Client-Side Options (Part 2)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - Some client-side options
    - Mobile clients
    - Android clients
    - React Native clients
    - Appendix: iOS clients

Agenda

- **Mobile clients**
- Android client
- React Native client
- (Appendix) iOS client

Mobile Clients

- Suppose you want your app to run on a mobile device…

- So far:
  - Mobile web apps
- Now:
  - Native mobile apps

Mobile Clients

- Which option (mobile web app vs native mobile app) is right for you?
  - **Step 1**: Consider whether you have an option!

Mobile Clients

- See https://progressier.com/what-web-can-do-today

Mobile Clients

- Which option (mobile web app vs. native mobile app) is right for you?
  - **Step 1**: Consider whether you have an option!
  - **Step 2**: Consider the broader context…

Mobile Clients

Desirable Factor	Mobile Web App	Native Mobile App
Easy discoverability	✔	
Easy launchability		✔
Native look & feel		✔
Good performance (speed)		✔
Easy installation	✔	
Low development cost *	✔	
Low maintenance cost *	✔	
Few content restrictions, easy approval process, low/no fees	✔	

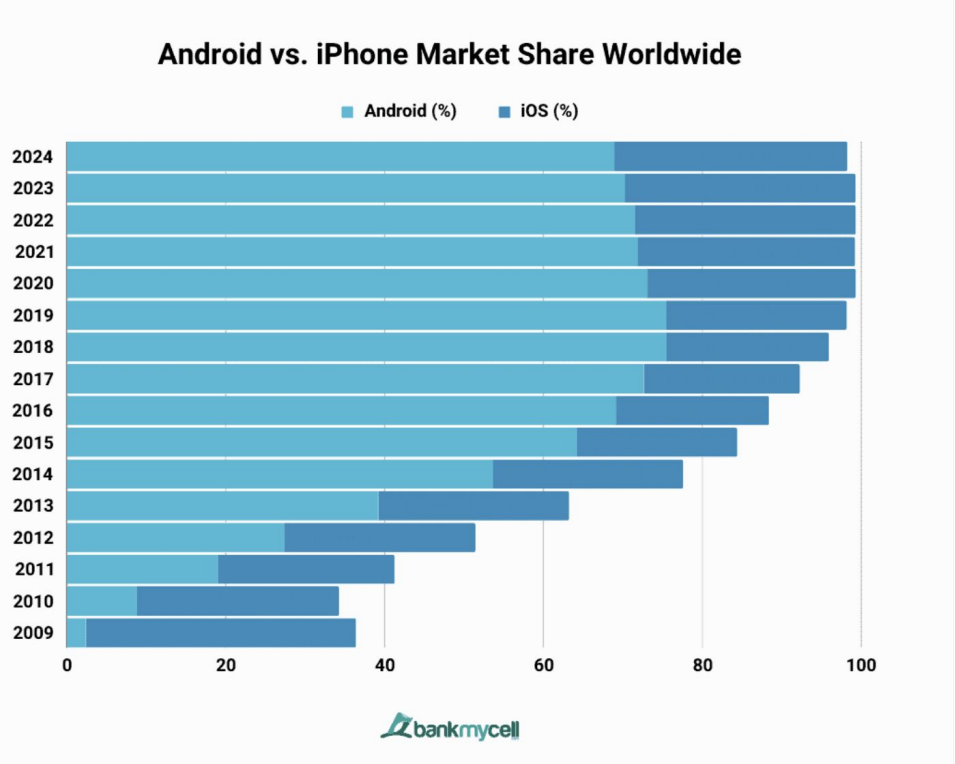
* May need multiple native mobile apps

https://www.nngroup.com/articles/mobile-native-apps/

Mobile Clients

Android vs. iOS?

Mobile operating systems' market share worldwide



In 2024:
Android: 69.88%
iOS: 29.39%

https://www.bankmycell.com/blog/android-vs-apple-market-share/

Mobile Clients

- **Problem**:
  - A native **Android** app works only on **Android** devices
    - Expressed using Java or Kotlin
  - A native **iOS** app works only on **iOS** devices
    - Expressed using Swift or Objective-C
- **Solution**:
  - Cross-platform frameworks

Mobile Clients

Framework	Development Language	Developer	Kinds of Apps
Flutter	Dart	Google	Android, iOS, web, desktop
React Native	JavaScript	Facebook, now Meta Platforms	Android, iOS
Kotlin Multiplatform	Kotlin	Jet Brains	Android, iOS, web, desktop, server-side
Ionic	JavaScript	Drifty Co.	Android, iOS, Windows, web, desktop
.NET Multi-Platform App UI	C#	Microsoft	Android, iOS, macOS, Windows
NativeScript	JavaScript	Telerik Progress Software	Android, iOS

Each runs on an emulator, which runs on the Android or iOS device
React Native and **NativeScript** are not **native**!

Agenda

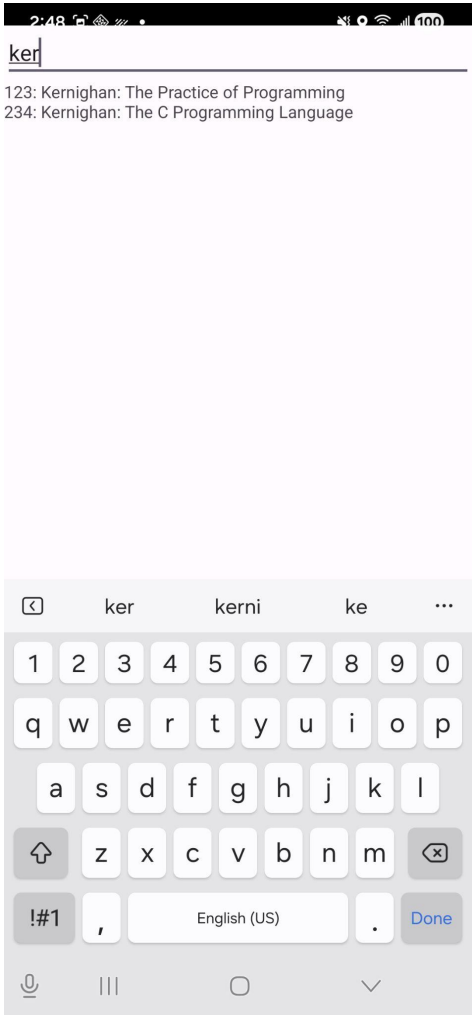
- Mobile clients
- **Android client**
- React Native client
- (Appendix) iOS client

Android Client

- Preliminary
  - Deploy PennyJson to https://pennyjson.onrender.com
    - So PennyAndroid client can access its searchresults route

Android Client

The goal:



An Android client for the PennyJson server

Android Client

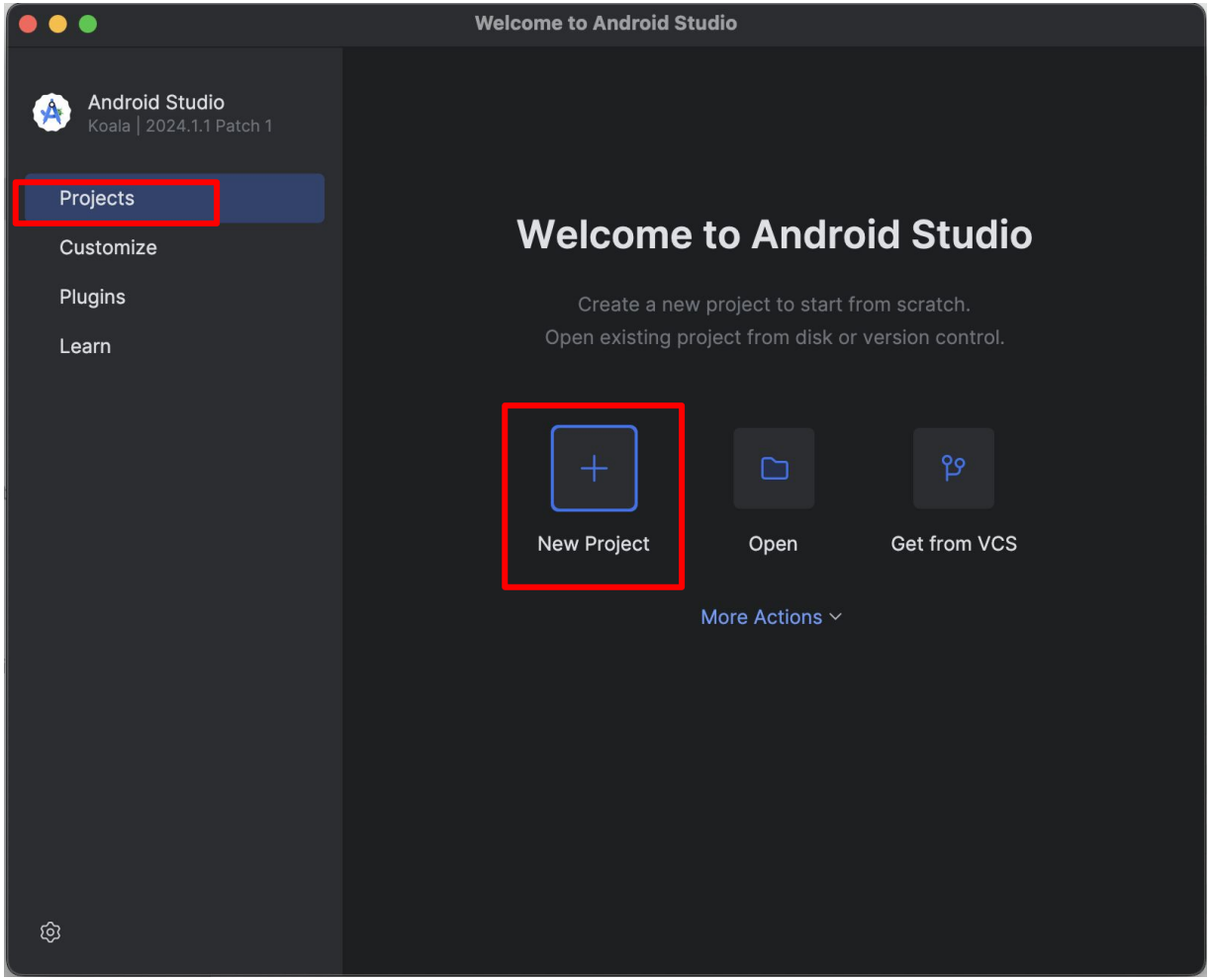
- To define the client…

Android Client

- Download and install ***Android Studio***
  - Browse to https://developer.android.com/studio/install.html
  - Follow the instructions to download, unpack, and install Android Studio for your kind of computer (Mac, MS Windows, Linux)
  - In the wizard, use all default values

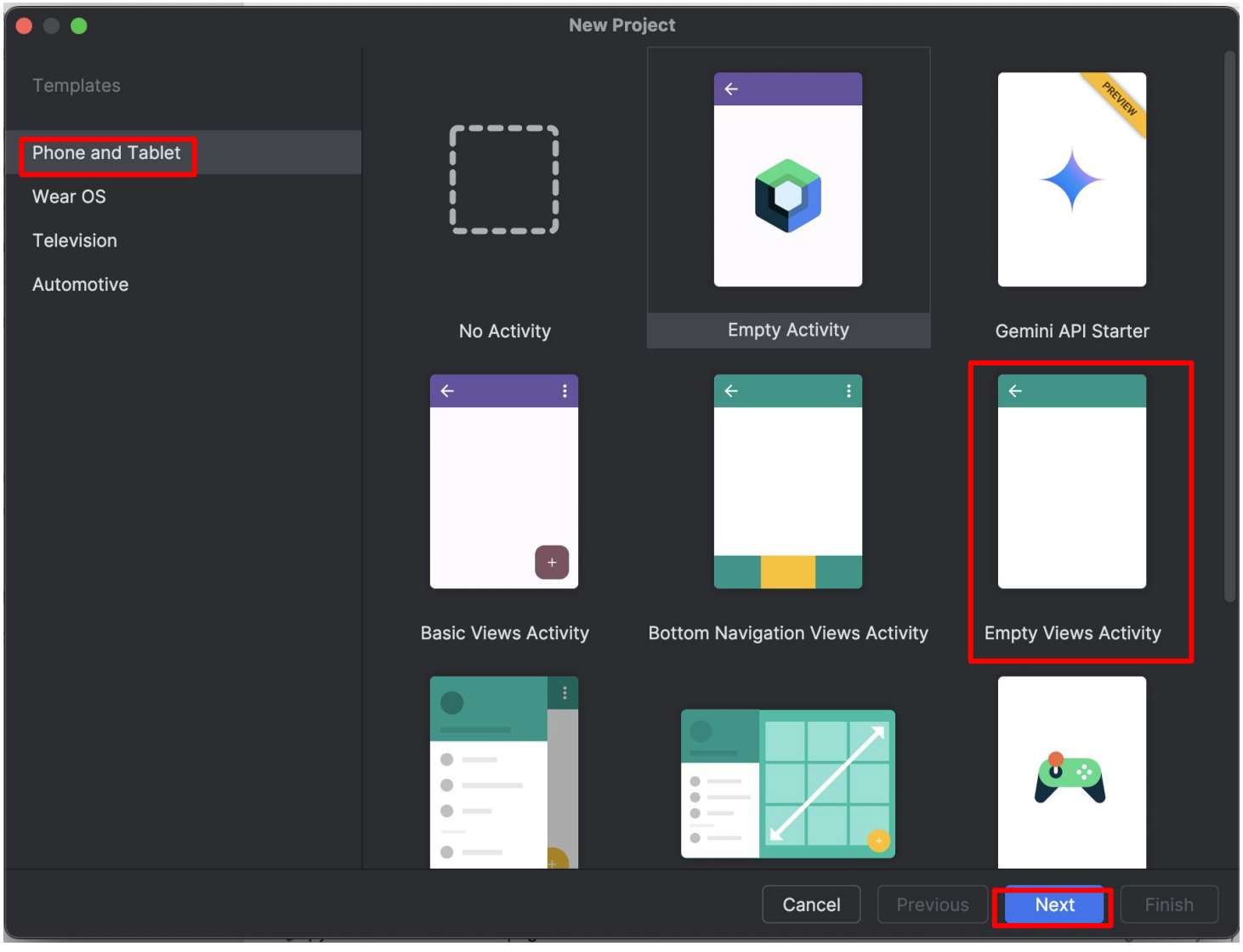
Android Client

Launch Android Studio; select *Projects*; click on *New Project*



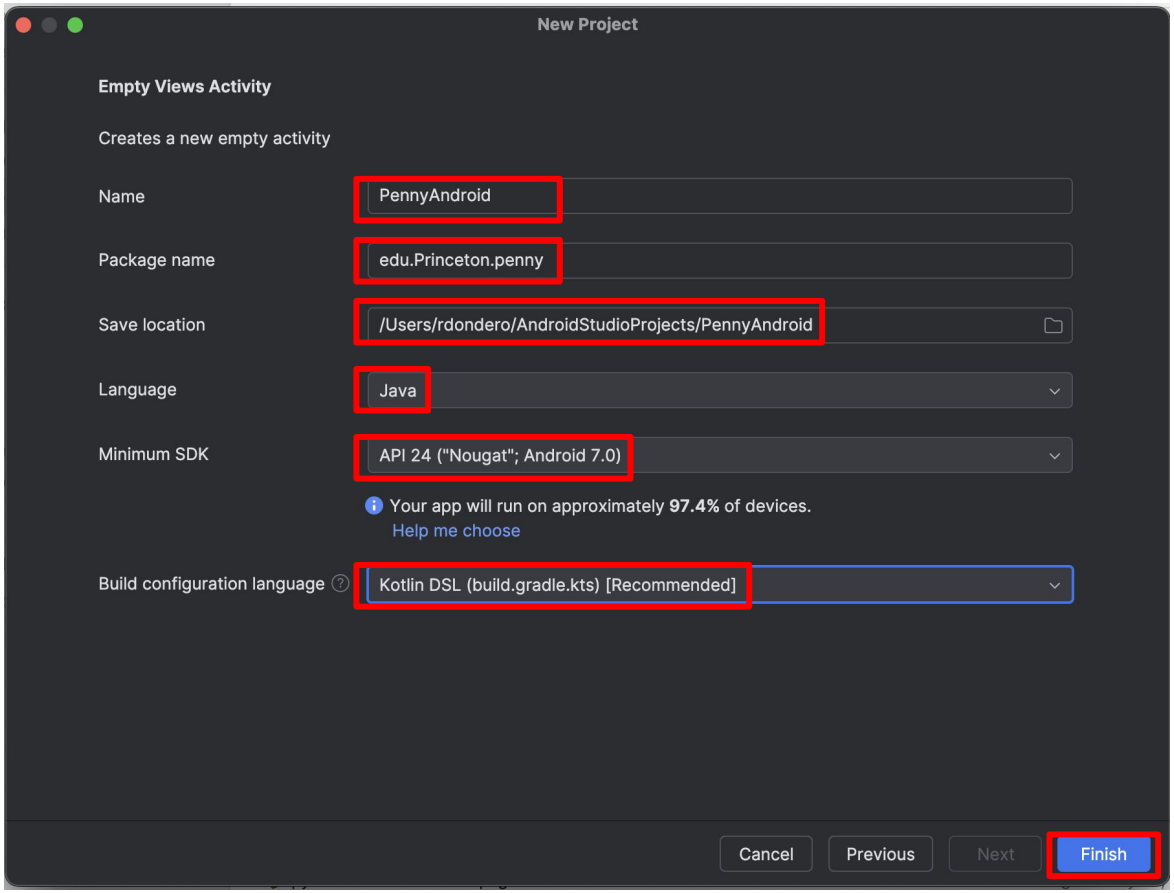
Android Client

Select Phone and Tablet; select *Empty Views Activity*; click on *Next*



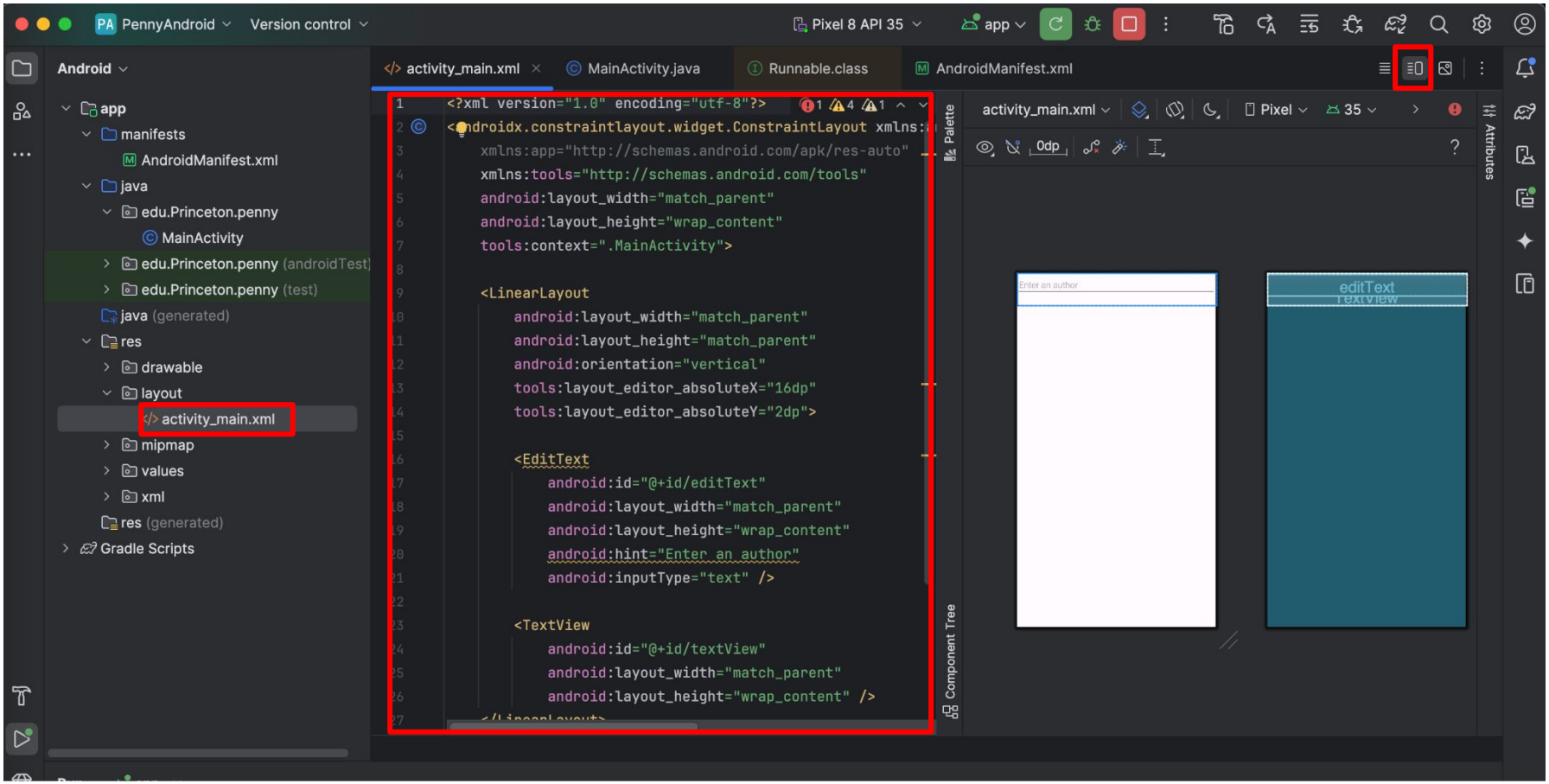
Android Client

For *Name* enter `PennyAndroid`; for *Package Name* enter `edu.Princeton.penny`; for *Save Location* choose whatever directory you want; for *Language* select `Java`; for *Minimum SDK* choose whatever you want; for Build configuration language choose `Kotlin DSL`; click on *Finish*



Android Client

In left panel double click on *app* > *res* > *layout* > *activity_main.xml*; at upper right, click on *Split* icon; copy **activity_main.xml** into editor

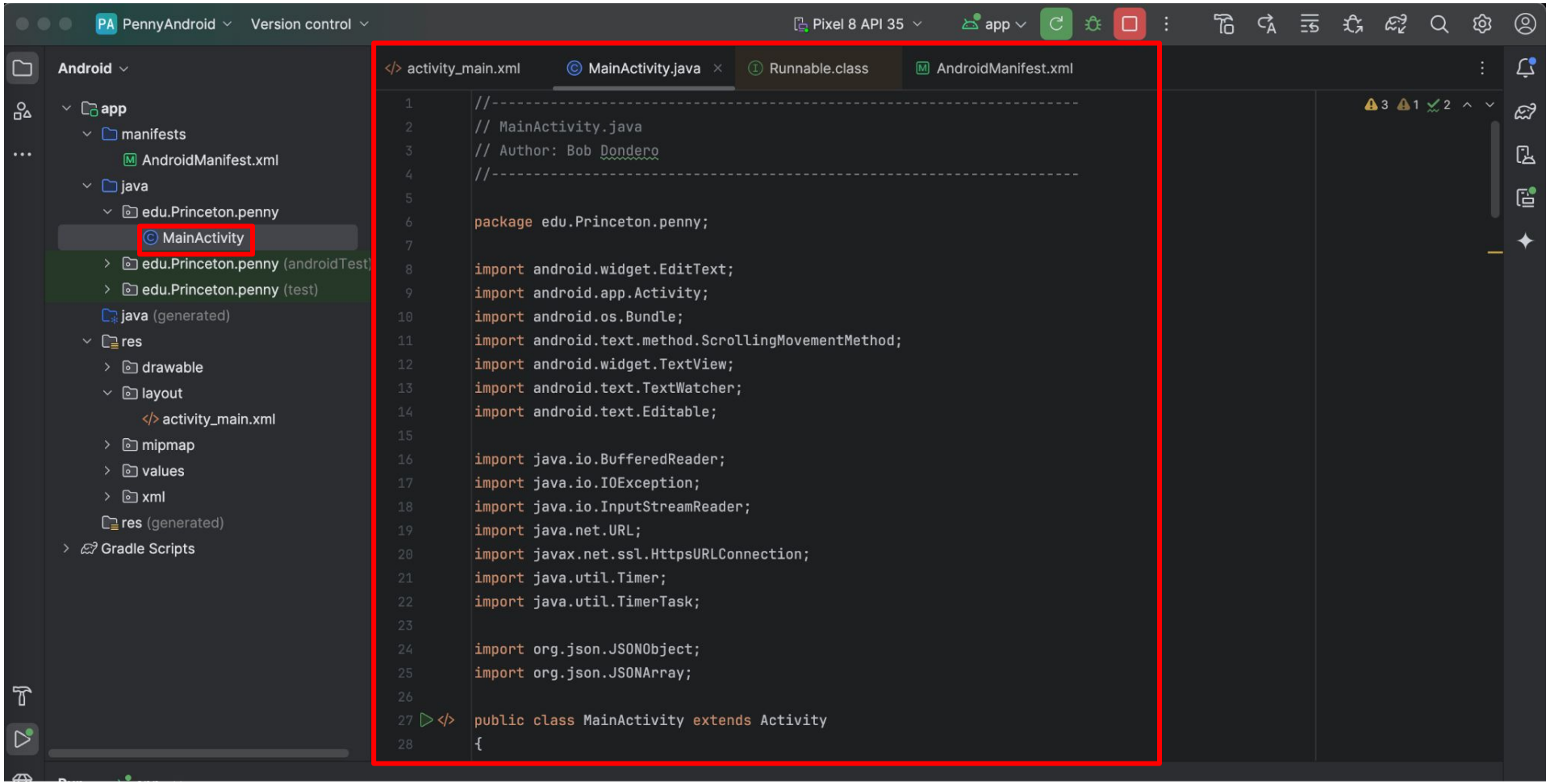


Android Client

- See **activity_main.xml**
  - Defines GUI
    - `LinearLayout` containing `EditText` and `TextView`
  - Suggestion:
    - Experiment with the graphical editor
    - Look at resulting XML code

Android Client

In left panel, double-click on *app* > *java* > *edu.Princeton.edu* > *MainActivity*; copy **MainActivity.java** into editor



Android Client

- See **MainActivity.java**
  - **Android design constraint 1**
    - GUI thread is not allowed to do networking
    - Implications:
      - GUI thread must spawn a child/worker thread
      - Child/worker thread must comm with server

Android Client

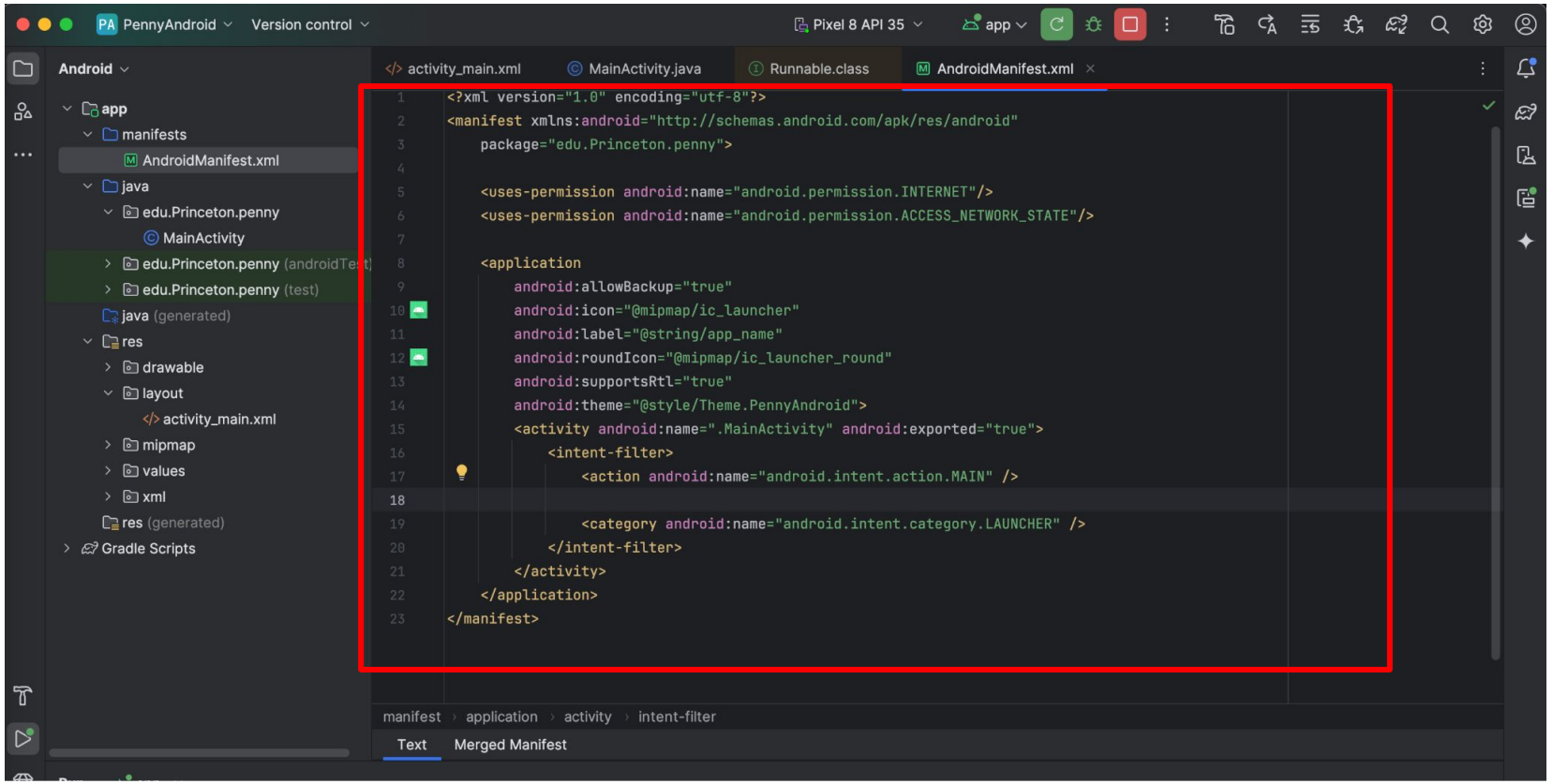
- See **MainActivity.java** (cont.)
  - **Android design constraint 2**
    - GUI thread must remain responsive
    - GUI thread laggy => typing/tapping fast generates "App is unresponsive" messages
    - Implications:
      - GUI thread cannot wait for child/worker thread to finish
      - GUI thread and child/worker thread must run concurrently

Android Client

- See **MainActivity.java** (cont.)
  - **Android design constraint 3**
    - Child/worker thread is not allowed to update GUI
    - Implications:
      - Child/worker thread must send changes to GUI thread, and ask GUI thread to update the GUI

Android Client

In left panel double-click on *app* > *manifests* > *AndroidManifest.xml*; copy text from **AndroidManifest.xml** into editor



Android Client

- See **AndroidManifest.xml**
  - `<uses-permission>` elements give app permission to access Internet

Android Client

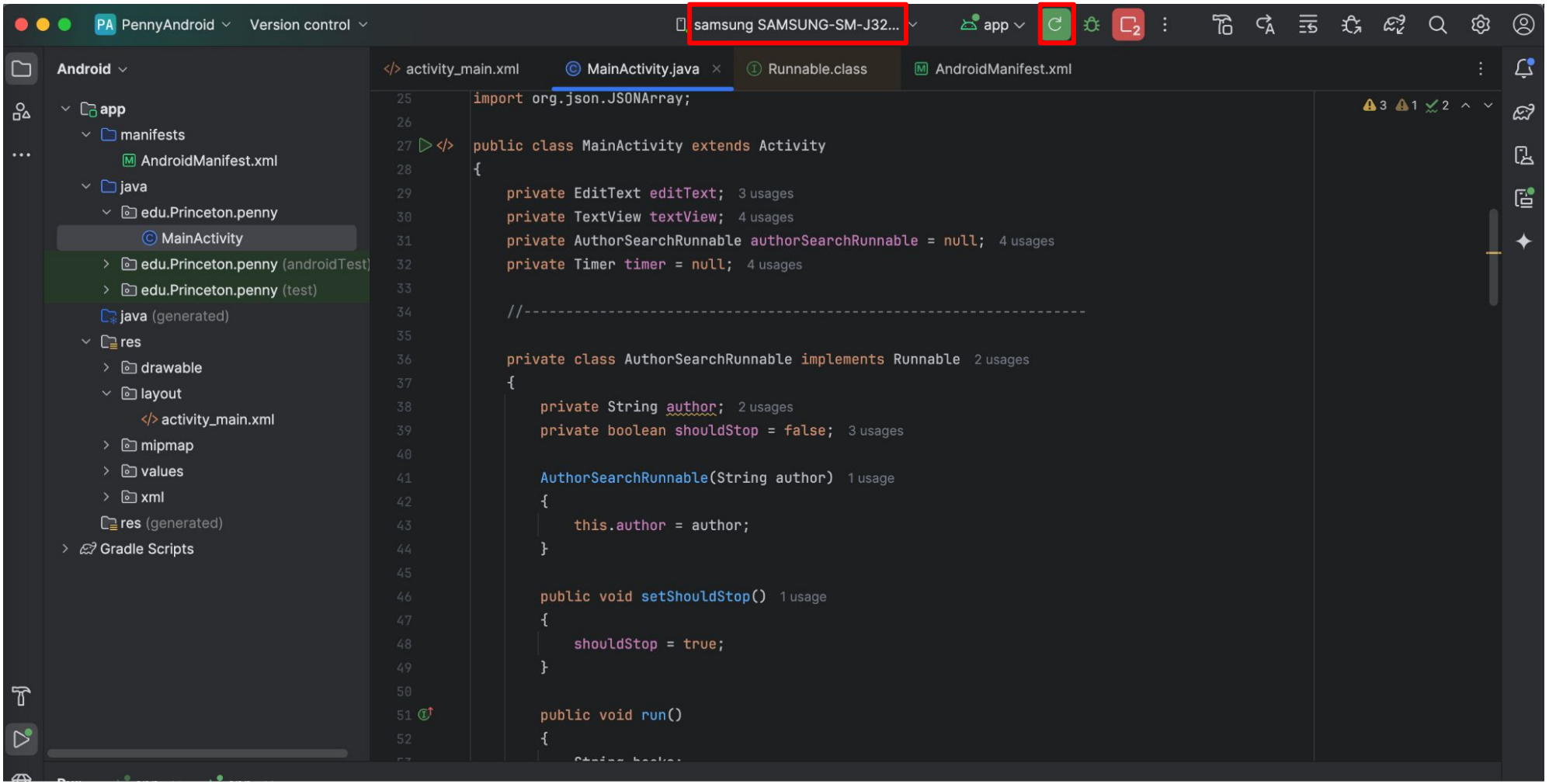
- To run the client…
- Option 1:
  - Use an Android **device**
  - Pro: Fast
- Option 2:
  - Create an Android **virtual device**
  - Run it (on any computer) using the Android **emulator**
  - Pro: Maybe more convenient

Android Client

- **Option 1**: Android device
  - Attach your Android device to your computer's USB port
  - Make sure your computer can access files stored on your Android device
    - Could be tricky; use the help available within Android Studio

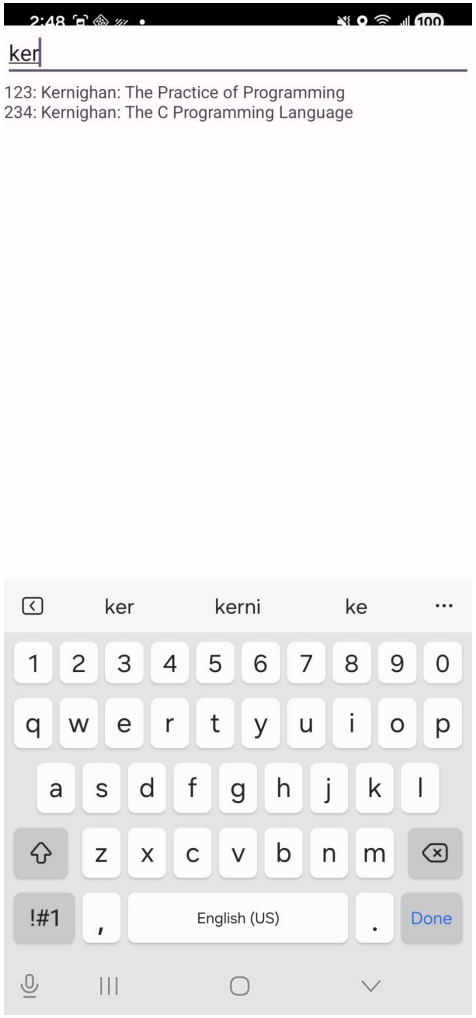
Android Client

Select your phone; click on *run* button; type an author!



Android Client

The result:



An Android client for the PennyJson server

Android Client

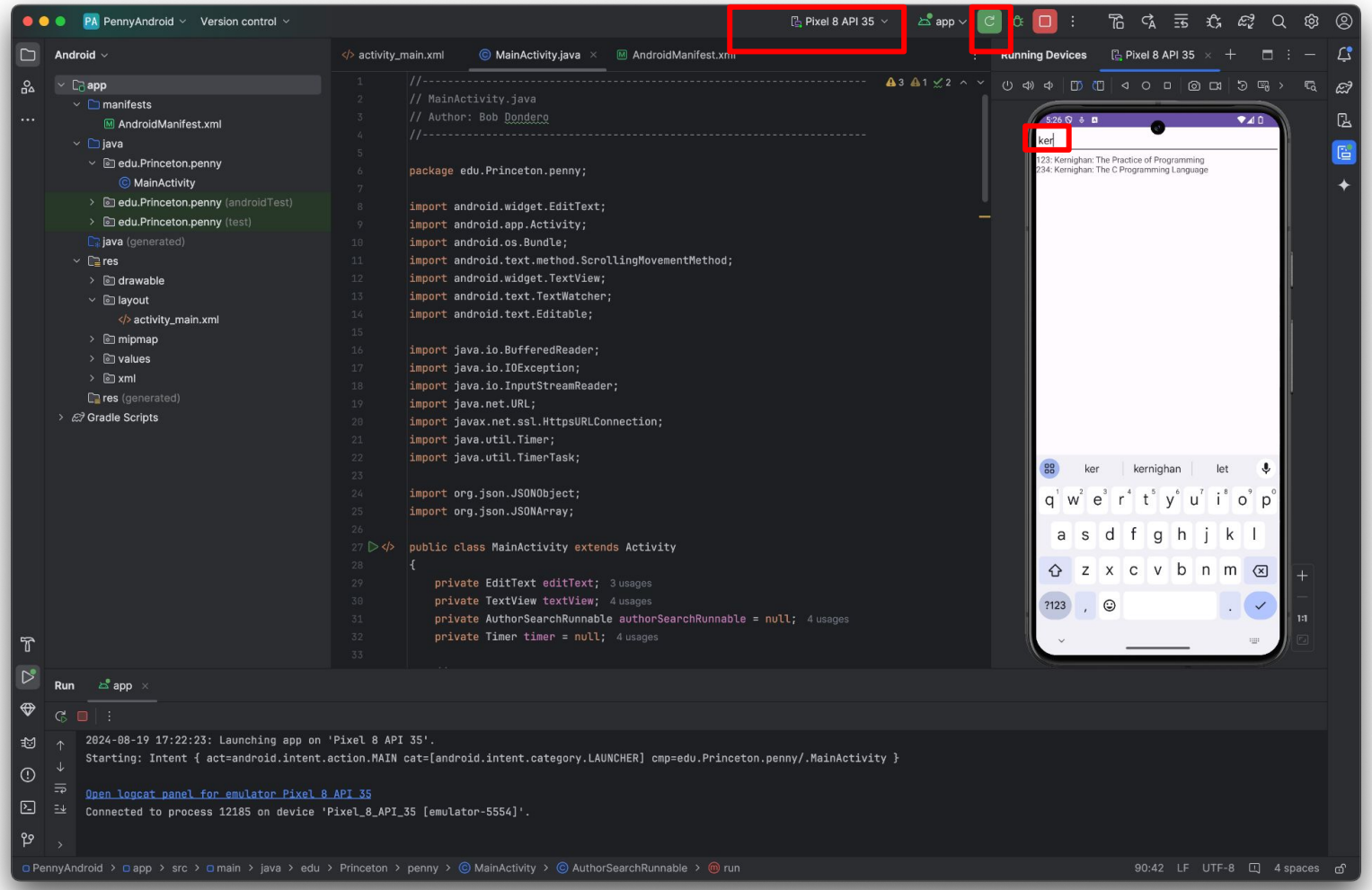
- **Option 2**: Android emulator…

Android Client

- Create an Android virtual device
  - In Android Studio
    - From the menu bar click *Tools*
    - Click *DeviceManager*
    - In the *Device Manager* panel,
    - Click the *"+"* button
    - Click *Create Virtual Device*
    - Click *Pixel 8*; click *Next*
    - Click *API 35*; click *Next*
    - Use the default PIXEL 8 API 35 name
    - Click *Finish*

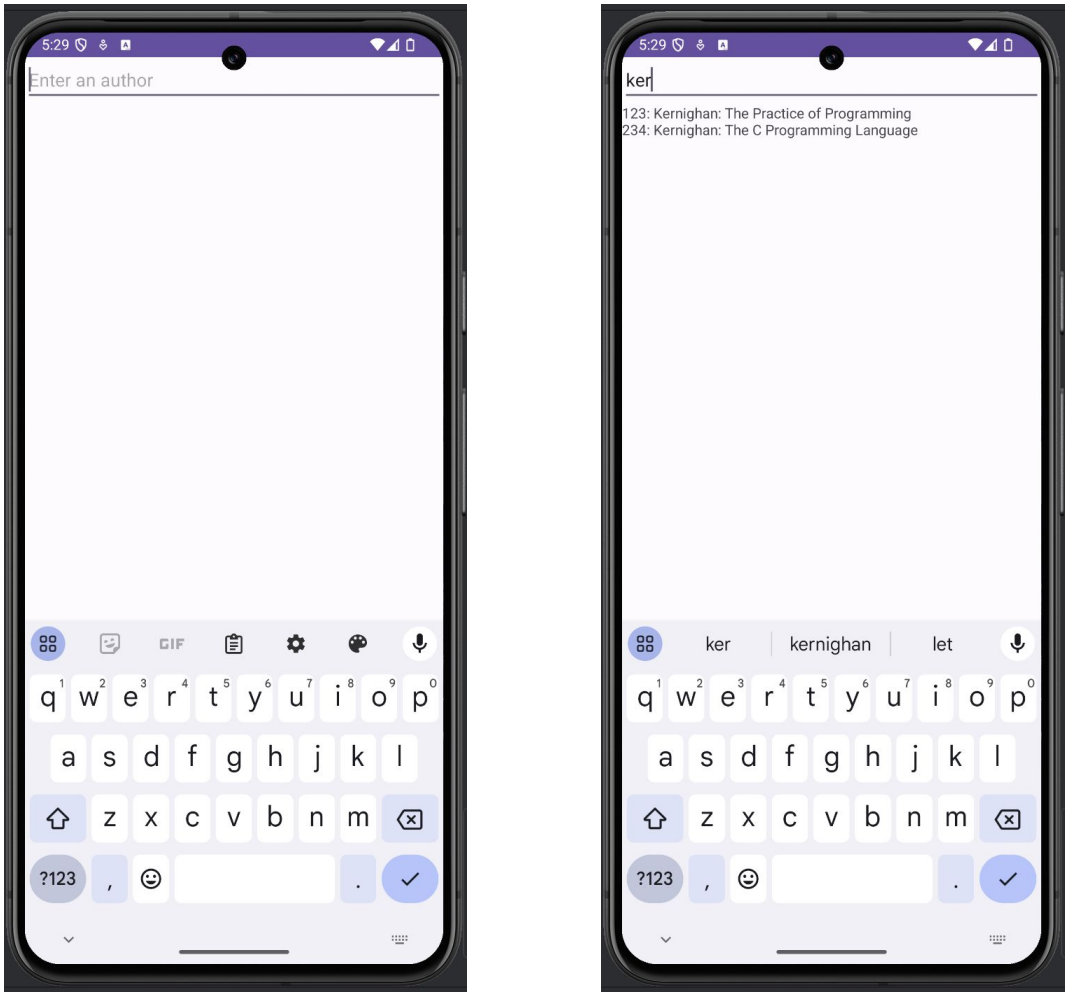
Android Client

Select the `Pixel 8 API 35` emulator; click *run* button; type an author!



Android Client

The result:



An Android client for the PennyJson server

Agenda

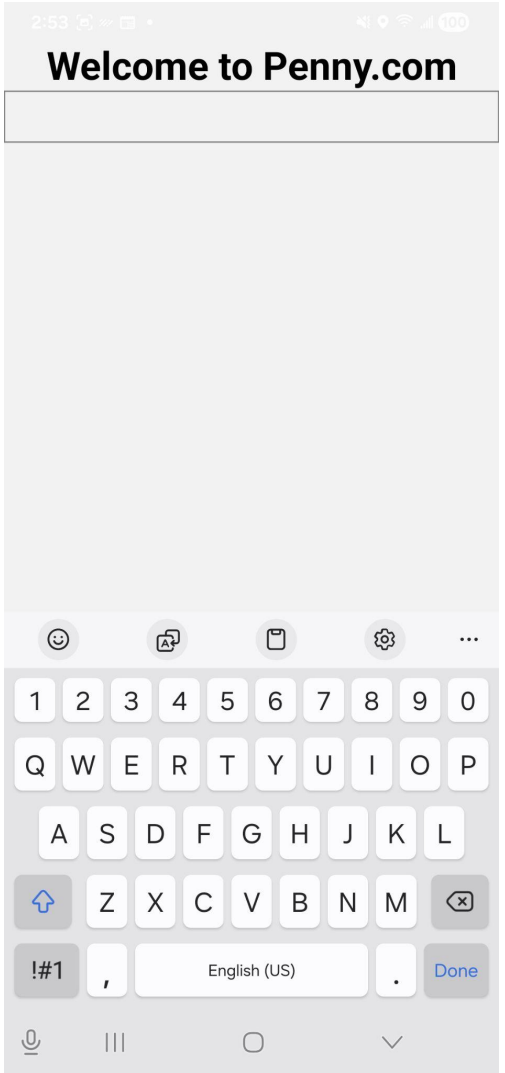
- Mobile clients
- Android client
- **React Native client**
- (Appendix) iOS client

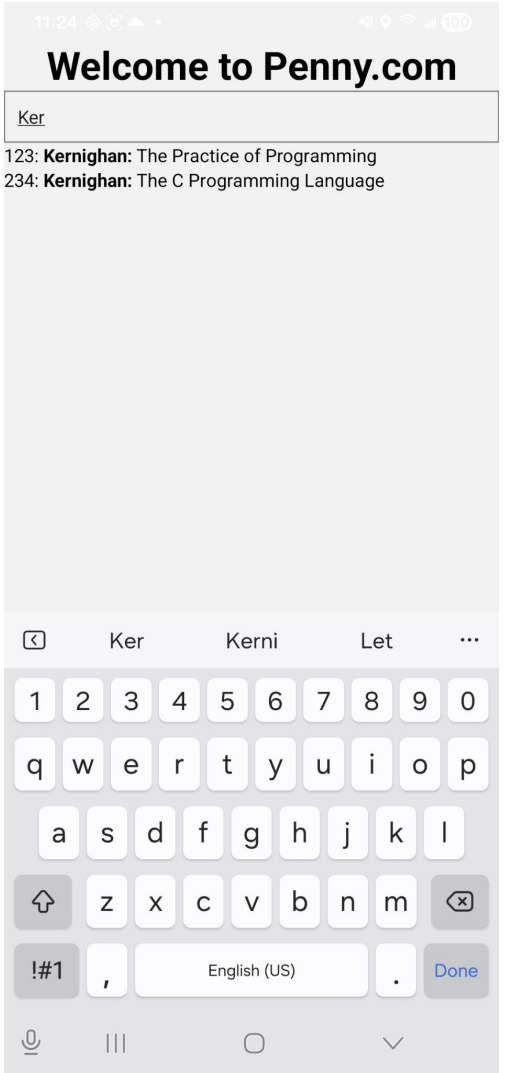
React Native Client

- Preliminary
  - Deploy PennyJson server to https://pennyjson.onrender.com
  - So PennyReactNative client can access it

React Native Client

The goal:





A React Native client for the PennyJson server

React Native Client

- To define the client…

React Native Client

On laptop computer:

```
$ npm install -g expo-cli
$ npx create-expo-app@3.5.3 \
  --template default@sdk-54 PennyReactNative
…
✅ Your project is ready!

To run your project, navigate to the directory
and run one of the following npm commands.

- cd PennyReactNative2
- npm run android
- npm run ios # you need to use macOS to build
the iOS project - use the Expo app if you need to
do iOS development without a Mac
- npm run web
$ cd PennyReactNative
$
```

React Native Client

The result:

```
PennyReactNative/
   app/
      (tabs)/
         layout.tsx
         explore.tsx
         index.tsx
      layout.tsx
      modal.tsx
   app.json
   assets/
      …
   components/
      …
   constants/
      …
```

```
PennyReactNative/
   hooks/
      …
   node_modules/
      …
   scripts/
      …
   package.json
   package-lock.json
   README.md
   tsconfig.json
```

.tsx => TypeScript and JSX code

React Native Client

- To run the client...

React Native Client

On laptop computer:

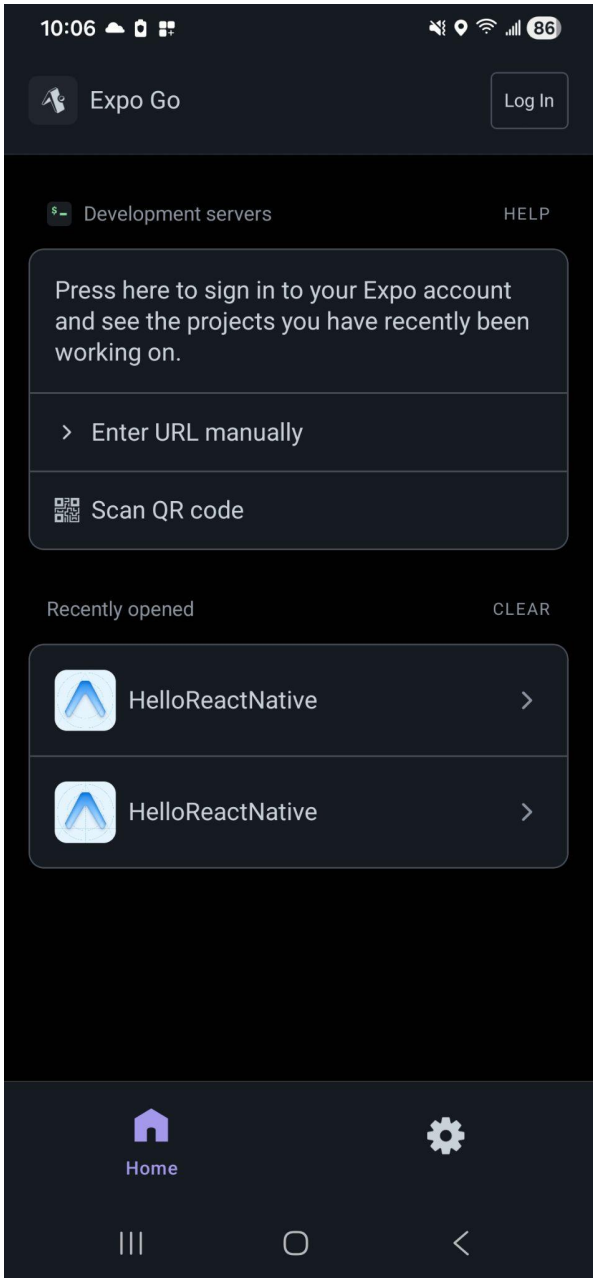
Note display of QR code

React Native Client

- **Option 1**: Android device
  - Install ExpoGo from PlayStore
  - Run ExpoGo

React Native Client

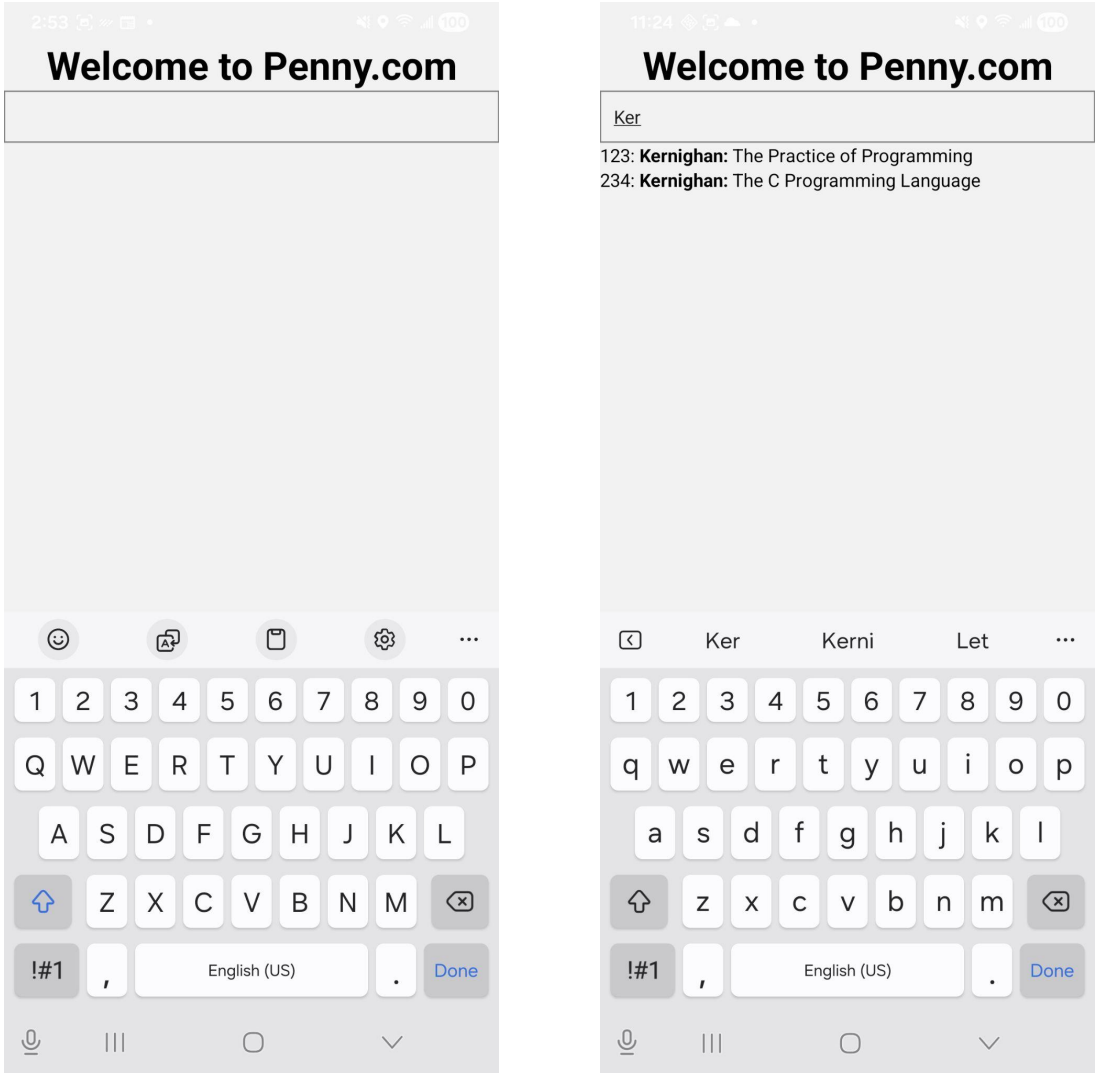
On Andriod device:



Click *Scan QR code*

React Native Client

The result:



A React Native client for the PennyJson server

React Native Client

- **Option 2**: Android emulator
  - Create and start an emulator in Android studio
  - Then:
    - `export ANDROID_HOME=/home/rdondero/Android/Sdk`
    - `npx expo start --android`

React Native Client

The result:

A ReactNative client for the PennyJson server

React Native Client

- Untested…
- **Option 3**: iOS device
  - Use Camera app to Scan QR code

React Native Client

- Untested…
- **Option 4**: iOS emulator
  - Create and start an emulator in XCode
  - Then:
    - `npx expo start --ios`

Lecture Summary

- In this lecture we covered:
  - Mobile programming
  - Android mobile programming
  - React Native mobile programming

- See also:
  - **Appendix 1**: iOS Mobile Programming

Lecture Summary

Server-side technologies

Python/CGI
Python/WSGI
Python/Flask
Python/Django (optional lecture)
Java/Spark
Java/Servlets (appendix)
Java/Spring (appendix)
JavaScript/Express

Lecture Summary

Client-side technologies

HTML/JavaScript
HTML/JavaScript/jQuery
HTML/JavaScript/React
Python/CLI
Java/Swing
Java/Android
JavaScript/React Native
Swift/iOS (appendix)

Can match any client-side technology with any server-side technology

Appendix 1:
iOS Mobile Programming

Thanks to
Katie DiPaola (‘26)...

iOS Client

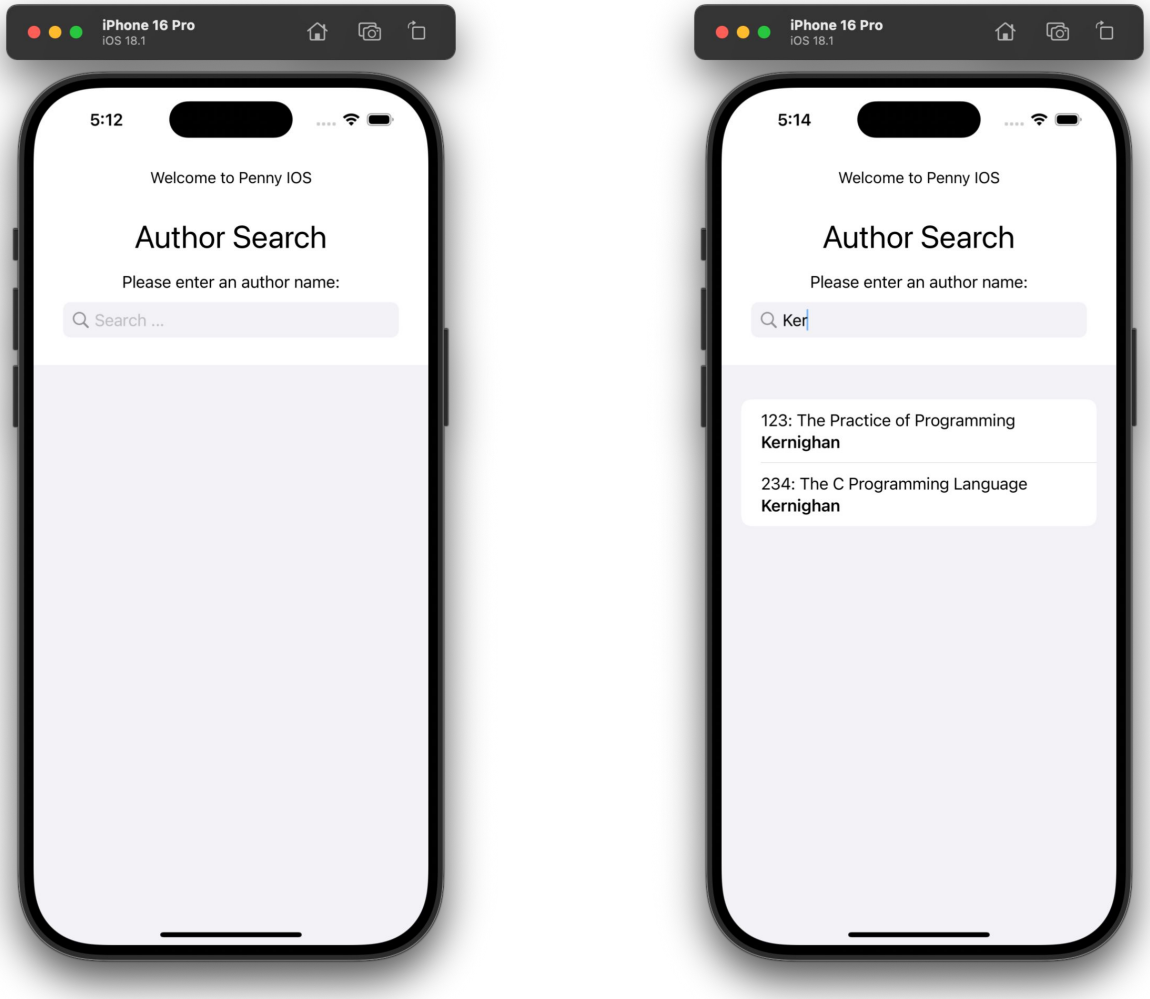
- Preliminary
  - Deploy PennyJson server to https://pennyjson.onrender.com
  - So Pennylos client can access it

iOS Client

- To define the client…

iOS Client

The goal:



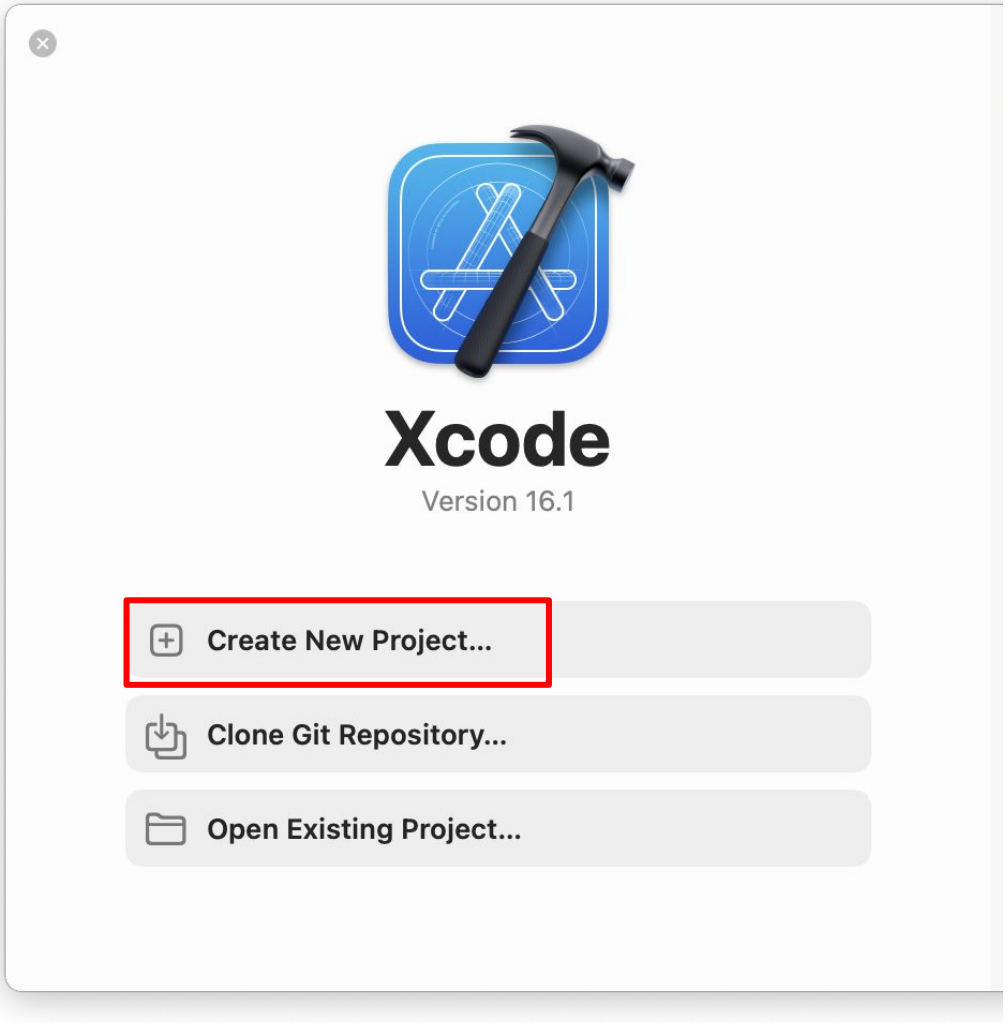
An iOS client for the PennyJson server

iOS Client

- Download and install ***XCode***

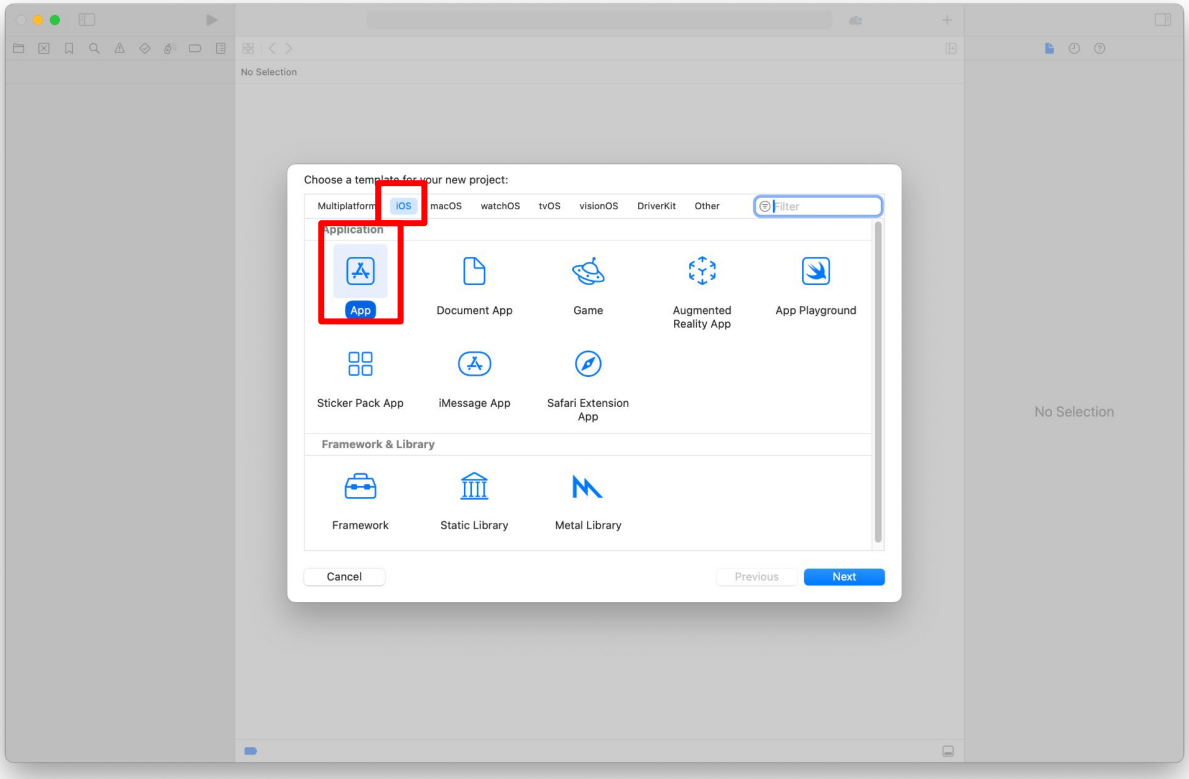
iOS Client

Launch XCode; select *Create New Project…*



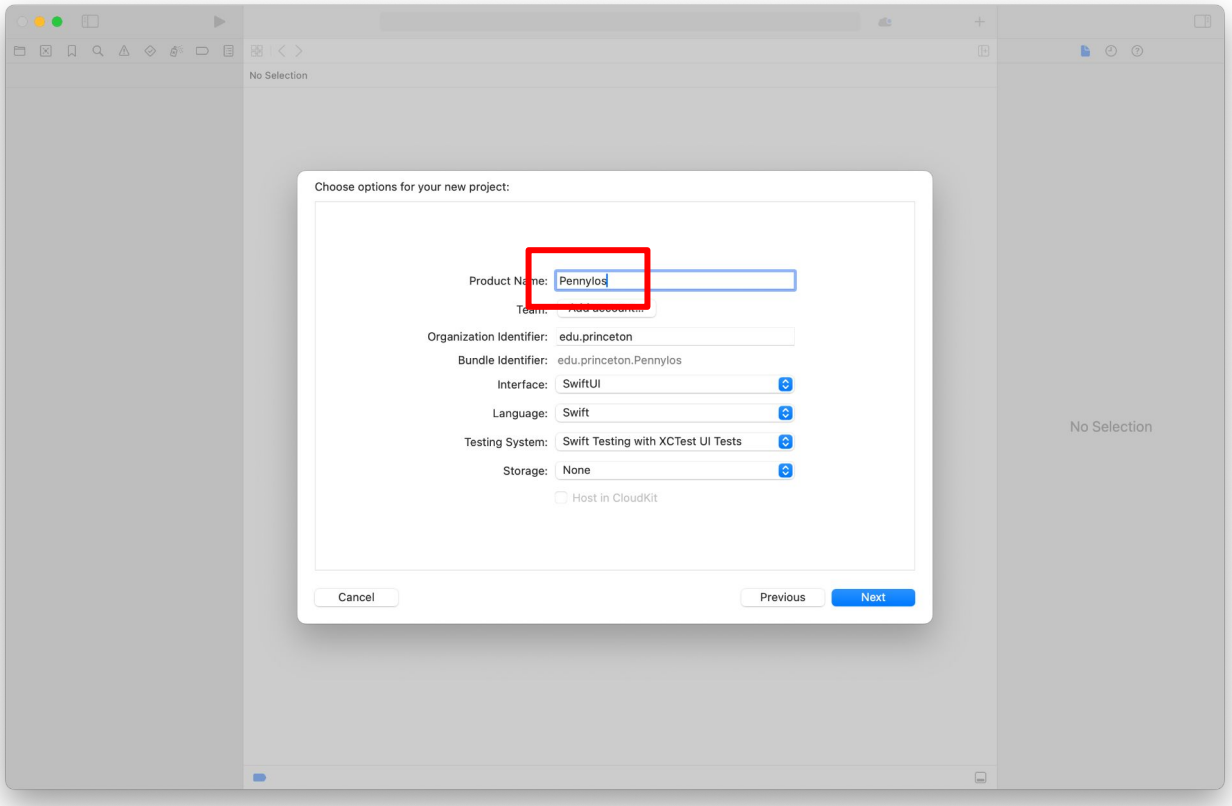
iOS Client

Select *iOS*, *App*; click on *Next*



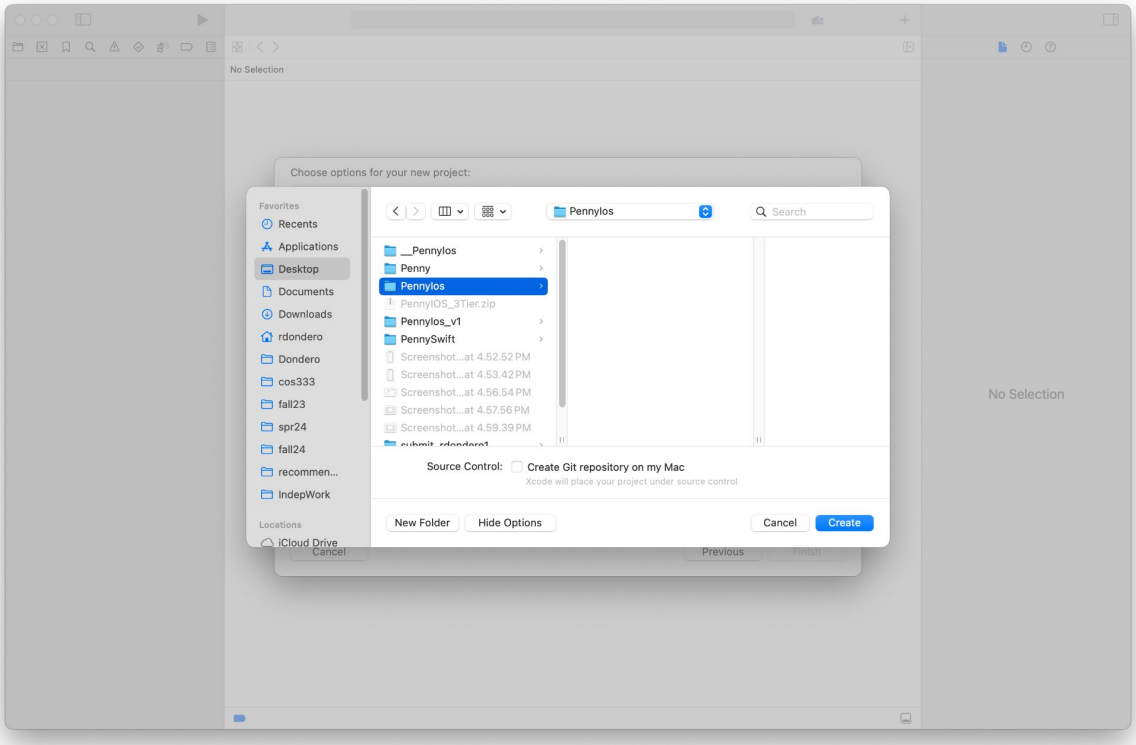
iOS Client

For Product *Name* enter `PennyIos`; click on *Next*



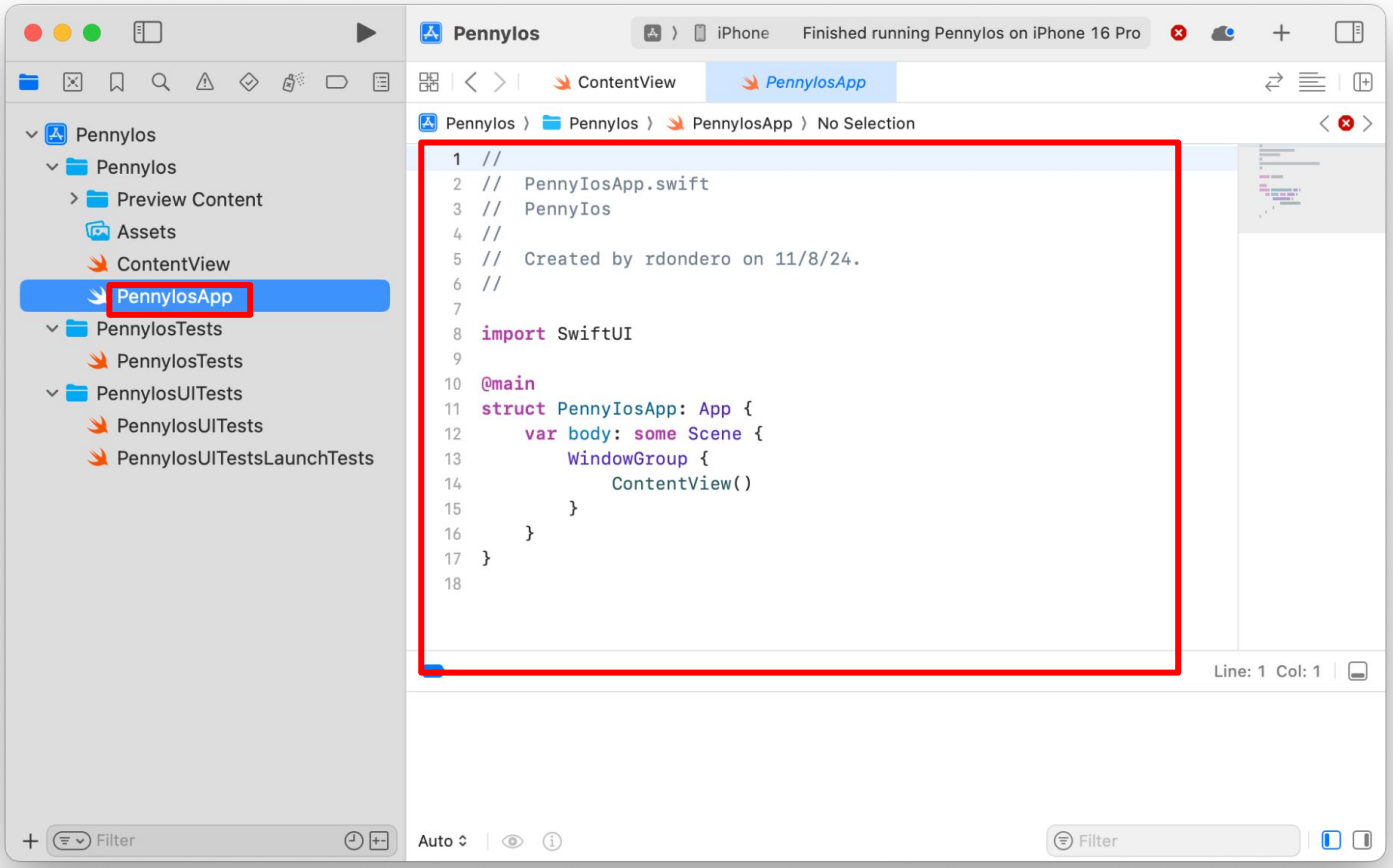
iOS Client

Name a directory in which the app should be stored; click on *Create*



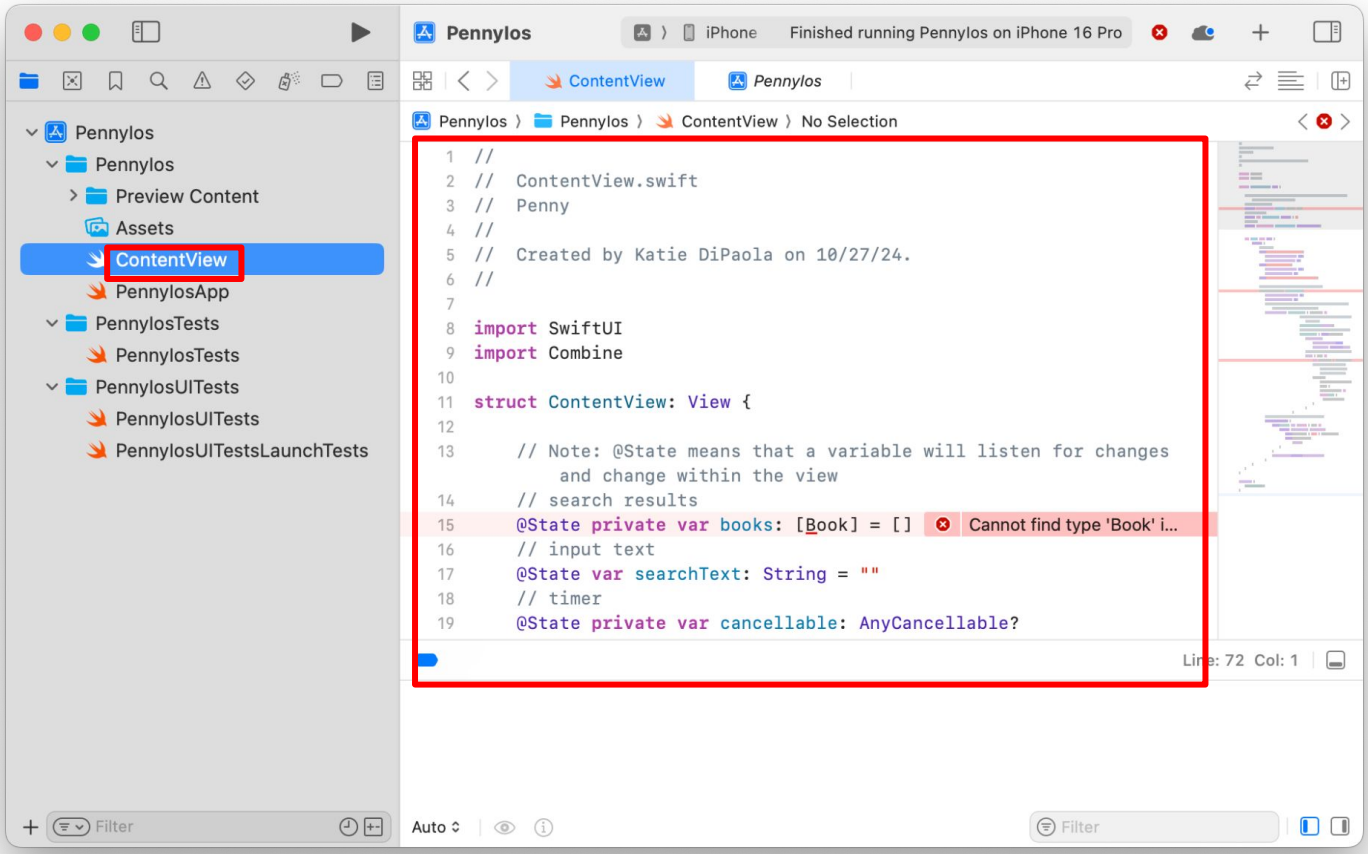
iOS Client

Click on *PennylosApp*; examine the generated code



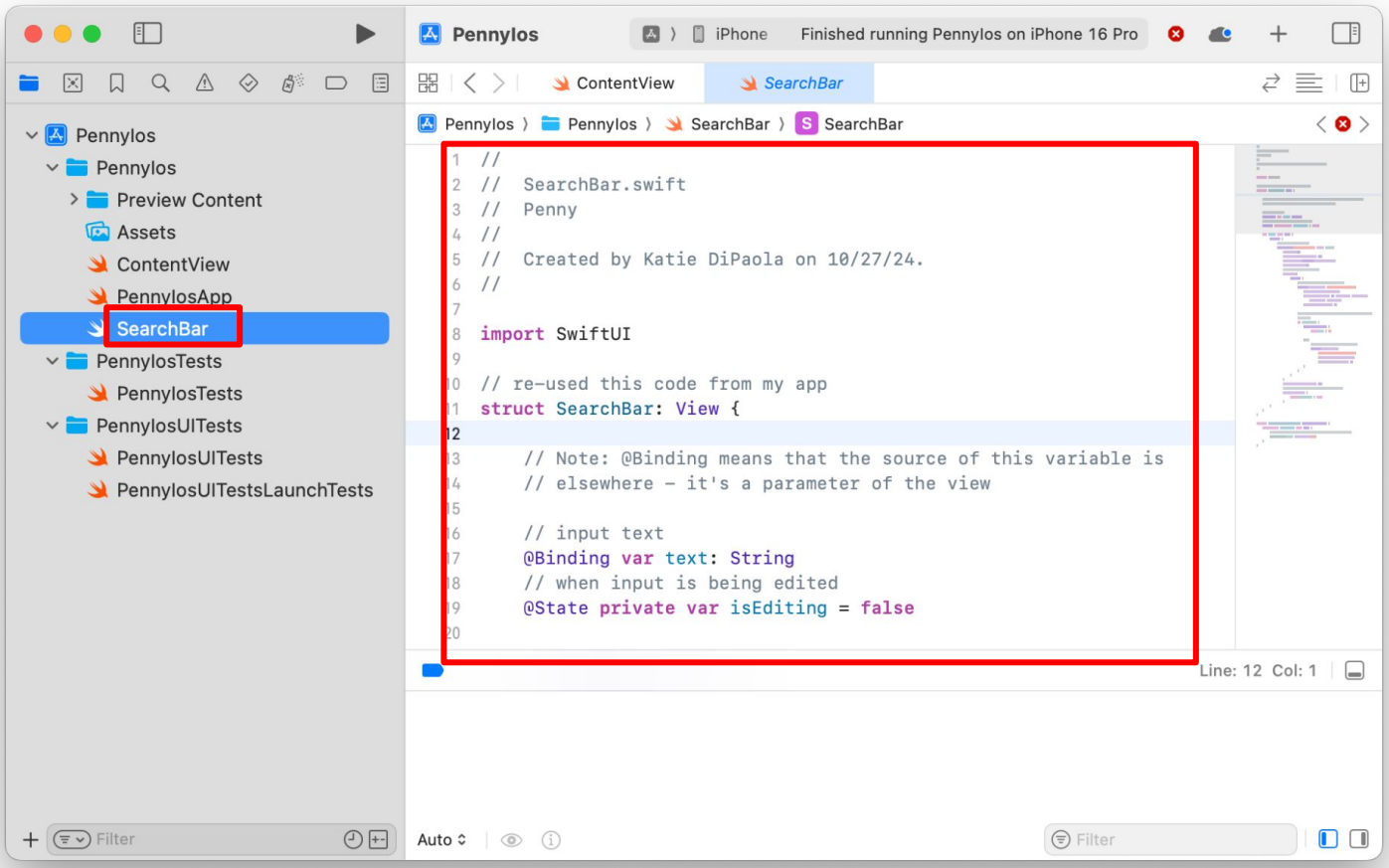
iOS Client

Click on *ContentView*; copy/paste the given code



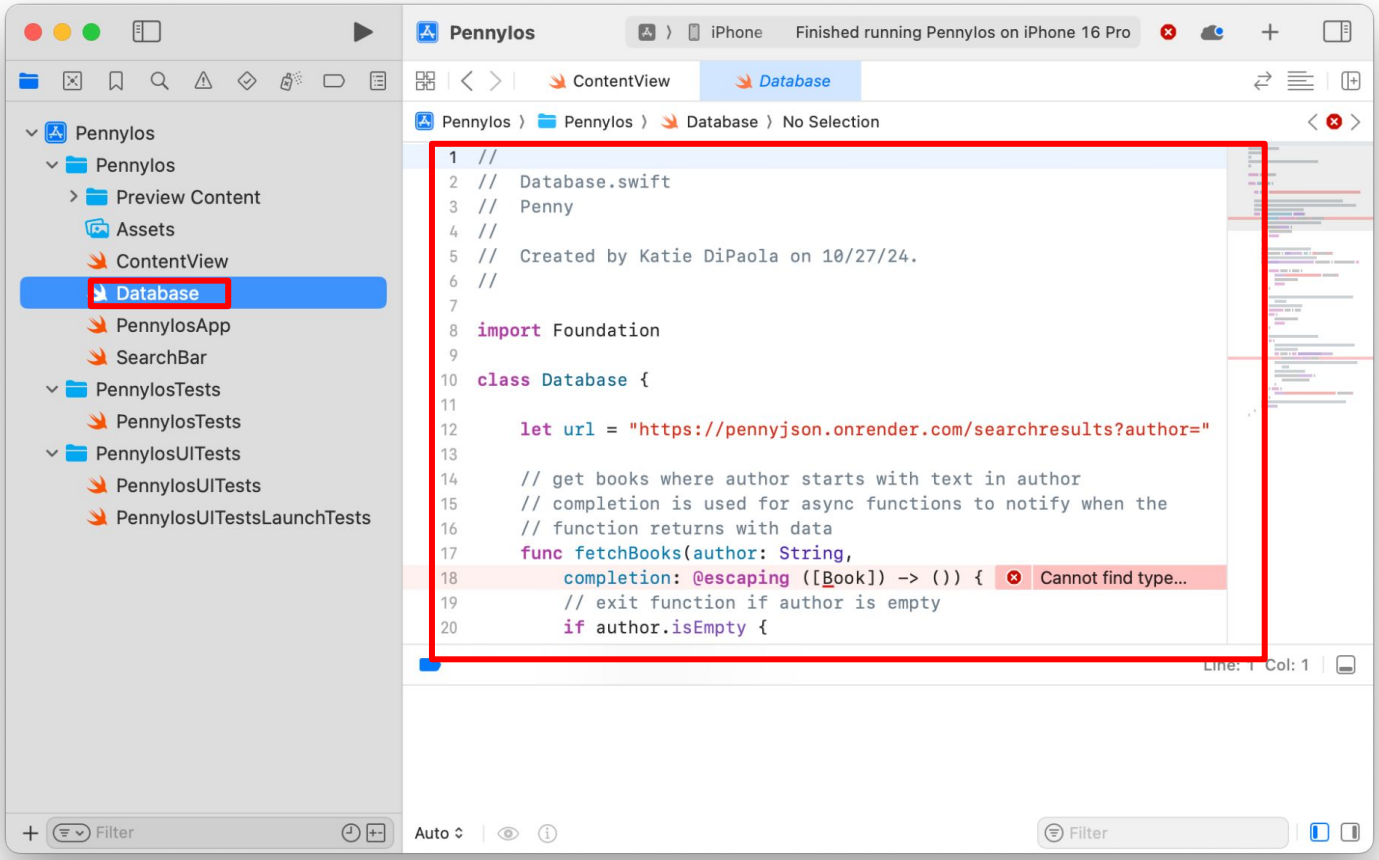
iOS Client

Create new file *SearchBar*; copy/paste the given code



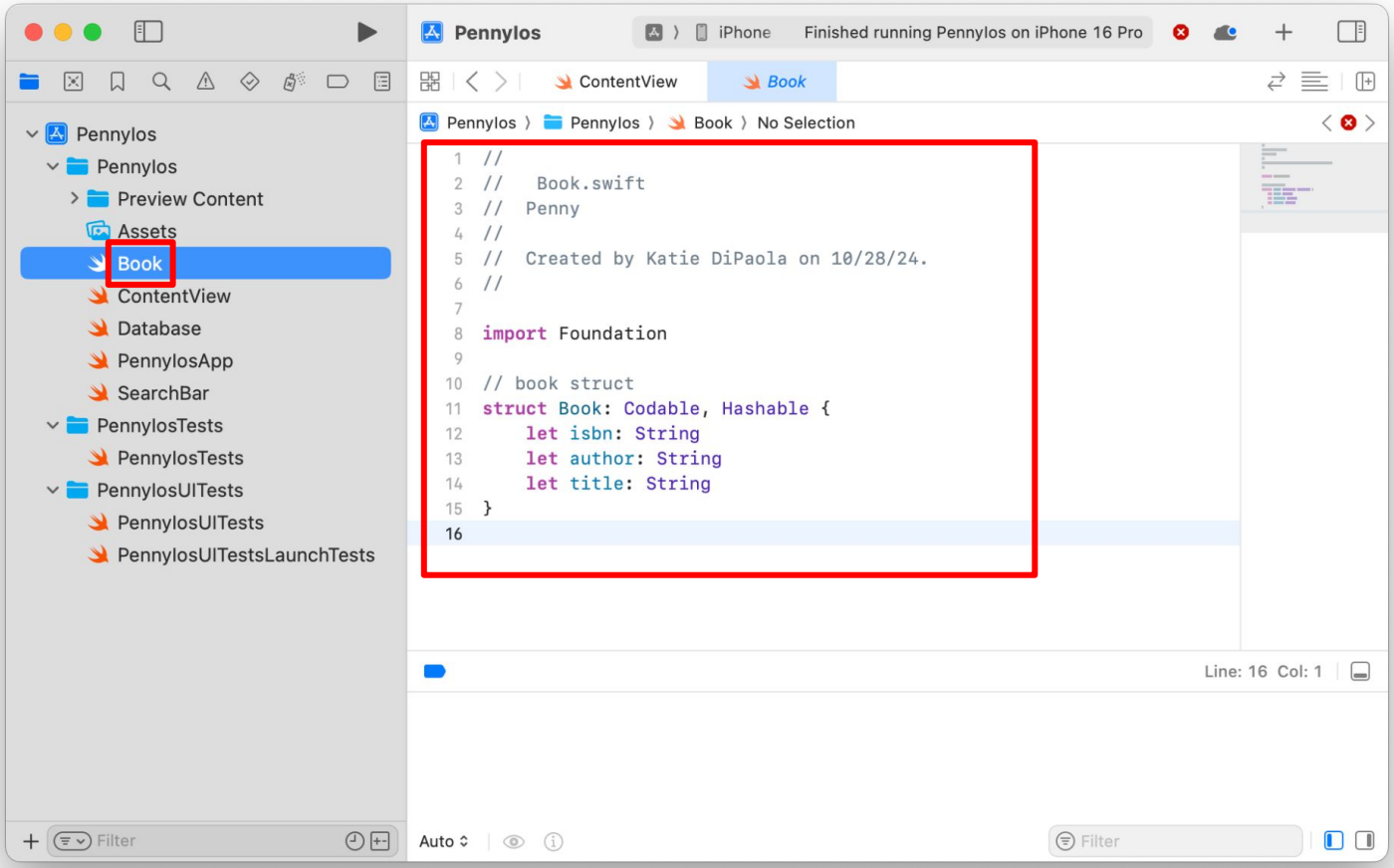
iOS Client

Create new file *Database*; copy/paste the given code



iOS Client

Create new file *Book*; copy/paste the given code

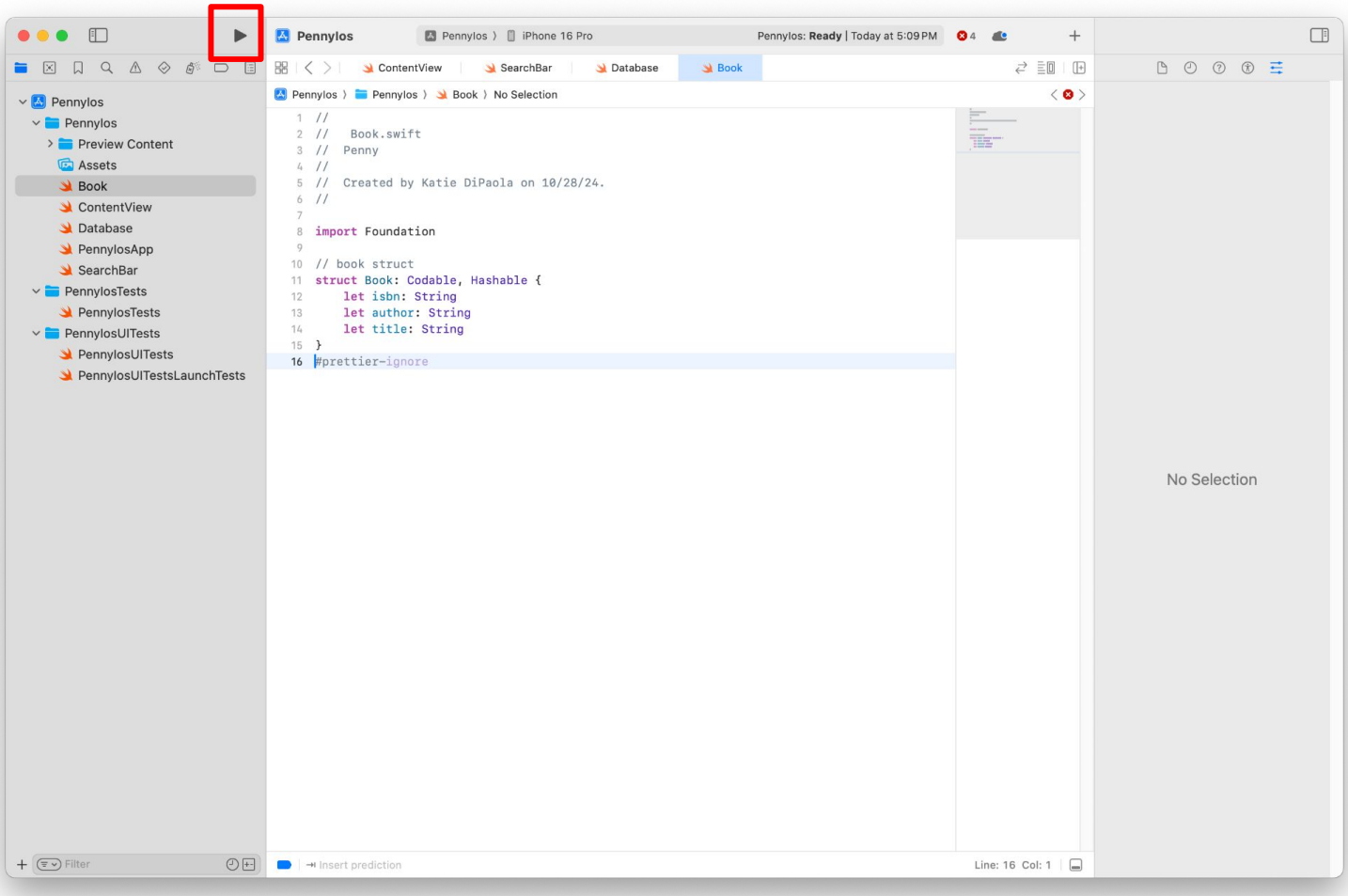


iOS Client

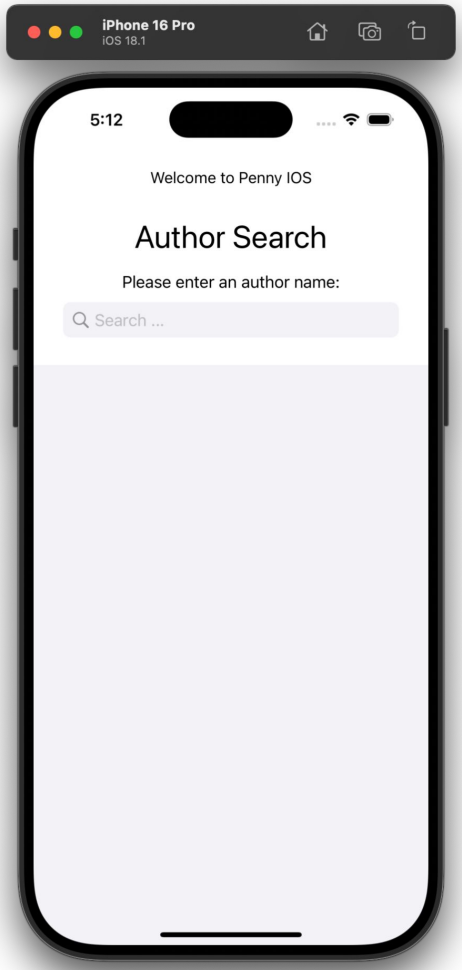
- To run the client…

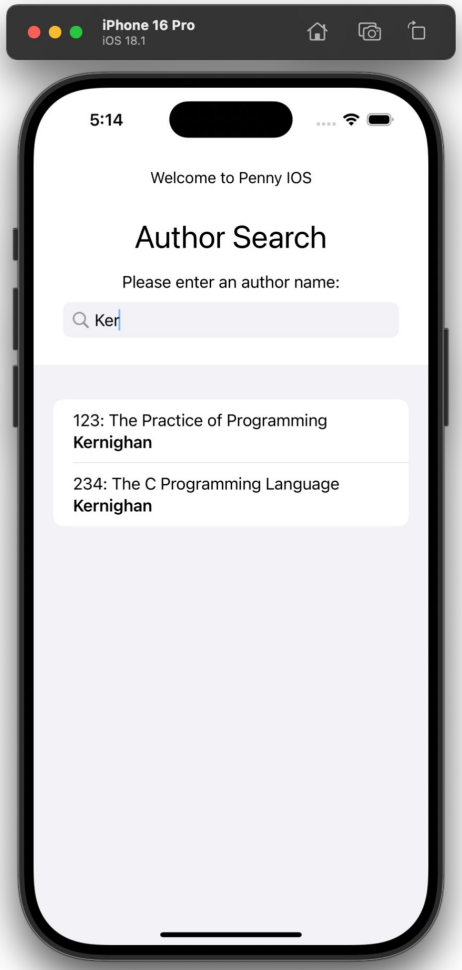
iOS Client

Click on the *Build/Run* button



iOS Client


iPhone 16 Pro
iOS 18.1
5:12
Welcome to Penny IOS
Author Search
Please enter an author name:
Search ...


iPhone 16 Pro
iOS 18.1
5:14
Welcome to Penny IOS
Author Search
Please enter an author name:
Ker
123: The Practice of Programming
Kernighan
234: The C Programming Language
Kernighan

Acknowledgement

Our thanks go to **Katie DiPaola** (‘26) for contributing the Pennylos application, and to **Christine Sun** (‘24) for contributing a prior (now outdated) version