログラミングの記述ミスやバグ（不具合）が起こりにくい仕組みを採用しています。

特徴④：たくさんの Apple 製品で動くアプリが作れる

Swift で作れるアプリは iPhone や iPad 上で動くアプリだけではありません。Mac、Apple Watch などたくさんの Apple 製品で動くアプリを作ることができます。

特徴⑤：Swift はオープンソースである

オープンソース（Open Source）とは、プログラムソースを一般に公開して、誰もが使ってよいとする考え方です。

オープンソースであれば、一般人も Swift の改良に参加することができますので、ネット上では日々意見交換され、さらなる改良・進化していくことが期待されています。

Tips

Swift が発表される前は、「Objective-C」（オブジェクティブシー）というプログラミング言語を使って、アプリ開発を行っていました。

Objective-C は 1980 年代から開発がはじまり、機能拡張を経て現在も使用されています。Objective-C は、記述が長くなり複雑であったり、プログラムを書いていく効率があまりよくなかったりと、初心者には難しい言語です。

Swift では、学びやすいような工夫がされていて、新しい考え方や機能を積極的に取り入れています。

Swift もバージョンアップを重ねて、十分に開発が行える言語として成長しています。また、過去の資産を有効活用できるように、Objective-C で作成されたプログラムを Swift から呼び出すこともできます。

本書では、これからの開発言語である「Swift」でプログラムを記述します。

Tips

WWDC 2019 にて、新しい開発プラットフォーム SwiftUI が発表されました。

SwiftUI では、宣言型シンタックスにより、プログラムコードとデザインを完全に同期しながらユーザーインターフェースを作成することができます。

宣言型シンタックスは、テキストフィールドを作成すると書いてから、フォントサイズ、カラー設定などを順にプログラムコードとして追加していくことで、ユーザーインターフェースを作成します。

SwiftUI は、発表されたばかりですが、今後も機能追加・修正がすすむことで開発環境の主力ツールになると思われます。

本書では、SwiftUI を使用せずに、従来どおりの Storyboard で解説をします。

SwiftUI

SwiftUI については、次の Apple 公式ページをご参照ください。

SwiftUI - Apple Developer
https://developer.apple.com/jp/xcode/swiftui/

Day 1 → Day 2

Lesson 2

アプリ開発の環境を整えて、Xcode の使い方を学ぼう

START
Lesson 2-1
開発するために必要な準備をしよう

Lesson 2-2
Apple ID を取得しよう

Lesson 2-3
Xcode をインストールしよう

Lesson 2-4
Xcode を起動して、プロジェクトを作成しよう

Lesson 2-5
Xcode をより使いやすくするための設定をしよう

GOAL
Lesson 2-6
「Hello Swift!」と表示してみよう

iOS アプリを開発するために、必要なものを学びましょう。

最初に、まだ Apple ID を取得していない方のために、Apple ID の基礎知識と取得方法についても解説します。

Day 1

Lesson 2-1

開発するために必要な準備をしよう

このレッスンで学ぶこと

- アプリ開発をするために用意しなければならないことを学びます。
- Mac や Apple ID、Xcode の概要について理解します。

1 アプリ開発に必要な 3 つのものを準備しよう

iOS アプリを開発するためには、次の 3 つが必要になります。

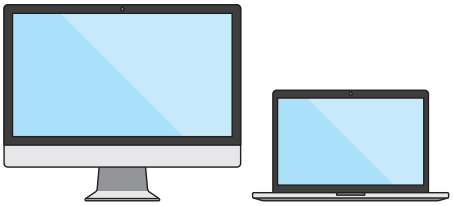
- **Mac**
- **Apple ID アカウント**
- **Xcode**

上記の 3 つに加え、開発したアプリを全世界の人たちに利用してもらうには、別途、**「Apple Developer Program」**への登録も必要です。

このレッスンでは開発したアプリを、iPhone に転送（実機転送）して利用するところまでをゴールにしています。

では、必要な 3 つのものを確認しましょう。

1-1 Mac



2015 年 12 月に、Swift は誰でも利用・改変できるオープンソースとしてソースコードが公開されました。そのため、将来 Mac 以外のパソコンでも iOS アプリ開発ができるようになる可能性もありますが、本書を執筆している時点では Mac が必要になります。現時点では Mac を準備してください。

1-2 Apple ID

▼ App Store と iTunes

iOS アプリを開発するためには、**Apple ID** の作成が必要です。Apple ID は、Apple が提供しているオンラインサービスを利用するために必要なアカウントです。

オンラインストアの「iTunes Store」では、音楽・映画やオーディオブックを購入できます。また、「App Store」というサービスでは、iPhone や iPad、Mac で利用できるアプリがダウンロードできます。さらに、有料で販売されているアプリも購入することができます。iPhone や iPad をお持ちの方は、すでに Apple ID を利用していると思います。

それぞれのサービスを利用するのに必要な Apple ID ですが、Xcode のダウンロードでも必要になるため、事前に作成する必要があります。

1-3 Xcode（エックスコード）

▼ Xcode

Apple ID が作成できたら、「App Store」で、iOS アプリ開発に必要な Xcode をインストールします。

Xcode は、Mac、iPhone、iPad、Apple Watch、Apple TV 向けのアプリを開発できる環境を提供してくれます。Xcode を利用して、画面やコードの作成、デバッグ、App Store へのアプリの提出ができます。

一般にこのようなツールは、**統合開発環境**、もしくは **IDE（Integrated Development Environment）** と呼ばれています。IDE は、ソフトウェアを効率よく開発できるように、さまざまな機能を提供してくれている開発ツールです。Swift や Objective-C を使って開発ができる Xcode 以外にも、さまざまな言語（Java、Ruby、PHP など）で、有料無料問わずにたくさんの IDE があります。

Xcode 以外にも iOS アプリ開発ができる IDE はありますが、一般的に多く利用されているのは Xcode です。

Day 1

Lesson

2-2 Apple ID を取得しよう

このレッスンで学ぶこと

- Apple ID の作成手順を学び、Apple ID を取得します。
- Apple ID の調べ方や、メールアドレスの変更方法も確認します。

1 Apple ID をすでに取得されている方

すでに Apple ID を取得済みの方は、その Apple ID が 1 つあれば、Apple のすべてのサービスが利用できます。App Store から Xcode をダウンロードできますので、このレッスンは読み飛ばしていただいて大丈夫です。

1-1 Apple ID を持っているか不明な方

取得しているご自身の Apple ID がよくわからないときがあります。その際は、下記の URL から調べることができます。

Apple ID を忘れた場合 - Apple サポート
https://support.apple.com/ja-jp/HT201354

1-2 Apple ID のメールアドレスを変更したい方

すでにお持ちの Apple ID のメールアドレスを変更できます。メールアドレスを変更しても、いままで利用してきた Apple ID をそのまま使い続けることができます。下記の URL に、メールアドレスの変更方法が記載されています。

Apple ID を変更する - Apple サポート
https://support.apple.com/ja-jp/HT202667

2 Apple ID をまだ取得されていない方

次の手順に沿って、一緒に作成しましょう。

2-1 Apple ID のサイトにアクセス

Apple ID のサイトにアクセスします。

Apple ID を管理 - Apple
https://appleid.apple.com/jp

［Apple ID を作成］をクリックします。

▼ Apple ID ログイン画面

2-2 Apple ID を作成

各項目を入力しましょう。個人の情報と、個人確認をするときのセキュリティ質問などを設定します。

赤枠の入力は、**文字認証**と呼ばれている仕組みです。左に表示されている画像の英数字を読み取って、右の入力欄に入力します。読みづらい英数字の場合は、「新規コード」をクリックして、読める英数字を表示させましょう。

これは、悪意のあるプログラムから自動的にアクセスされ不正に利用されないようにするためです。人間にしか読めない読みにくい画像で英数字を読んで入力させることで、悪意のあるプログラムからは利用できないようにしています。

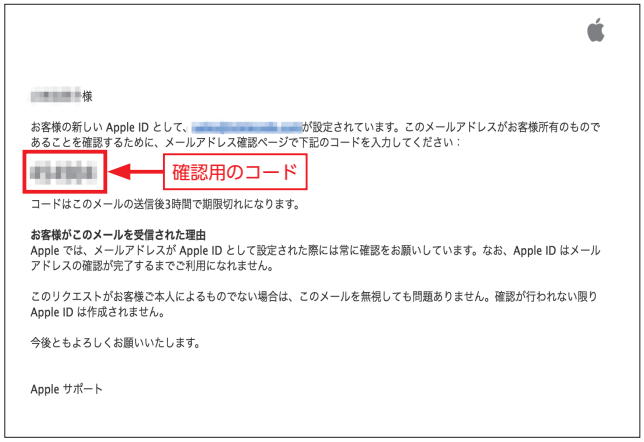
よく利用される認証方法ですが、Apple の場合は音声で読みあげるサポートもしてくれています。「音声サポート」をクリックすると、表示されている英数字を読みあげてくれます。

［次に進む］をクリックすると、確認用のコードを入力する画面が表示されます。

▼ Apple ID を作成

入力したメールアドレスに確認用のコードが送信されます。メールを確認すると、確認用のコードが記載されています。

▼ 確認コードの記載があるメールが届く



確認用のコードを右の画面に入力します。

▼ Apple ID 作成時のコード入力

これで、Apple ID の作成は完了です。Xcode のダウンロードが行える準備が整いました。
続いて、Xcode のダウンロードとインストールを行います。

Point

Apple ID は、App Store、iTunes Store、iCloud をはじめとした Apple サービスへのアクセスに使う個人アカウントです。Apple すべてのサービスで使う連絡先、支払情報などが含まれています。
アプリ開発においても、Xcode から iPhone へアプリを転送する際に Apple ID が必要になりますし、アプリを App Store へ申請登録する際にも必要となります。
とても重要な ID とパスワードですので、大切に管理しておきましょう。

Xcode をインストールしよう

このレッスンで学ぶこと

- App Store から、Xcode のインストール方法を学びます。
- Xcode でアプリ開発が行える環境を整えます。

1 Xcode をダウンロードしよう

1-1 Xcode をダウンロードする前に

本書では、Xcode を利用して開発を進めますので、最初に App Store から Xcode のダウンロードを行います。

インターネット回線の速さによって変わりますが、Xcode のダウンロードには数十分かかります。MacBook などの場合は、ダウンロードとインストール実行中にバッテリーが切れないように気をつけてください。

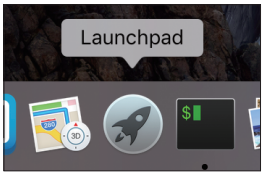
また、Xcode 自体のファイルサイズが数 GB あり大きいので、インストールする Mac のストレージ容量も充分かどうか確認してください。

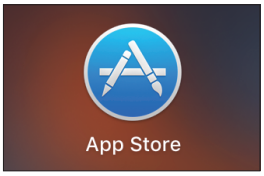
1-2 App Store からのインストール

App Store から Xcode をインストールしましょう。

Mac の Dock にある「Launchpad」アイコンをクリックして起動します。Launchpad の中にある App Store を起動します。

▼ Launchpad から App Store を起動





App Store が起動するので、画面の左上の検索ボックスに「Xcode」と入力して、「enter」キーを押します。

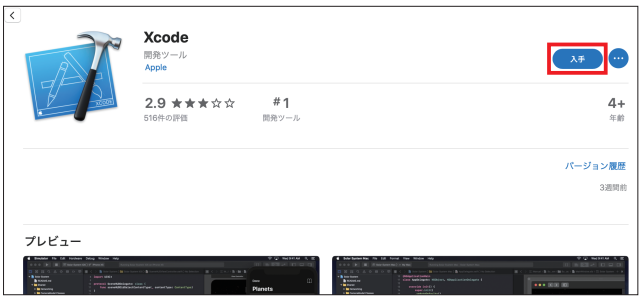
▼ App Store から Xcode を検索

検索結果から「Xcode」をクリックします。

▼ 検索結果から Xcode を選択

Xcode の詳細ページが表示されます。［入手］ボタンをクリックしてください。

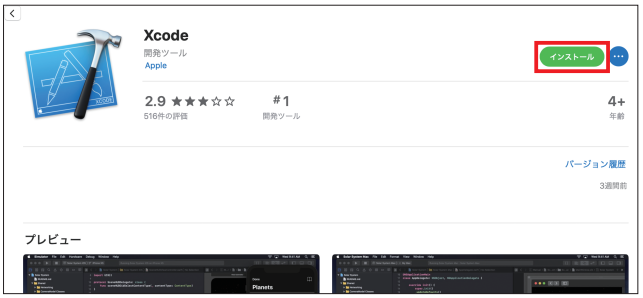
▼ Xcode の入手



［インストール］をクリックすると、Xcode のダウンロードとインストールがはじまります。

Xcode のファイルサイズはとても大きいため、ダウンロードには時間がかかることがあります。本書を読み進めながら、気長にお待ちください。

▼ App をインストール



Xcode を起動して、プロジェクトを作成しよう

このレッスンで学ぶこと

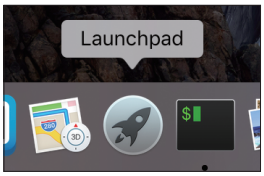
- Xcode を起動して、プロジェクトの作成を行います。
- Xcode の基本的な画面構成について学習します。

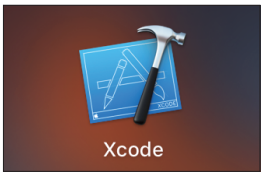
1 Xcode を起動しよう

1-1 Xcode を起動

Xcode の起動方法はいくつかありますが、今回は Launchpad から、Xcode のアイコンをクリックして起動しましょう。

▼ Launchpad から Xcode を起動

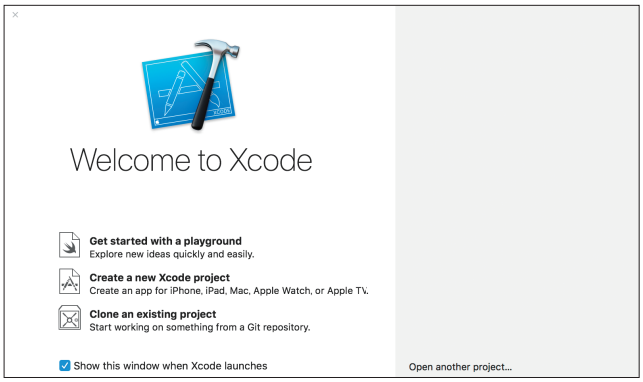




1-2 Xcode の起動画面

「Welcome to Xcode」と書かれた画面が表示されます。本書では Xcode を起動すると、毎回、この画面が表示されます。画面には、「Get started with a playground」、「Create a new Xcode project」、「Clone an existing project」の 3 つのメニューが表示されています。次のページより、メニューを 1 つずつ確認しましょう。

▼ Xcode の起動画面



- **Get started with a playground**
  Xcode 6 から、Playground（プレイグラウンド）という機能が追加されました。
  Playground とは「遊び場」という意味で、Swift のコードを書きながら結果がすぐに確認できます。Swift を書きながら、動きを確認するのに適しています。
- **Create a new Xcode project**
  新規のプロジェクトを作成します。本書では、すべてこのメニューから新規にプロジェクトを作成して、アプリを開発します。
- **Clone an existing project**
  バージョン管理システムを使ってプロジェクトを作成する方法です。バージョン管理とは、ファイルの変更履歴を管理してくれるシステムです。高度な機能になるので、本書では解説の対象外になります。

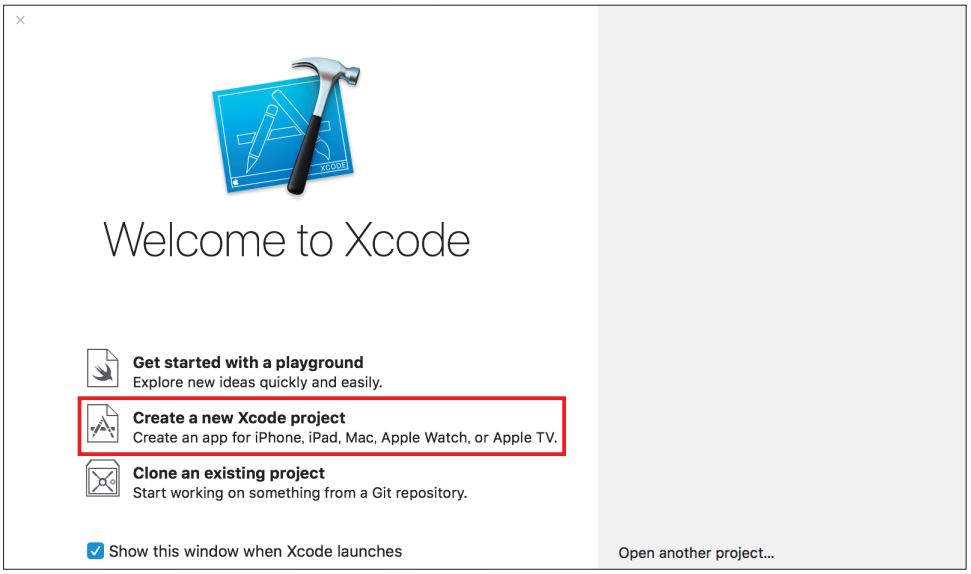
2　プロジェクトを作成しよう

2-1　プロジェクトを作成

［Create a new Xcode project］をクリックして、はじめてのプロジェクトを作成してみましょう。

プロジェクトはアプリ開発を束ねるものです。1 つのプロジェクトで 1 つのアプリを開発します。プロジェクトの中に、アプリの画面設計や画像ファイル、プログラムが入っています。

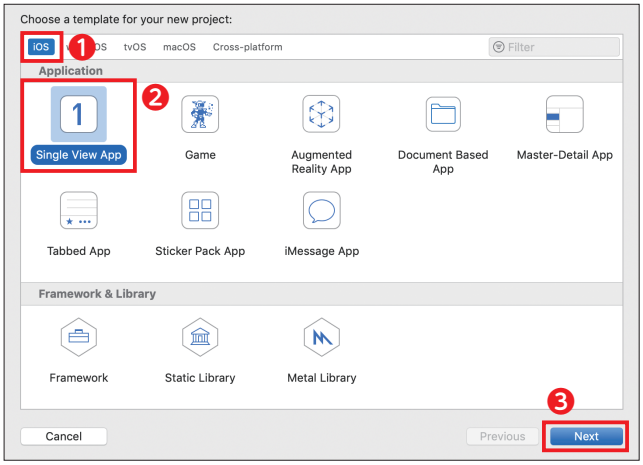
▼ Xcode プロジェクトの作成



［Choose a template for your new project:］ダイアログが表示されます。これは、プロジェクトのテンプレート（雛形）を選ぶ画面です。この画面でプロジェクト、テンプレートを選ぶことができます。次の手順に沿って選びましょう。

❶ 画面の左上部の［iOS］を選択します。
❷［Single View App］を選択します。
❸［Next］を選択しましょう。

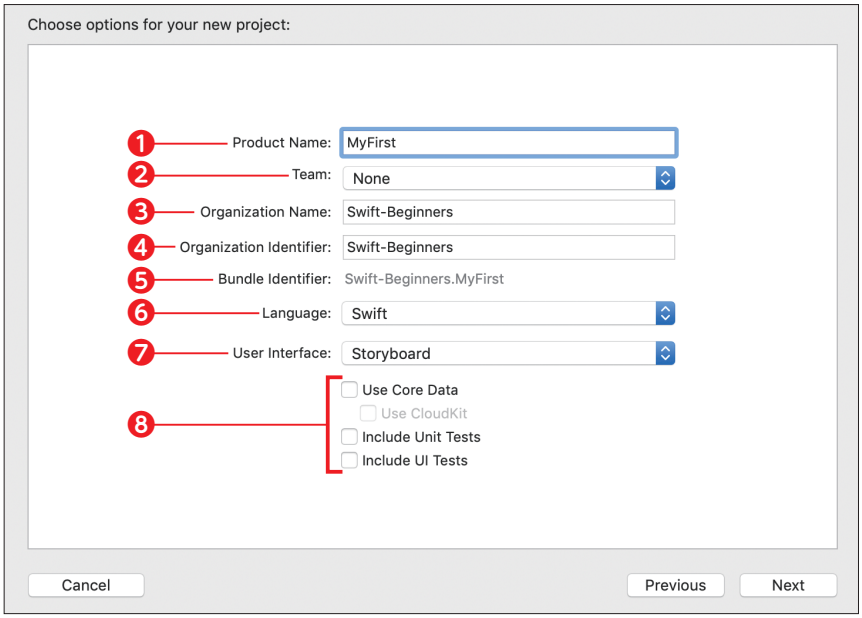
▼ プロジェクトテンプレートの選択



2-2　プロジェクトの情報を入力

［Choose options for your new project］と記載された画面が表示されます。

▼ プロジェクトの新規作成



この画面では、プロジェクトの情報を設定します。設定する情報を一つ一つ確認しましょう。

❶ Product Name（製品名）

プロジェクトの名前です。アプリの初期段階での名前になります。プロジェクトの名前には好きなものが入力できます。ここでは、「MyFirst」と入力します。

❷ Team（チーム）

Xcode に登録済みの「Apple ID」を選択します。シミュレータで動作確認をする場合には設定の必要はないですが、実機転送を行う際には設定が必要です。本レッスンの最後で実機転送を説明していますので、いまは設定をしなくても大丈夫です。

❸ Organization Name（組織名）

会社名、組織名を入力します。本書では「Swift-Beginners」と入力していますが、自由に入力して大丈夫です。

❹ Organization Identifier（組織識別名）

［Organization Identifier］は、❶［Product Name］と組み合わされて、❺［Bundle Identifier］に設定されます。

［Bundle Identifier］は世界中のアプリの中でユニーク（一意）にする必要があり、重複が許されません。

［Organization Identifier］は、アプリが所属するグループ（組織や個人）を識別します。

識別として利用されるのは、ドメイン表記を逆にした記述（逆ドメイン）です。ドメインが「swift-beginners.jp」の場合は、「jp.swift-beginners」と入力します。

ドメインを持っていない方は、メールアドレスを逆から入力する方法もあります。なぜなら、メールアドレスも世界に一つだけのものなので、ユニークになるからです。

ここでは、個人のメールアドレスを入力してみましょう。そのときに、メールアドレスは逆から入力し、@（アットマーク）は .（ドット）に置き換えます。

例えば、「swift-beginners@example.jp」の場合は、「jp.example.swift-beginners」と入力します。

Point

本書では、［Organization Identifier］を、「Swift-Beginners」と入力しています。ですが、学習の際には、別の ID にしてください。たとえば、ご自身のメールアドレスのように、重複しにくい ID を入力してください。

❺ Bundle Identifier

この項目は、入力することはできません。［Product Name］と［Organization Identifier］を組み合わせて自動で生成されます。今回の設定では、「Swift-Beginners.MyFirst」と表記されています。

❻ Language（プログラミング言語）

アプリを作るプログラミング言語を選択します。「Swift」と「Objective-C」が選べます。「Swift」を選択します。

❼ User Interface

開発環境を選択します。「SwiftUI」と「Storyboard」が選べます。「Storyboard」を選択します。

❽ Use Core Data、Include Unit Tests、Include UI Tests

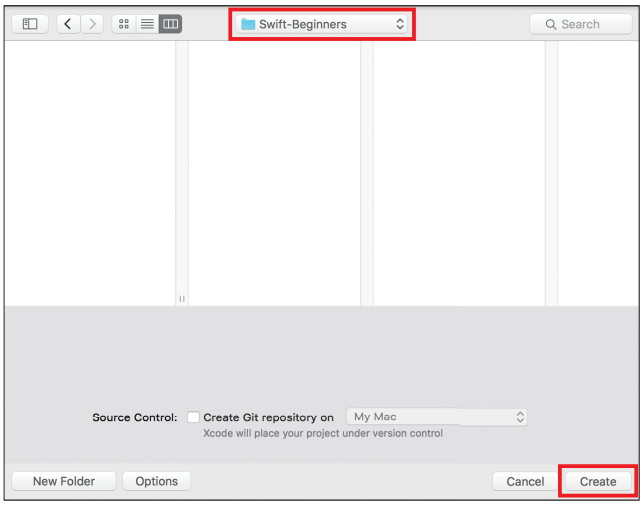
すべてチェックを外します。これらの項目は、高度な開発で必要になるデータベースやテストコードの指定です。

2-3　プロジェクトを保存

プロジェクトの保存場所を聞かれるので、適当な場所を選択します。本書では、「Swift-Beginners」というフォルダを作成して、そこにプロジェクトを保存します。

［Create］ボタンをクリックします。

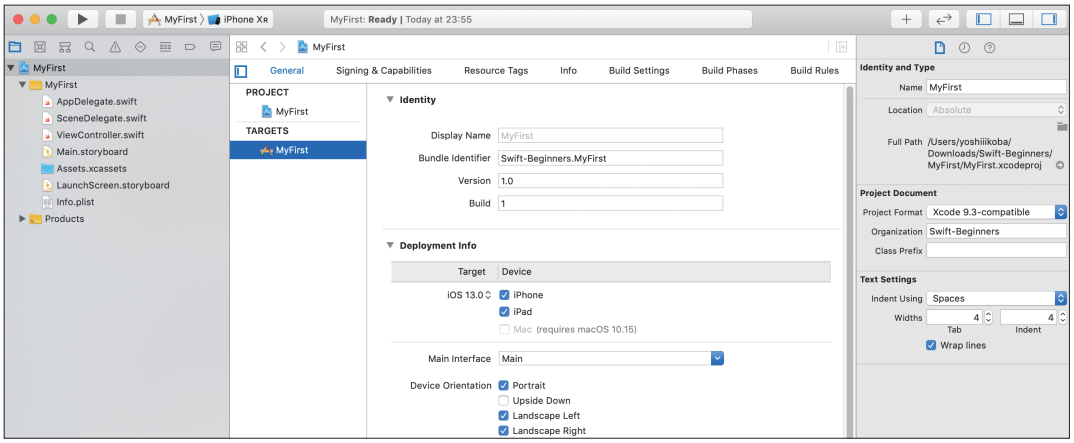
▼ プロジェクトの保存場所



プロジェクトが作成され、Xcode プロジェクト画面が表示されます。

この Xcode プロジェクト画面でアプリの画面を作成し、プログラムを書きます。

▼ Xcode プロジェクト画面



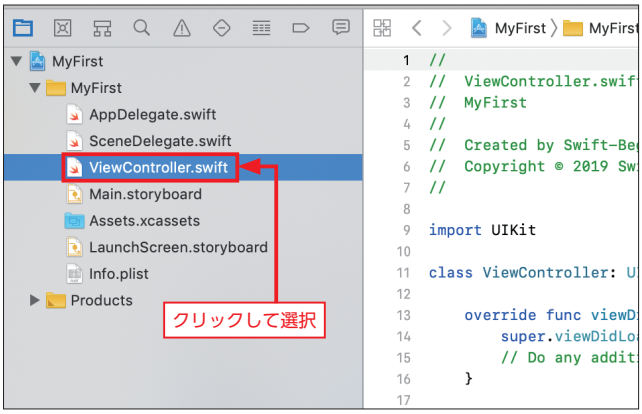
3 Xcode 画面のナビゲーションを理解しよう

最初は、Xcode プロジェクト画面が表示されています。

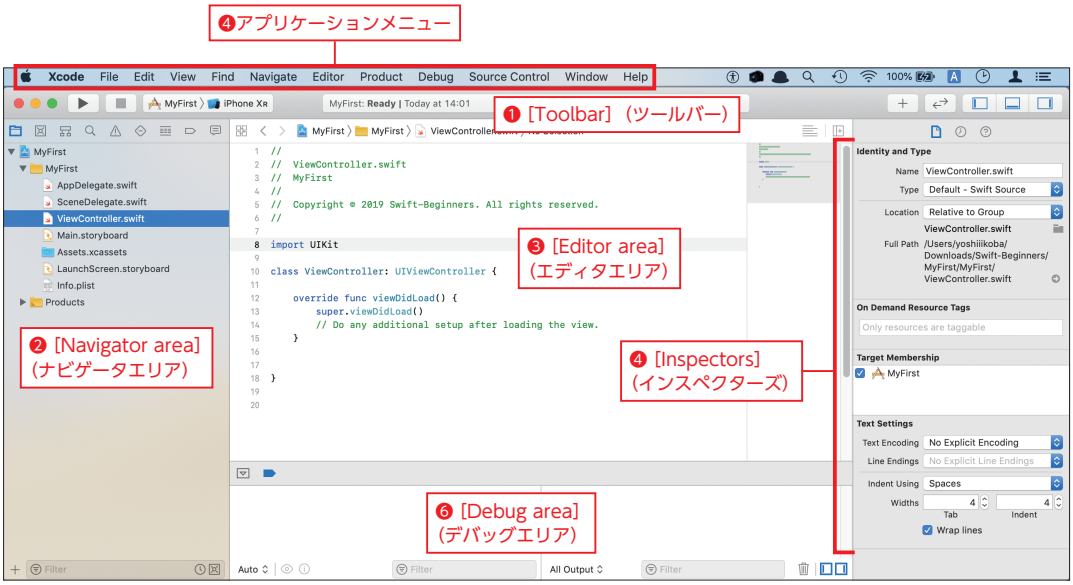
まずは、[ViewController.swift] を選択してみてください。ここからは、Xcode のナビゲーションや画面配置を確認します。

Xcode のプロジェクトエリアでファイルを選択するときはシングルクリックです。ダブルクリックするとファイルが別の画面や別のタブとして新しく開かれます。

▼ [ViewController.swift] の選択



▼ Xcode 画面構成



Xcode はいくつかの画面エリアに分かれています。

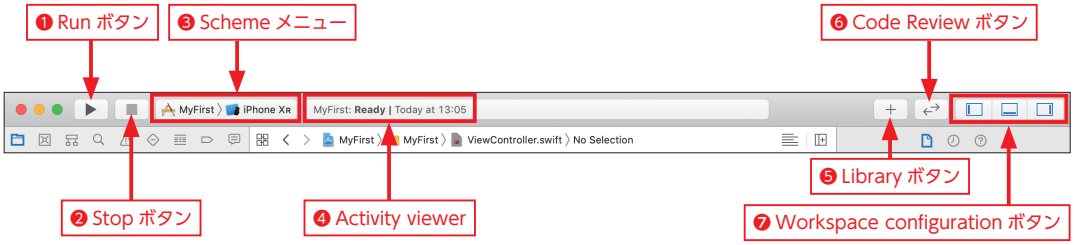
▼ Xcode の画面エリア

画面のエリア	役割
❶［Toolbar］（ツールバー）	画面上部のメニューでアプリの実行を行え、画面を切り替えることができます
❷［Navigator area］（ナビゲータエリア）	左サイドのナビゲータエリアで、ファイルを選択します
❸［Editor area］（エディタエリア）	アプリ画面の作成とプログラムを書くことができます
❹［Inspectors］（インスペクターズ）	エディタエリアで、表示または選択されているパーツ、ソースファイルに対応する情報を、表示、編集できます
❺［アプリケーションメニュー］	Mac のメニューバーの中に、使用中のアプリケーションのメニューバーが表示されます
❻［Debug area］（デバッグエリア）	シミュレータ実行時などに状況が出力される場所。自分でプログラムを書いて出力できるので、動作確認やテストを行うときにも使用します

まずは、ツールバーの機能から確認します。

3-1　［Toolbar］（ツールバー）

▼［Toolbar］の構成



❶ ▶［Run］ボタン（実行ボタン）

プログラムを、ビルド・実行できます。

ところで、**「ビルド」**とは何なのでしょう？「ビルド」は、私たちが書いたコードをコンピューターが理解できる最終的な実行ファイルとして作成することです。プログラミング関連の書籍にはよく出てくる用語なので、覚えておきましょう。

❷ ■［Stop］ボタン（ストップボタン）

実行中のプログラムを停止します。

❸［Scheme］メニュー（スキームメニュー）

プロジェクトで実行する、「iOS シミュレータ」を選択できます。

iOS シミュレータは、Xcode で利用できる仮想デバイスです。実際に iPhone、iPad などを Mac につながなくても、Xcode 上でアプリの動作を確認できます。たとえば iPhone XR を持っていなくて

も、iOS シミュレータを起動することで、iPhone XR ではどのようにアプリが表示されるのかの確認が行えます。

❹ [Activity viewer]（アクティビティビューワー）

実行中のプログラムの状態や、ビルドの進捗状況が表示されます。

❺ ＋ [Library] ボタン（ライブラリボタン）

画面の UI パーツ（部品）や、プログラムのコードスニペット（テンプレート）が収められています。

❻ [Code Review] ボタン（コードレビューボタン）

プログラムコードの変更履歴を記録・追跡できる Git を活用する際に使用します。本書では使用致しません。

❼ [Workspace configuration] ボタン

左のアイコンから① [Navigator area]、② [Debug area]、③ [Inspectors] のそれぞれを表示・非表示に切り替えることができます。

▼ [Workspace configuration] ボタン

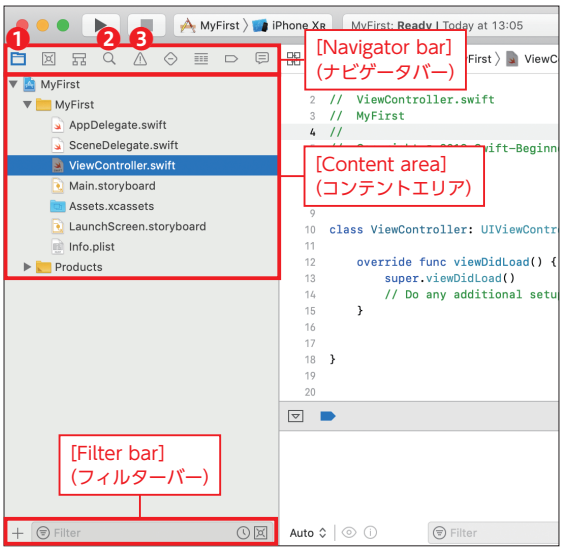
3-2　[Navigator area]（ナビゲータエリア）

[Navigator area] は Xcode の左端に表示されているナビゲーションで、[Navigator bar] [Content area] [Filter bar] で構成されています。

それぞれの役割を確認します。

- **[Navigator bar]（ナビゲータバー）**

  [Navigator bar] には、各種ナビゲータが用意されています。

  よく利用するナビゲータに絞って、説明します。

▼ [Navigator area] プロジェクトのファイル表示エリア