

第27回 日本医療情報学会春季学術大会

チュートリアル6

二次利用ユーザーのためのデータ抽出、変換、登録（ETL）入門  
－ ETLの理解がデータ活用の近道－

## PHRデータ利活用事始め

（Apple Healthcareからのデータ取得と医療情報との連携）

内村 祐之（東京医科歯科大学）

## 本日の内容

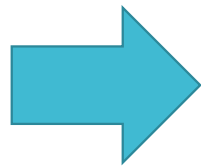
1. 利用可能なPHRデータについて
2. Apple Healthcareについて
3. Apple Healthcareからデータを取得するiPhoneアプリについて
4. iPhoneアプリから別のサーバへのPHRデータ送信について
5. サーバでのPHRデータ処理について
6. ヘルスケア情報と医療情報との連携について

## 利用可能な PHRデータに ついて

- iPhone「ヘルスケア」 - Apple Watchと連携
- Androidアプリ「ヘルスコネクト」 - Fitbitと連携
- quick 非接触型体温計 → AppleヘルスケアApp
- omron 血圧計、体重体組成計、活動量計、体温計、パルスオキシメータ、 → AppleヘルスケアApp
- 医療情報との連携も始まっています
  - 札幌医科大学と富士通、ヘルスケア領域のデータポータビリティ実現に向けて、個人の健康データの活用推進に合意
    - <https://pr.fujitsu.com/jp/news/2023/01/16.html>
- 他にもたくさんあります

## Apple Healthcare について

- Appleのヘルスケア機能はすべて、あなたのデータを安全に保護し、プライバシーを守るように作られています。デバイス上のあなたの健康データは暗号化され、自分のパスコード、Touch ID、Face IDのいずれかを使わない限りアクセスできません。iCloudと同期されている健康データは、iCloudとの通信中も保存時も暗号化されます。  
(Appleサイトより)



自分の健康情報は自分の端末からのみアクセス可能。  
サードパーティアプリからアクセスする際には項目毎に許可設定が必要。

【Apple Healthcareについて】

iPhoneの健康  
データは公開  
されている？



健康データが公開されて  
いないか、心配ですか？  
プライバシー。これがiPhone。

【Apple Healthcareについて】

# iPhoneのヘル スケア



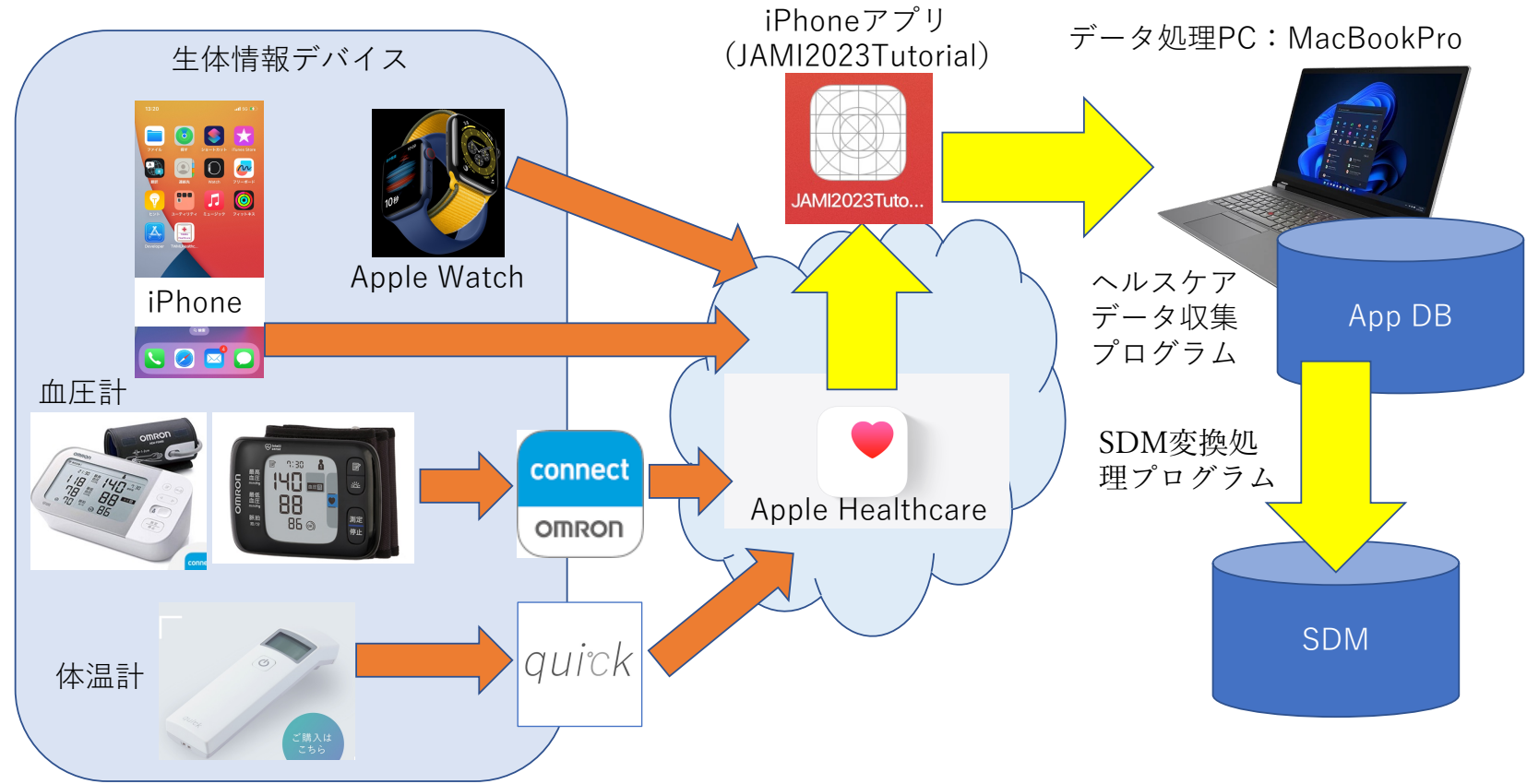
【Apple Healthcareについて】

# Apple Watch

	高心拍数と低心拍数の通知	不規則な心拍の通知	心電図アプリケーション	心肺機能レベルの低下の通知	体に取り込まれた酸素のレベル	転倒検出
センサー	光学式心拍センサー	光学式心拍センサー	電気心拍センサー	光学式心拍センサーと位置情報	光学式心拍センサー	加速度センサーとジャイロスコープ
Apple Watch Series 3	✓	✓	✗	✓	✗	✗
Apple Watch SE	✓	✓	✗	✓	✗	✓
Apple Watch Series 4, 5	✓	✓	✓	✓	✗	✓
Apple Watch Series 6以降	✓	✓	✓	✓	✓	✓
Apple Watch Ultra	✓	✓	✓	✓	✓	✓

# 今回のチュートリアル説明用システム

## 生体情報デバイスデータ取得・分析システム 概要





【iPhoneアプリについて】

## アプリ開発に必要なもの

- **iPhoneの実機**

(シミュレータでは制限があります)

- **開発用PC**

(Windowsでの開発もできなくはないですが、基本的にはMacになります。メモリは16GB以上が必要です。)

- **開発環境**

(標準的なものはXcodeになります)

- **実機とPCを繋げるケーブル**

(iPhoneに付属しているもので大丈夫ですが、別のケーブルを使う場合にはMFi認証Lightningケーブルでないといけないとダメです)

【iPhoneアプリについて】

## Xcodeについて

- AppleがiOSアプリ開発のために無償提供しているソフトウェアがXcodeです。
- iOSアプリの構築、実機検証、デバックという、アプリ開発を一元化した統合開発環境を持っており、iOSアプリ開発のためにXcodeは欠かせないソフトウェアとなっています。
- XcodeはApp Storeから入手可能です。

【iPhoneアプリについて】

# Apple Developer Program

- Apple Developer ProgramはアプリをApp Storeに公開可能になるApple公式のメンバーシッププログラムです。
- 年額\$99の費用がかかります。
- Apple Developer Programはさまざまな開発ツールやアプリの分析、最新テクノロジーが使用可能になります。
- 開発したiOSアプリはApp Storeを通じて販売・ダウンロードされますが、アプリ開発者が作成したiOSアプリをApp Storeで公開するためには、Apple Developer Programに登録を行う必要があります。
- Apple Developer Programを解約すると公開したアプリのダウンロードができなくなってしまうので、継続してダウンロードしてもらうには登録の継続が必須となります。

【iPhoneアプリについて】

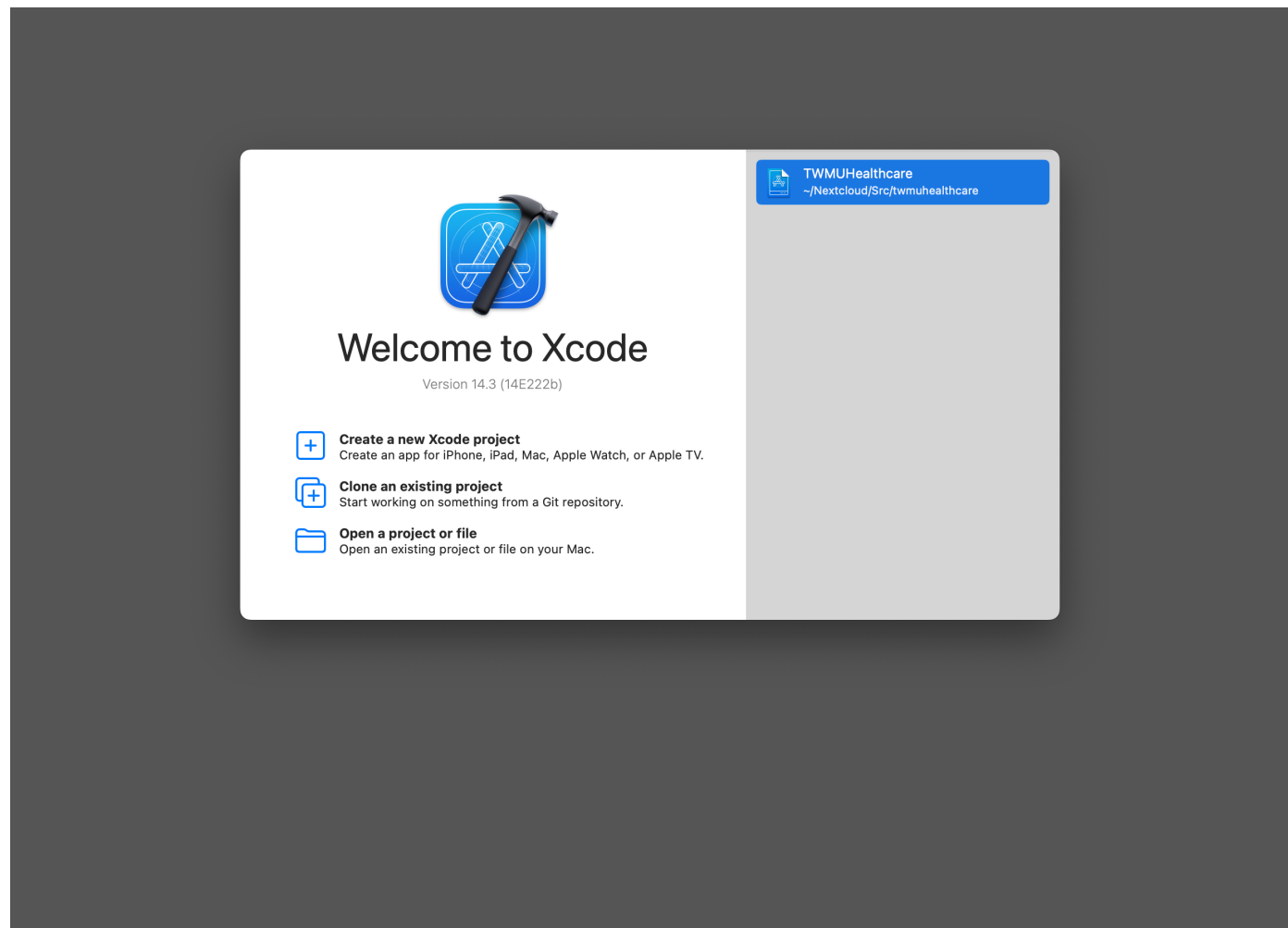
## SwiftUI開発について

- iPhoneネイティブアプリを開発するには、MacでXcodeというツールを使って開発を行うのが一般的です。
- XcodeではSwift言語を使って開発を行いますが、Swift開発は従来のStoryboardを使ったSwift開発と2019年から始まったSwiftUIによる開発の2パターンがあります。
- Storyboard開発とは、例えばオブジェクトをドラッグ&ドロップで配置してレイアウトを作っていく開発方法で、SwiftUIは画面のレイアウトを全てコードで実装するというのが特徴です。
- SwiftUIは2019年にリリースされた開発方法なので、ネット上に情報が少なかったり、古い本には書かれていないので注意が必要です。

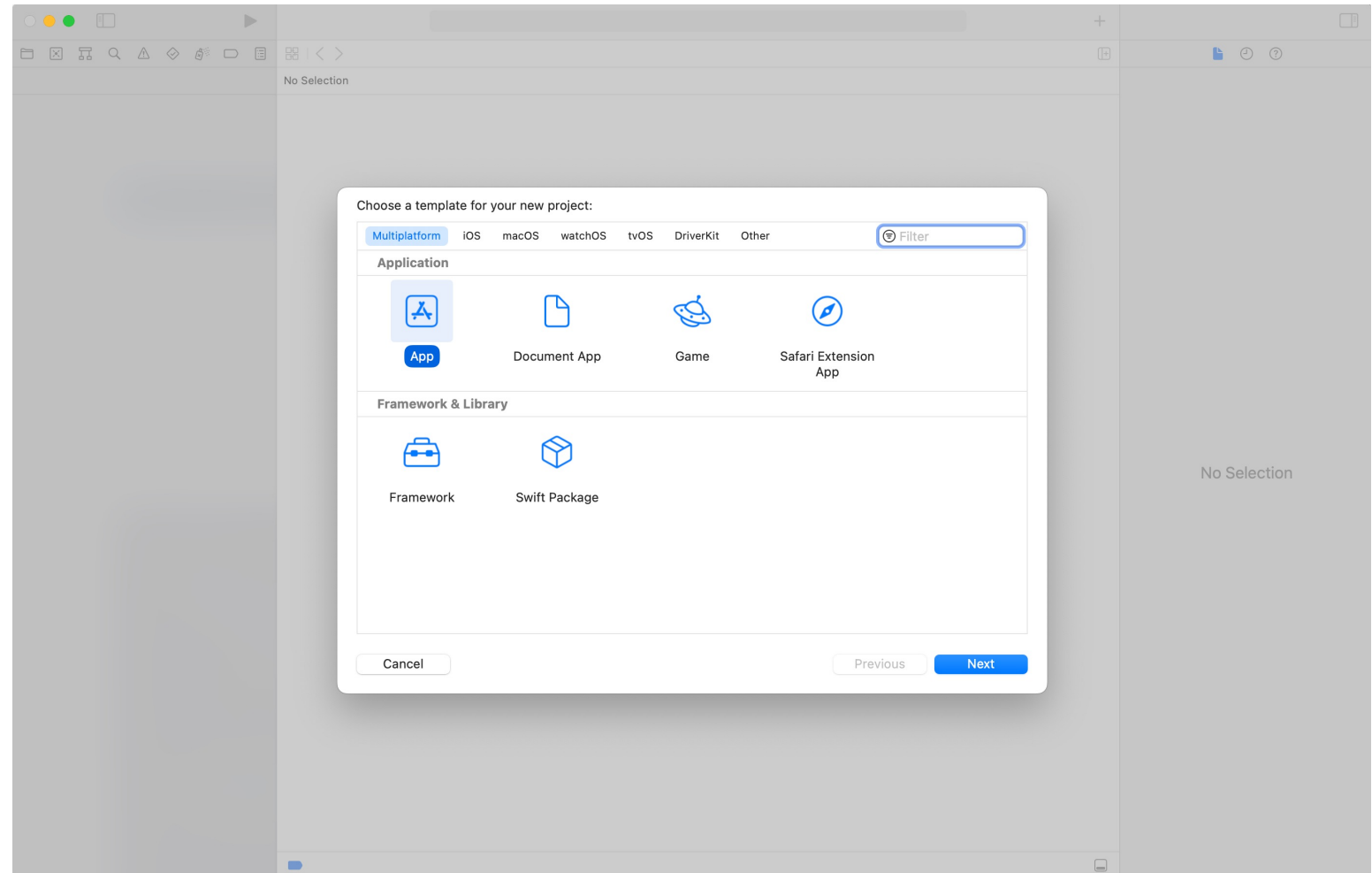
## 1. Xcodeで新規プロジェクトを作成します

【iPhoneアプリについて】

# アプリ開発の 流れ 1



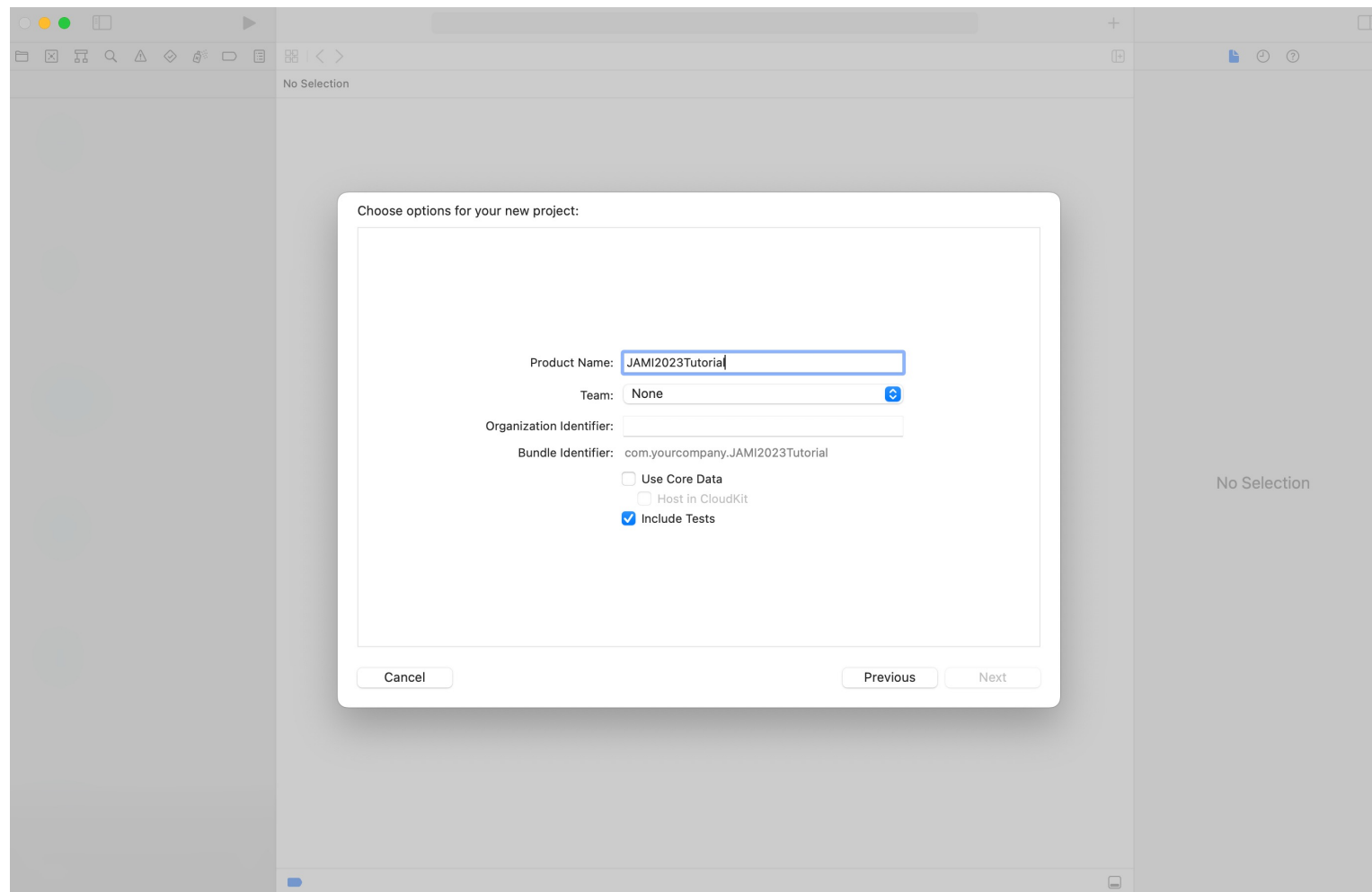
2. アプリケーションから「App」を選択します。



【iPhoneアプリについて】

# アプリ開発の流れ2

### 3. プロジェクト名を入力します (JAMI2023Tutorial)



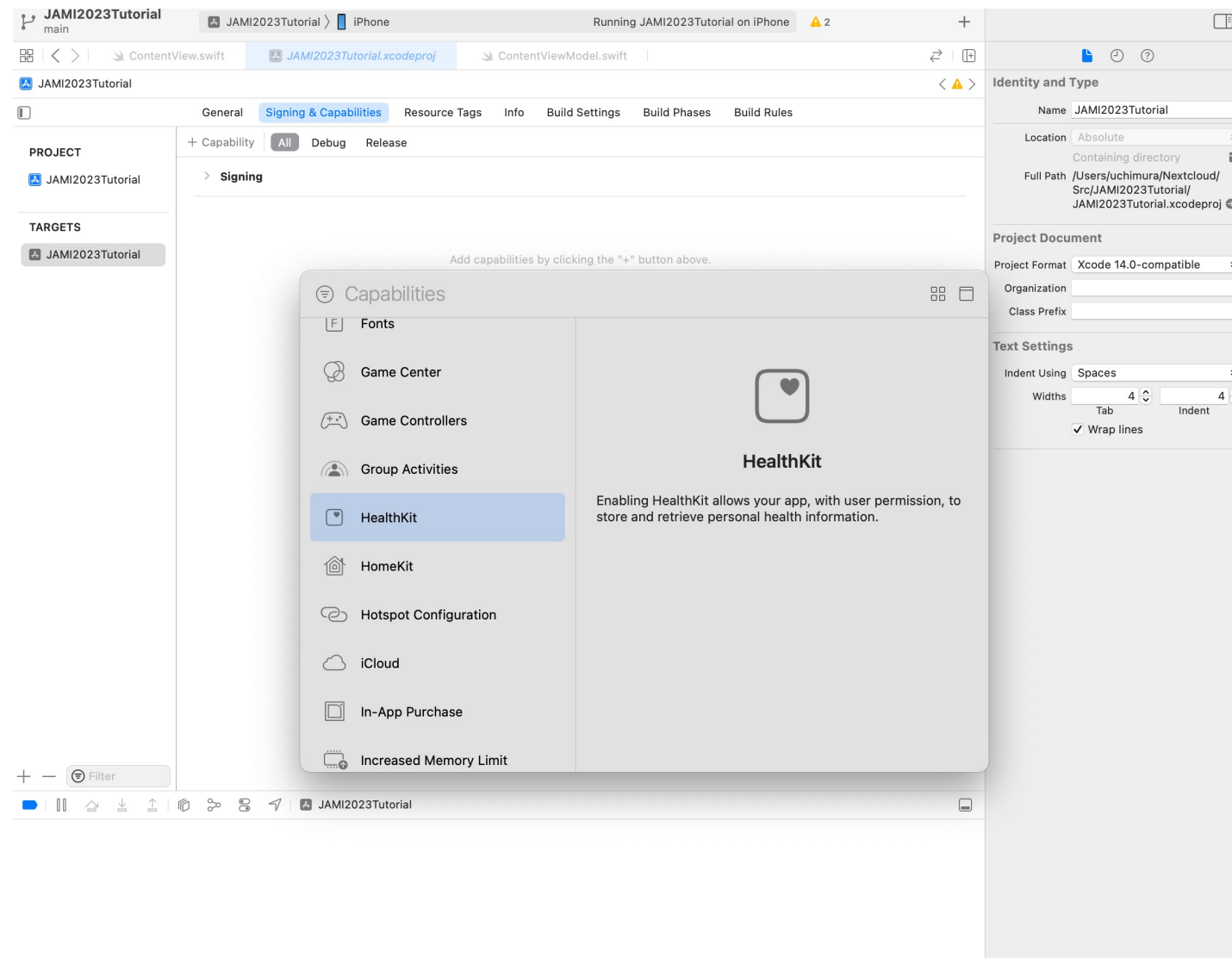
【iPhoneアプリについて】

# アプリ開発の流れ 3

【iPhoneアプリについて】

# アプリ開発の流れ5

## 5. Healthkitが使えるようにします①





## 5. Healthkitが使えるようにします②

The screenshot shows the Xcode interface for a project named 'JAMI2023Tutorial'. The 'Info' tab is selected, displaying the 'Custom macOS Application Target Properties' section. The table below lists various properties for the target.

Key	Type	Value
> Supported interface orientations (iPhone)	Array	(3 items)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
Bundle name	String	\$(PRODUCT_BUNDLE_NAME)
InfoDictionary version	String	6.0
Bundle version	String	\$(CURRENT_PROJECT_VERSION)
Executable file	String	\$(EXECUTABLE_FILE_PATH)
Privacy - Health Share Usage Description	String	read health data
Privacy - Health Update Usage Description	String	write health data
Bundle OS Type code	String	\$(PRODUCT_BUNDLE_OS_TYPE_CODE)
Default localization	String	\$(LOCALIZATION_ROOT)
> Supported interface orientations (iPad)	Array	(4 items)
Bundle version string (short)	String	\$(MARKETING_VERSION)

Below the table, several sections are listed with zero items:

- > Document Types (0)
- > Exported Type Identifiers (0)
- > Imported Type Identifiers (0)
- > URL Types (0)
- > Services (0)

【iPhoneアプリについて】

# アプリ開発の 流れ5

【iPhoneアプリについて】

# アプリ開発の 流れ 6

## 6. 項目毎にヘルスケアの利用許可を得ます①

HKHealthStoreのrequestAuthorization  
メソッドを呼び出してヘルスケアの  
利用許可を得る必要があります。  
以下は「体温」の例です。

```
let readTypes = Set([
    HKQuantityType.quantityType(forIdentifier:
        HKQuantityTypeIdentifier.bodyTemperature)!
])
```



【iPhoneアプリについて】

# アプリ開発の流れ6

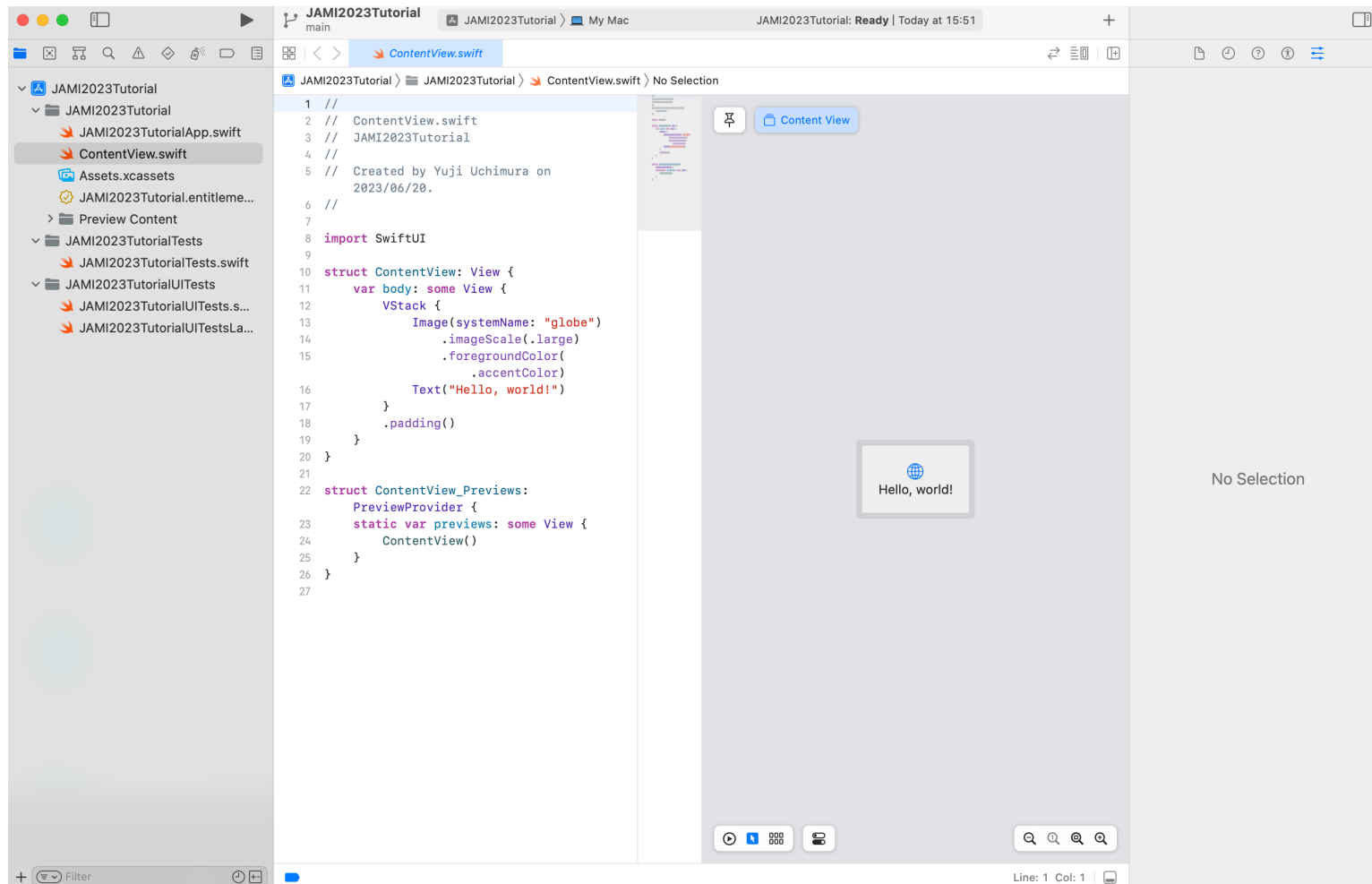
## 6. 項目毎にヘルスケアの利用許可を得ます②



【iPhoneアプリについて】

# アプリ開発の 流れ 7

## 7. あとは普通にコードを書きます



【iPhoneアプリについて】

# アプリ開発の流れ 8

## 8. 画面も作っていきます

The screenshot displays the Xcode IDE for a project named "JAMI2023Tutorial". The left sidebar shows the project structure with files like "ContentView.swift" selected. The central editor shows the following Swift code:

```
5 // Created by Yuji Uchimura on
6 // 2023/06/20.
7
8 import SwiftUI
9
10 struct ContentView: View {
11     var body: some View {
12         VStack {
13             Image(systemName: "globe")
14                 .imageScale(.large)
15                 .foregroundColor(.accentColor)
16             Text("Hello, world!")
17         }
18         .padding()
19     }
20 }
21
22 struct ContentView_Previews:
23     PreviewProvider {
24     static var previews: some View {
25         ContentView()
26     }
27 }
```

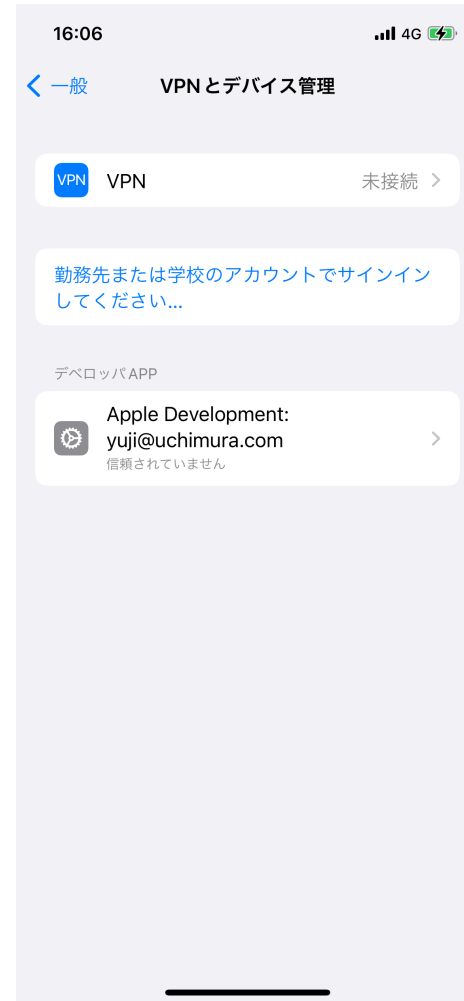
The right-hand side shows a live preview of an iPhone simulator. The screen displays a globe icon and the text "Hello, world!". The status bar at the bottom of the simulator shows the time as 15:52. The console at the bottom of the Xcode window displays a warning message:

```
2023-06-20 15:52:16.963312+0900 JAMI2023Tutorial[6919:1644862] [Window]
Warning: Window SwiftUI.AppKitWindow 0x145e44470 ordered front from a
non-active application and may order beneath the active application's
windows.
```

【iPhoneアプリについて】

# アプリ開発の流れ 9

9. 実機開発の場合、初回起動時に「信頼されていないデベロッパ」ダイアログが表示されるので設定します。



「設定」 - 「一般」 - 「VPNとデバイス管理」

で許可設定します。

【iPhoneアプリについて】

# アプリ開発の流れ 9

9. 実機開発の場合、初回起動時に「信頼されていないデベロッパ」ダイアログが表示されるので設定します。



「信頼されていません」から「検証済み」に変わったことを確認します。

## 標準データモデルを用いた PHRデータと 医療情報との 連携

- 東京医科歯科大学の病院情報システムのデータはDWHへ蓄積されている。
- 医系・歯系の電子カルテ、医事会計システム、主要部門システムの約50システムのデータを蓄積している。
- DWHは標準データモデルの1つである「SDM (Semantic Data Modeling)」を採用した「JUST DWH」を採用した。
- PHRデータをSDM (SDM\_VITAL\_RAW) へ変換を行う事で、医療情報との連携が可能となる。
- 今回はiPhoneアプリからREST-APIでPHR情報を取得し、DWHデータとの連携を見据えてサーバにPHRデータを集約するサンプルプログラムを作成した。



# iPhoneアプリ で取得した PHRデータを サーバへ送信 する

## 8. iPhoneアプリからREST-APIでサーバへ転送します

```
func sendServer(uuid: String, typeid: Int, type: String, startdate: String,
device: String, personalId: String, serverUrl: String) {
    // API call
    guard let url = URL(string: serverUrl) else { return }

    let data: [String: Any] = [
        "uuid": uuid,
        "personalid": personalId,
        "typeid": typeid,
        "type": type,
        "startdate": startdate,
        "enddate": enddate,
        "result": result,
        "unit": unit,
        "source": source,
        "device": device
    ]

    guard let httpBody = try? JSONSerialization.data(withJSONObject: data, c

var request = URLRequest(url: url)
request.timeoutInterval = 200*1000
request.httpMethod = "POST"
request.addValue("application/json", forHTTPHeaderField: "content-type")
request.httpBody = httpBody

let numberOfTimes = 100

let task = URLSession.shared.retryDataTask(withRequest: request, times:
DispatchQueue.main.async {
    self.count += 1
}
guard let response = response as? HTTPURLResponse,
(200...299).contains(response.statusCode) else {
    return
}
}
```

ヘルスケアから取得したPHR  
情報をまとめています。

REST-APIでPOSTします

iPhoneアプリ  
で取得した  
PHRデータを  
サーバへ送信  
する

8. iPhoneアプリ内で各種設定できるように設定画面を作っています

### サーバ情報

### 連携項目の設定



# iPhoneアプリ で取得した PHRデータを DBへ格納す る①

## 8. REST-APIでiPhoneからPHRデータ を取得してDBへ格納します。

- サーバプログラムはPythonの軽量WebアプリケーションフレームワークであるFlaskを使用して開発しました。
- DBはPostgreSQLを使っています。

```
@api.route('/results', methods=['POST'])
def post_result():
    # jsonリクエストから値取得
    payload = request.json
    uuid = payload.get('uuid')
    personalid = payload.get('personalid')
    typeid = payload.get('typeid')
    type = payload.get('type')
    startdate = payload.get('startdate')
    enddate = payload.get('startdate')
    result = payload.get('result')
    unit = payload.get('unit')
    source = payload.get('source')
    device = payload.get('device')

    # レコードの登録 新規作成したオブジェクトをaddしてcommit
    result = Result(uuid, personalid, typeid, type, startdate, enddate, result, unit, source, device)
    db.session.add(result)
    db.session.commit()
```

# iPhoneアプリ で取得した PHRデータを DBへ格納す る②

## 8. PHRデータを標準データモデル（SDM）でDBへ格納 します。（SDM\_VITAL\_RAWの抜粋）

	record_id [PK] character	group_id character		
1	VTLRAW,12345678,00000002,4,20230310100355	... VTLRAW,12345678,00000002,4,20230310100355		
2	VTLRAW,12345678,00000002,4,20230310140045	... VTLRAW,12345678,00000002,4,20230310140045		
3	VTLRAW,12345678,00000002,4,20230310144531	... VTLRAW,12345678,00000002,4,20230310144531		
4	VTLRAW,12345678,00000002,4,20230315181128	... VTLRAW,12345678,00000002,4,20230315181128		
	measurement_item_code character	measurement_item character varying	measurement_device character varying	measurement_bodypart character varying
5	VTLRAW,12345678,00000002,4,2023	4	取り込まれた酸素のレベル	N/A
6	VTLRAW,12345678,00000002,4,2023	4	取り込まれた酸素のレベル	N/A
		4	取り込まれた酸素のレベル	N/A
		4	取り込まれた酸素のレベル	N/A
		4	取り込まれた酸素のレベル	N/A
		4	取り込まれた酸素のレベル	N/A
		4	取り込まれた酸素のレベル	N/A

measurement_method character varying	measurement_result character varying	measurement_unit character varying	measurement_value real	measurement_previous_value real	ratio_previous_value real
Airec0さんのApple Watch	97	%	97	0	0
Airec0さんのApple Watch	97	%	97	0	0
Airec0さんのApple Watch	96	%	96	0	0
Airec0さんのApple Watch	100	%	100	0	0
Airec0さんのApple Watch	100	%	100	0	0
Airec0さんのApple Watch	96	%	96	0	0

## 参考文献 参考サイト

- Apple HealthKit関連
  - <https://developer.apple.com/jp/design/human-interface-guidelines/healthkit>
  - <https://developer.apple.com/documentation/healthkit>
- 札幌医科大学と富士通、ヘルスケア領域のデータポータビリティ実現に向けて、個人の健康データの活用推進に合意
  - <https://pr.fujitsu.com/jp/news/2023/01/16.html>
- Apple 「Apple Watchを使用した最大酸素摂取量による心肺機能の推定」
  - [https://www.apple.com/jp/healthcare/docs/site/Using\\_Apple\\_Watch\\_to\\_Estimate\\_Cardio\\_Fitness\\_with\\_VO2\\_max\\_JP.pdf](https://www.apple.com/jp/healthcare/docs/site/Using_Apple_Watch_to_Estimate_Cardio_Fitness_with_VO2_max_JP.pdf)
- Apple 「Apple Watchを使用した6分間の歩行距離の推定」
  - [https://www.apple.com/jp/healthcare/docs/site/Using\\_Apple\\_Watch\\_to\\_Estimate\\_Six-Minute\\_Walk\\_Distance\\_JP.pdf](https://www.apple.com/jp/healthcare/docs/site/Using_Apple_Watch_to_Estimate_Six-Minute_Walk_Distance_JP.pdf)
- Apple 「iPhoneの歩行指標を使用した歩行の質の測定」
  - [https://www.apple.com/jp/healthcare/docs/site/Measuring\\_Walking\\_Quality\\_Through\\_iPhone\\_Mobility\\_Metrics\\_JP.pdf](https://www.apple.com/jp/healthcare/docs/site/Measuring_Walking_Quality_Through_iPhone_Mobility_Metrics_JP.pdf)

ご清聴ありがとうございました