Client-Side Options (Part 2)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

So far:

Client	Browser + HTML + JavaScript Java + Swing
Server	Python + Flask Java + Servlets Java + Spring JavaScript + Express

Objectives

- We will cover:
  - Mobile programming
  - Android mobile programming
  - Appendix: iOS mobile programming

Agenda

- **Mobile programming**
- PennyAndroid app: defining
- PennyAndroid app: running

Mobile Programming

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

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

Mobile Programming

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

Mobile Programming

- See https://whatwebcando.today/

- Examples: If you need ___ do you have an option?
  - **Offline mode**: Yes
  - **Audio & video capture**: Probably
  - **Proximity sensors**: Probably not
  - **Contacts**: No

Mobile Programming

- 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 Programming

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

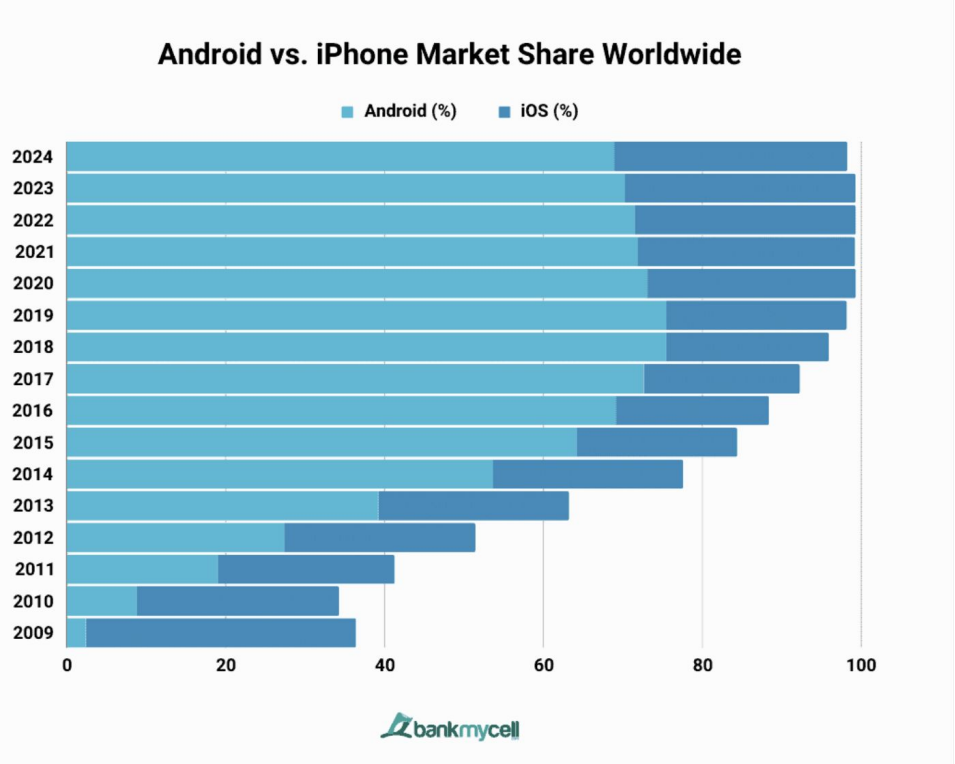
* May need multiple native mobile apps

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

Mobile Programming

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/

Agenda

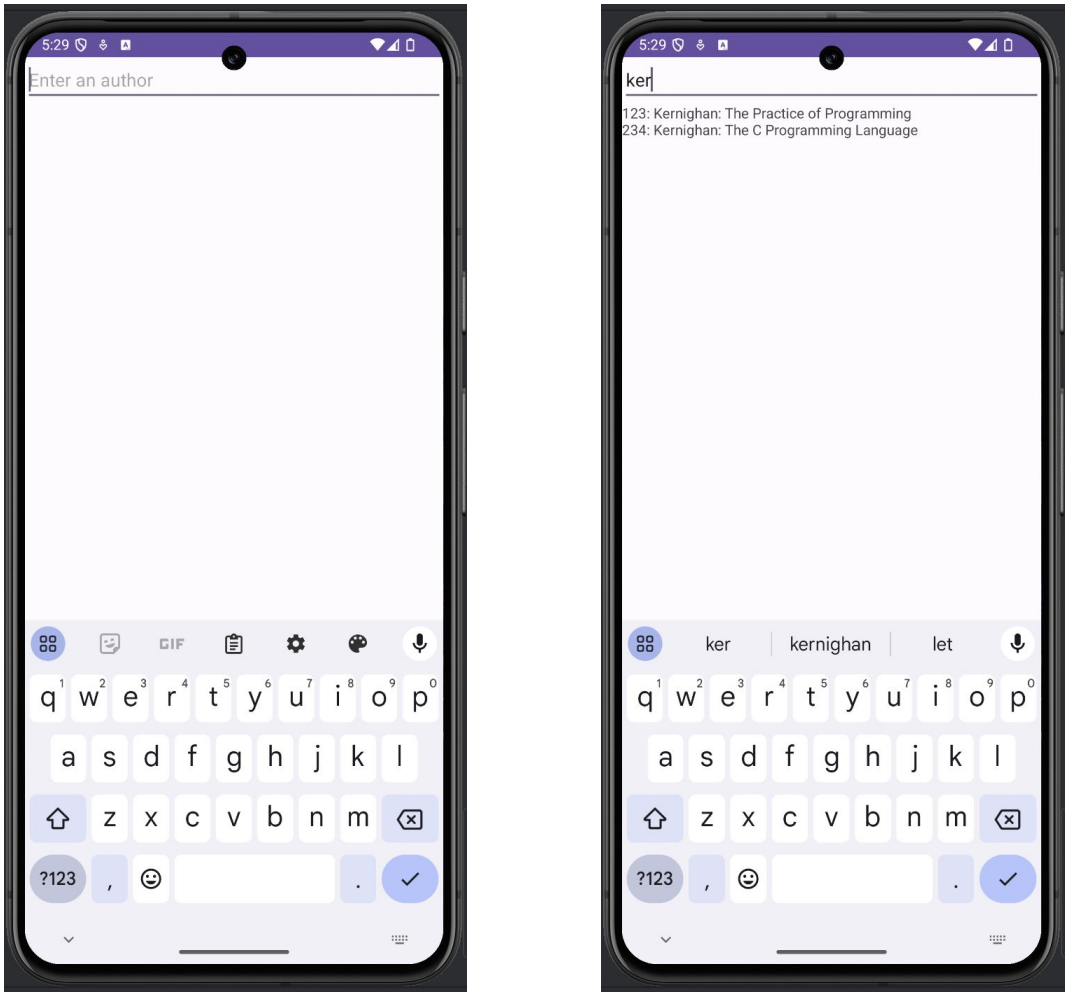
- Mobile programming
- **PennyAndroid app: defining**
- PennyAndroid app: running

PennyAndroid App: Defining

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

PennyAndroid App: Defining

The goal:



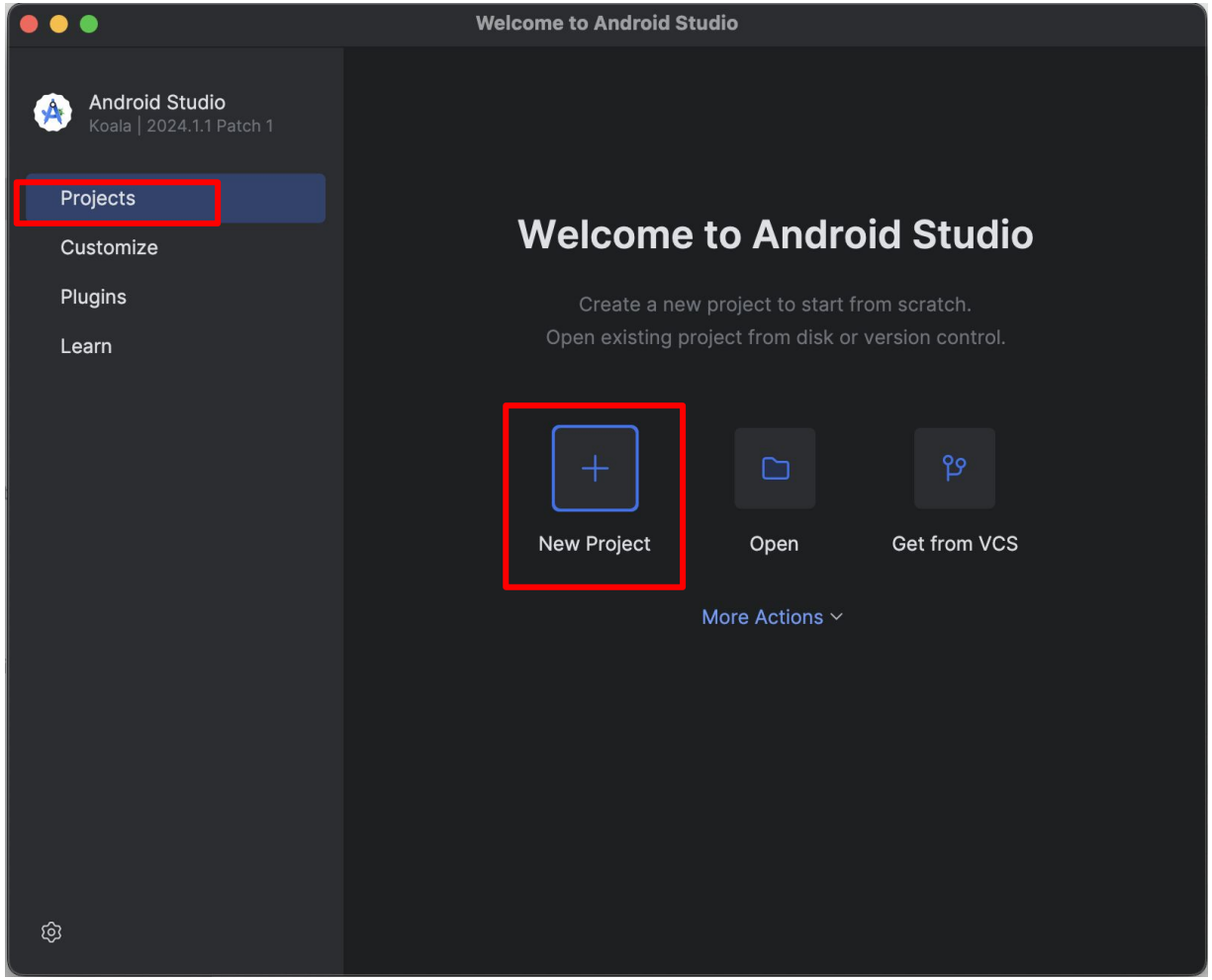
An Android client for the PennyJson server

PennyAndroid App: Defining

- Download and install ***Android Studio***
  - Browse to https://developer.android.com/studio/install.html
  - Complete the wizard; use defaults

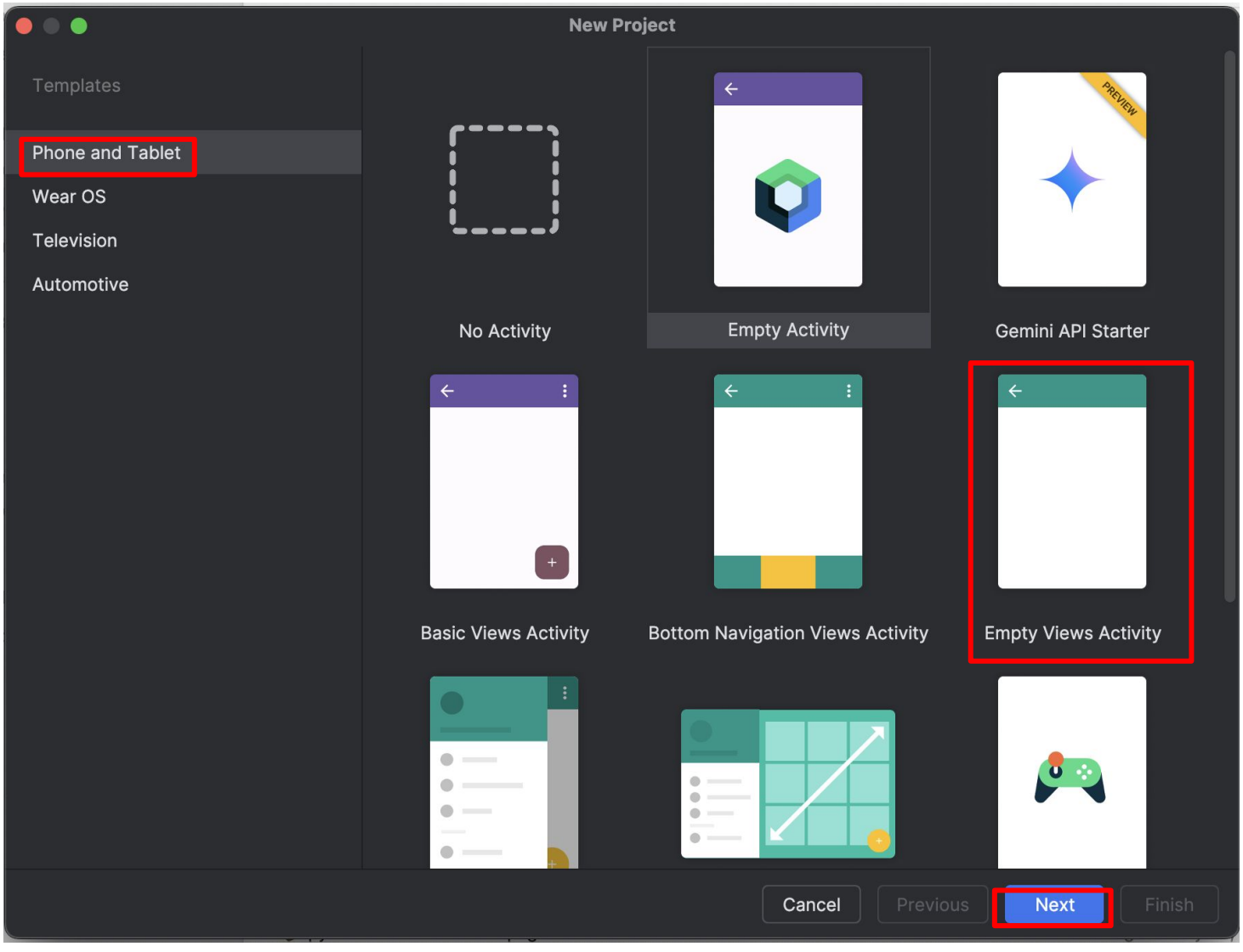
PennyAndroid App: Defining

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



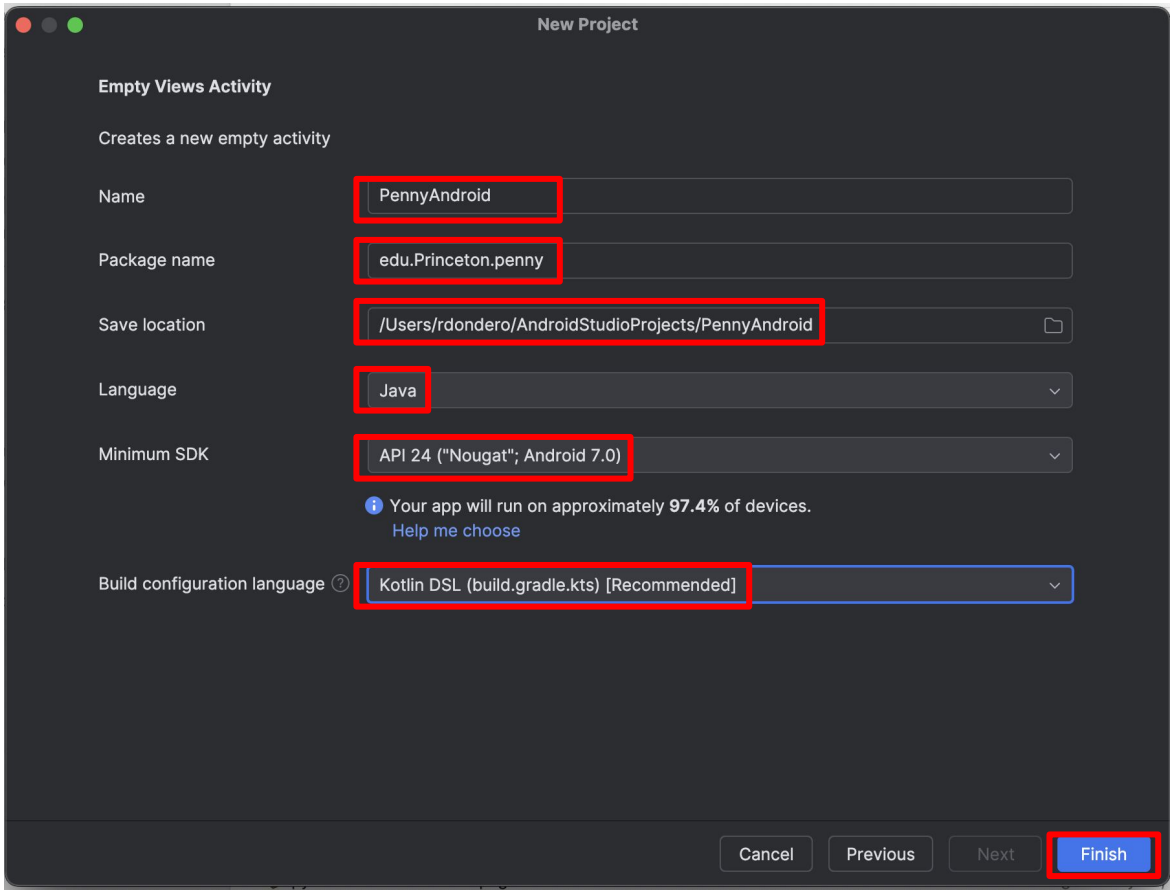
PennyAndroid App: Defining

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



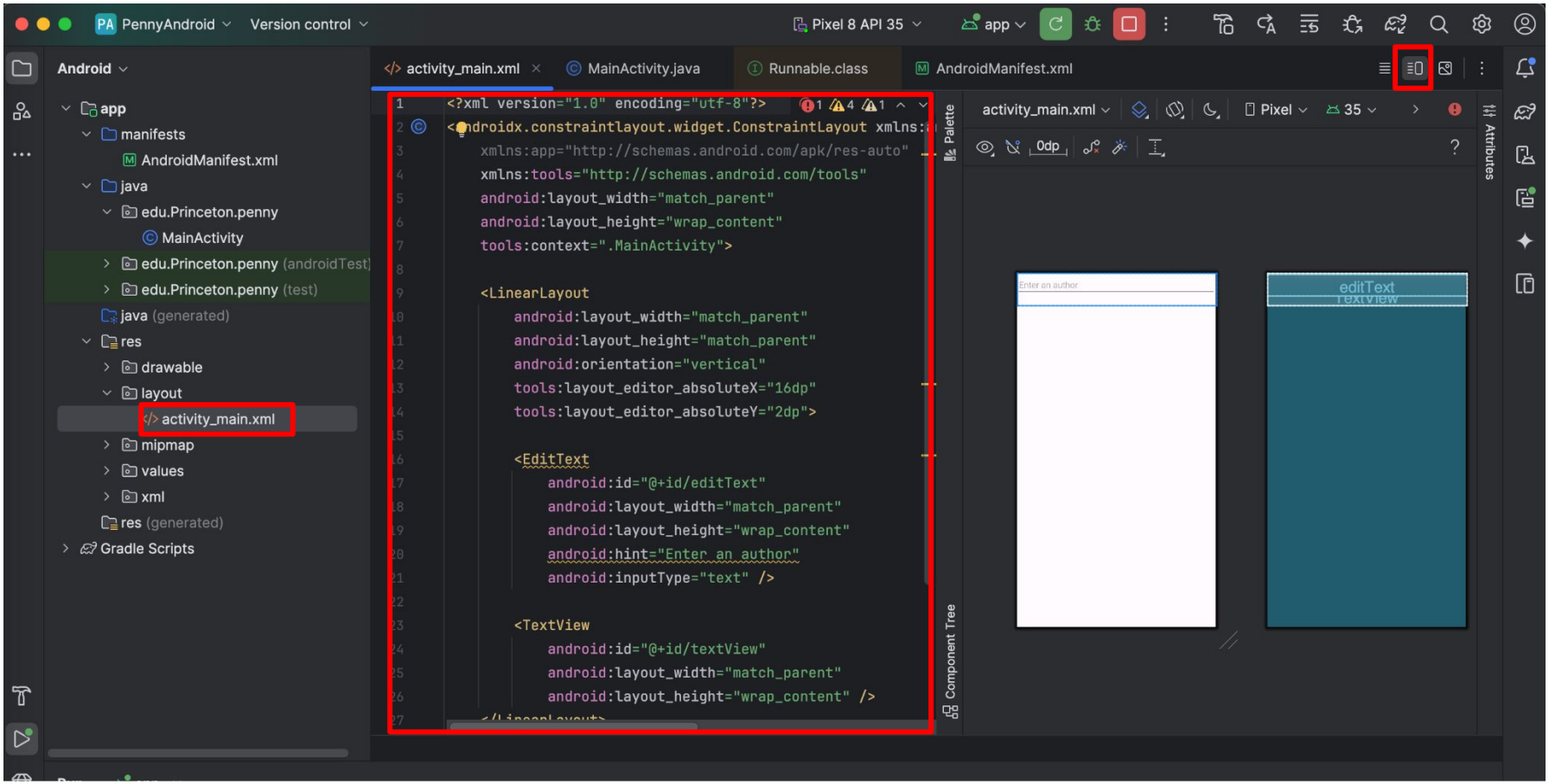
PennyAndroid App: Defining

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*



PennyAndroid App: Defining

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

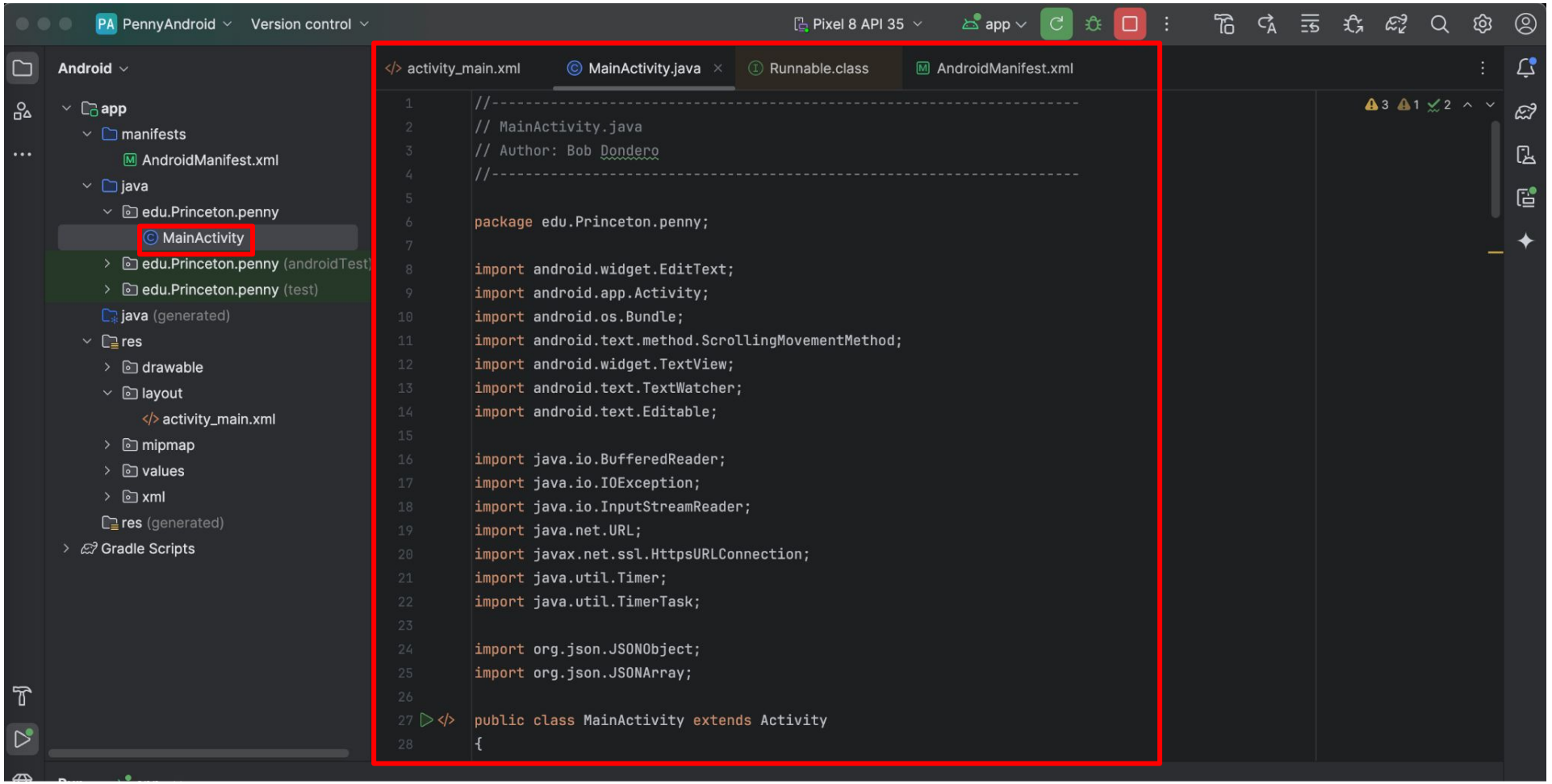


PennyAndroid App: Defining

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

PennyAndroid App: Defining

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



PennyAndroid App: Defining

- 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

PennyAndroid App: Defining

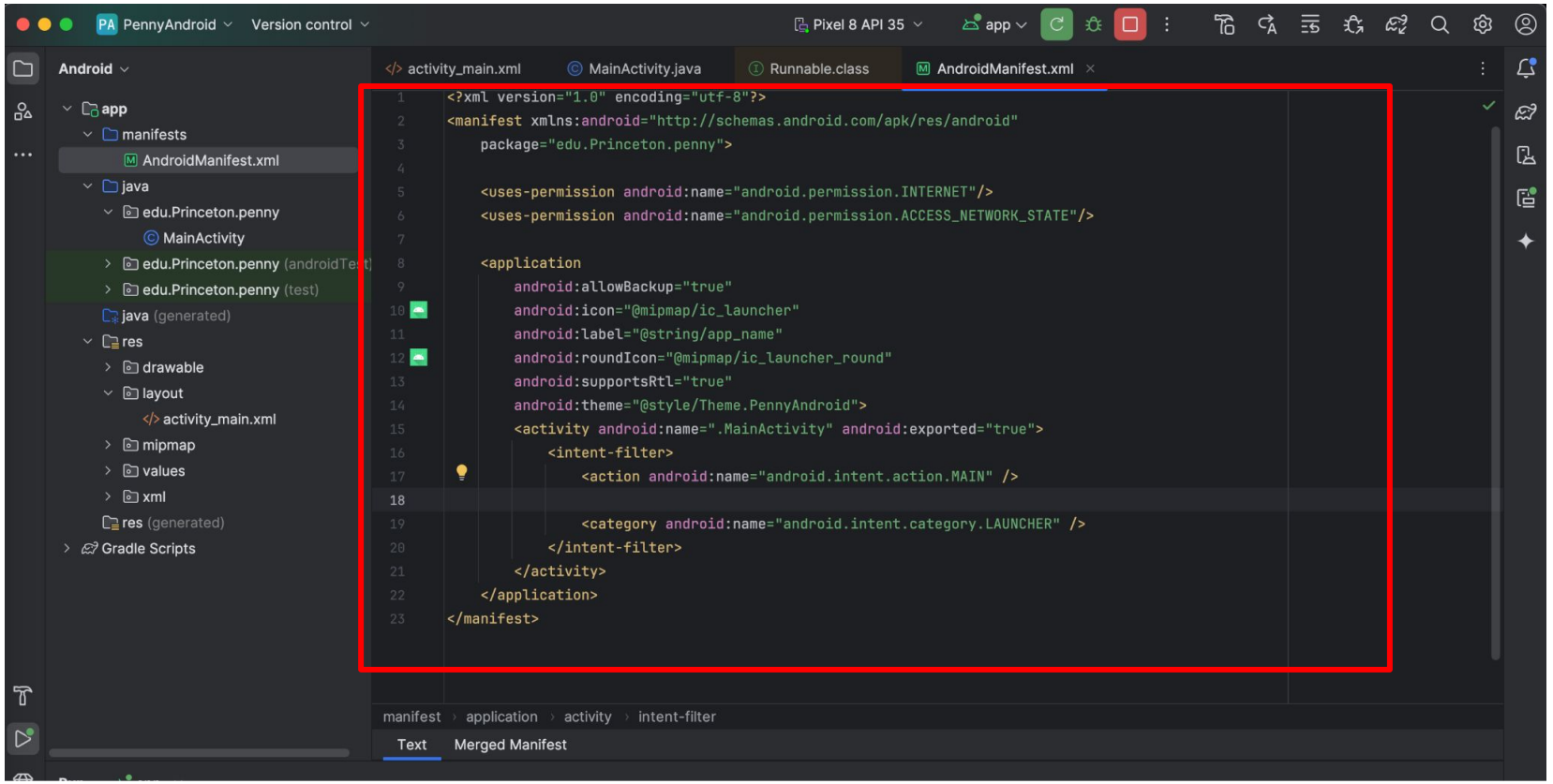
- See **MainActivity.java** (cont.)
  - **Android design constraint 2**
    - GUI thread must remain responsive
    - GUI thread laggy => typing/tapping fast generates "App is unresponsive" messages
    - Implications
      - GUI thread cannot wait for child/worker thread to finish
      - GUI thread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

PennyAndroid App: Defining

- 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

PennyAndroid App: Defining

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



PennyAndroid App: Defining

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

Agenda

- Mobile programming
- PennyAndroid app: defining
- **PennyAndroid app: running**

PennyAndroid App: Running

- To run an Android app…
- 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: Convenient

PennyAndroid App: Running

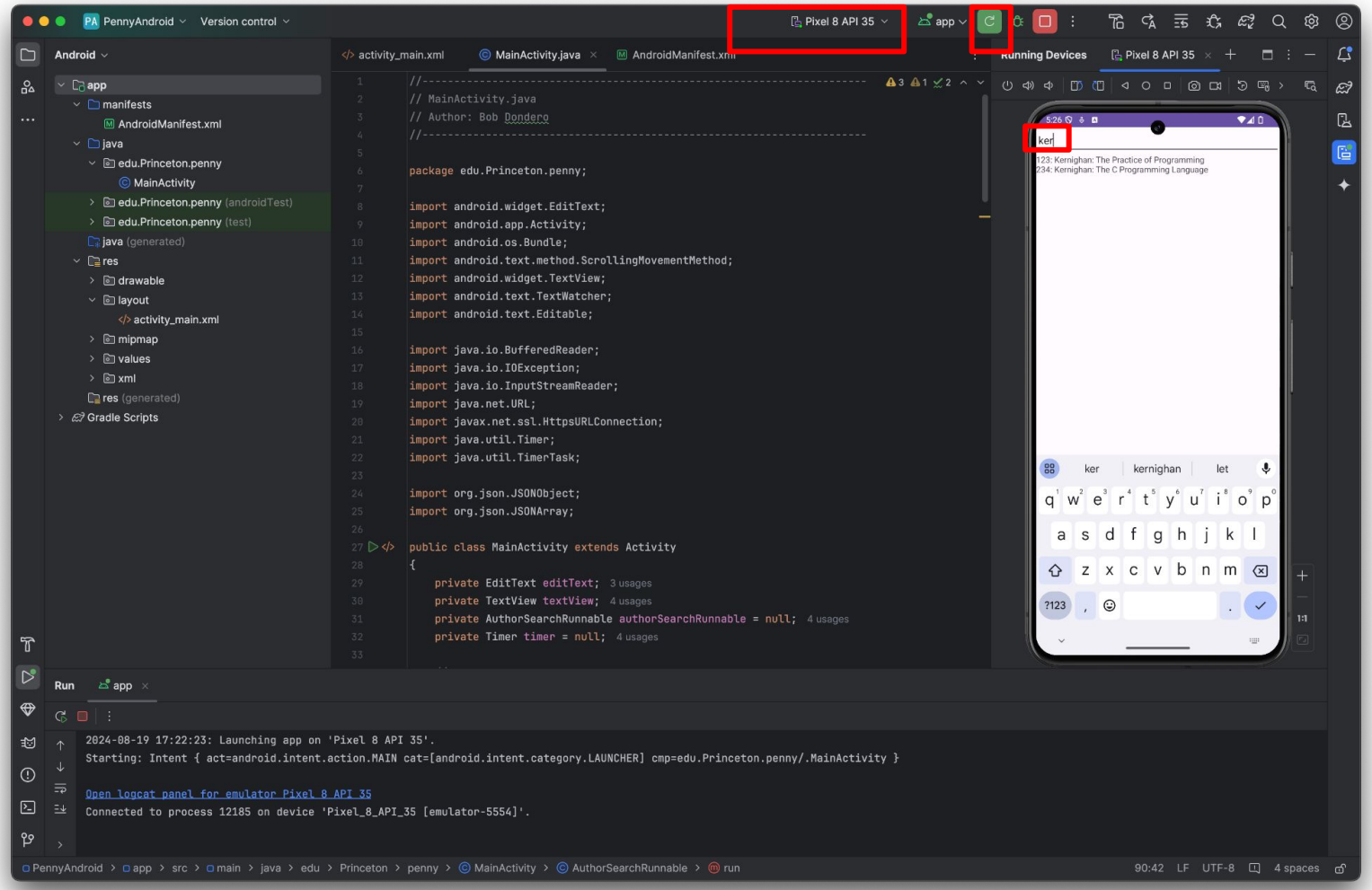
- To run on an Android emulator…

PennyAndroid App: Running

- 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*

PennyAndroid App: Running

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

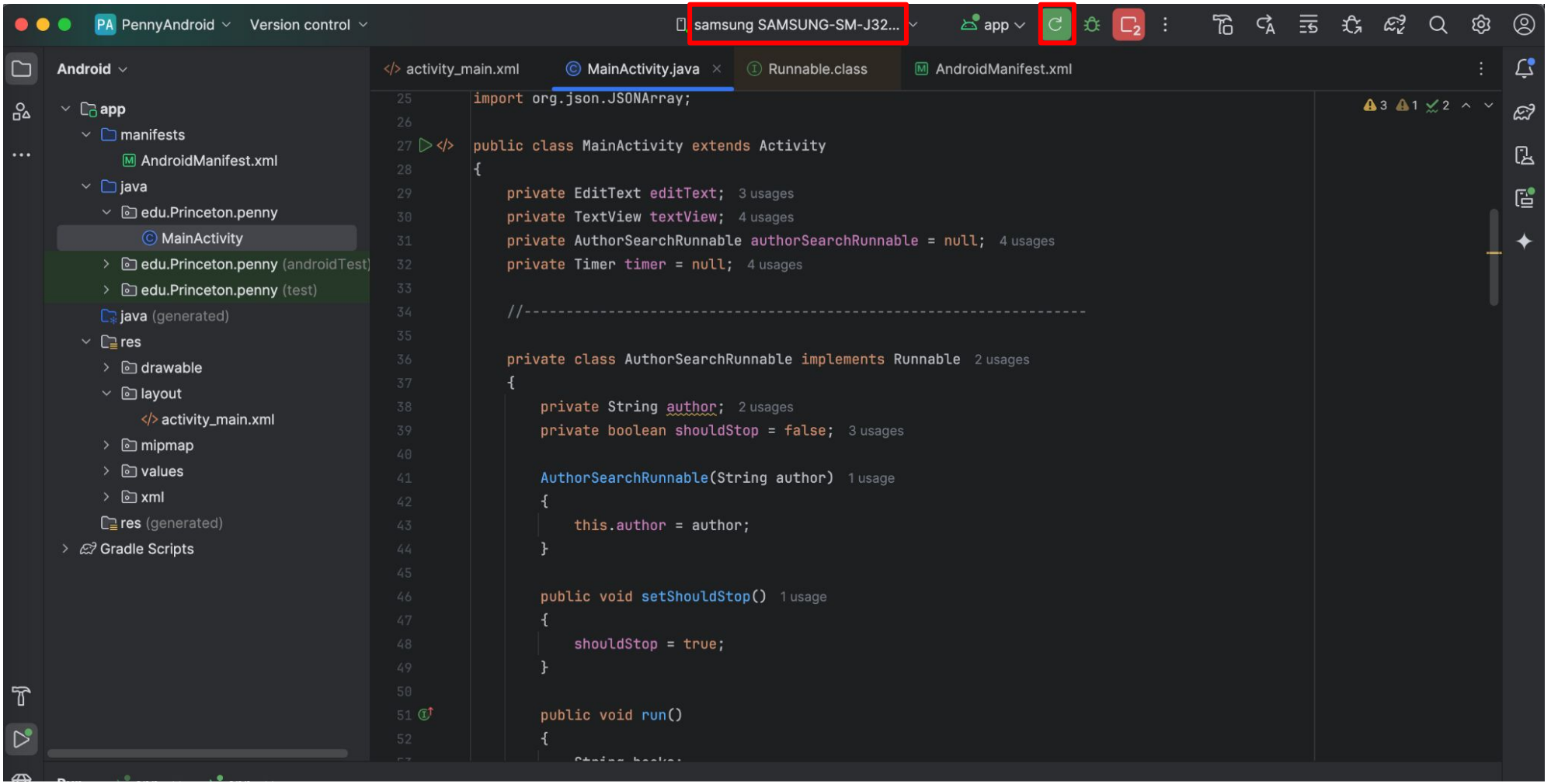


PennyAndroid App: Running

- To run on your Android phone…
  - Attach your Android phone to your computer's USB port
  - Respond to messages on your phone
    - Make sure your computer can access files stored on your phone

PennyAndroid App: Running

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



PennyAndroid App: Running

- To learn more about Android programming:
  - https://developer.android.com/guide

Objectives

So far:

Client	Browser + HTML + JavaScript Java + Swing Java + Android
Server	Python + Flask Java + Servlets Java + Spring JavaScript + Express

Summary

- We have covered:
  - Mobile programming
  - Android mobile programming

- See also:
  - **Appendix 1**: iOS Mobile Programming
  - **Appendix 2**: Cross-Platform Frameworks

Appendix 1:
iOS Mobile Programming

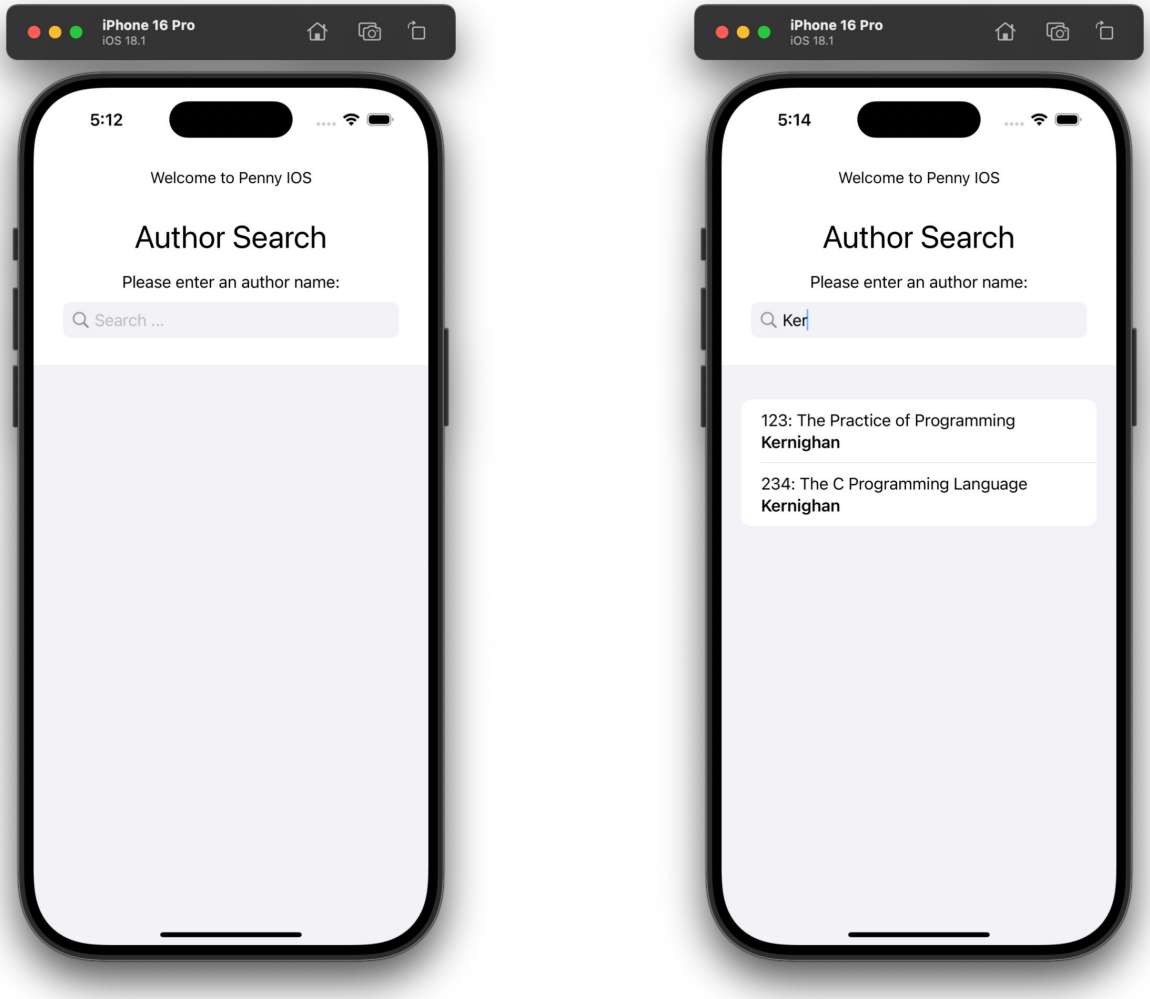
Thanks to
Katie DiPaola (‘26)...

Pennylos App: Defining

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

Pennylos App: Defining

The goal:



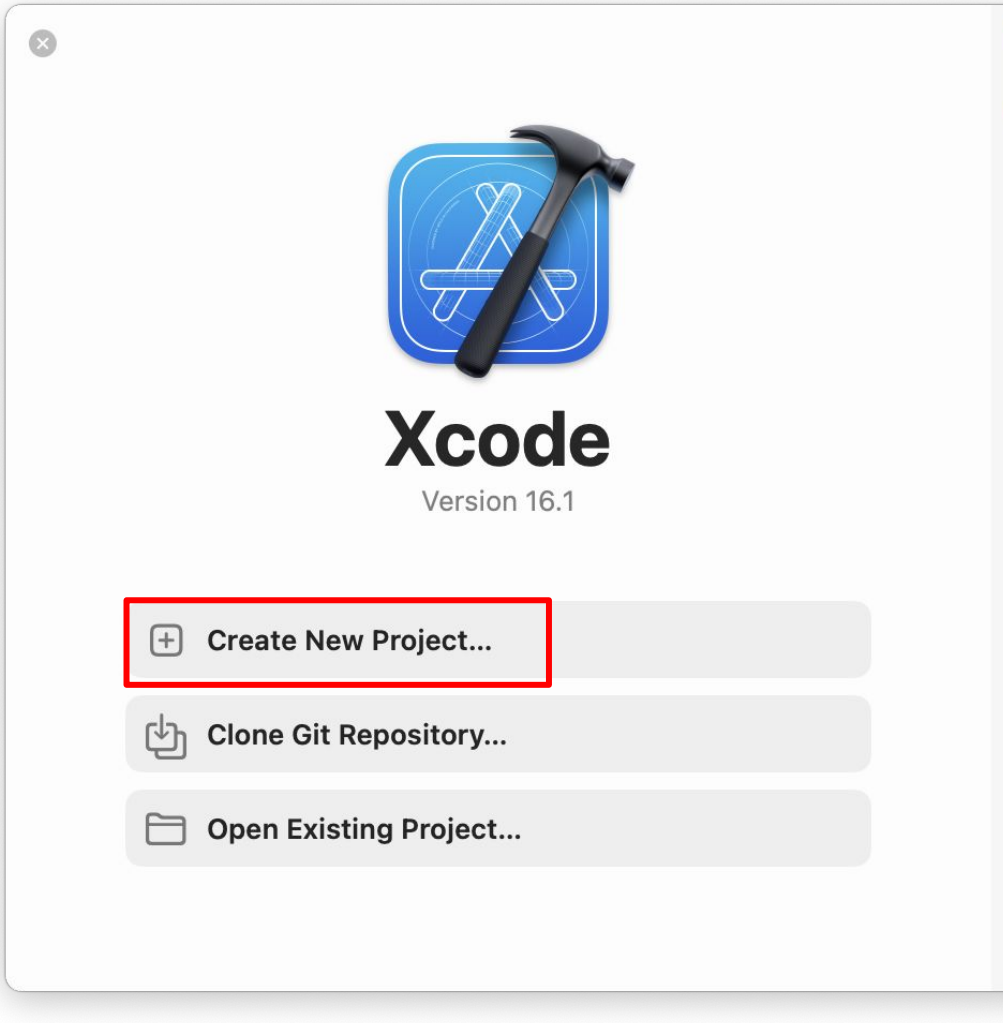
An iOS client for the PennyJson server

Pennylos App: Defining

- Download and install ***XCode***

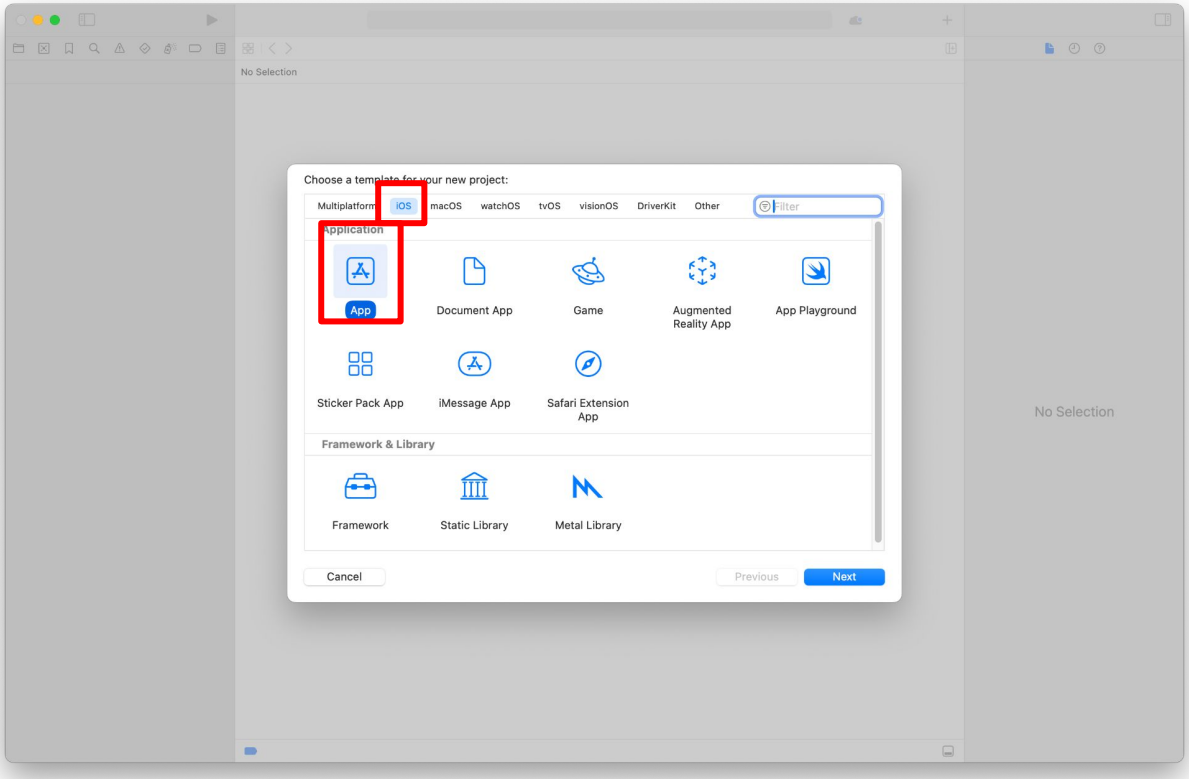
Pennylos App: Defining

Launch XCode; select *Create New Project…*



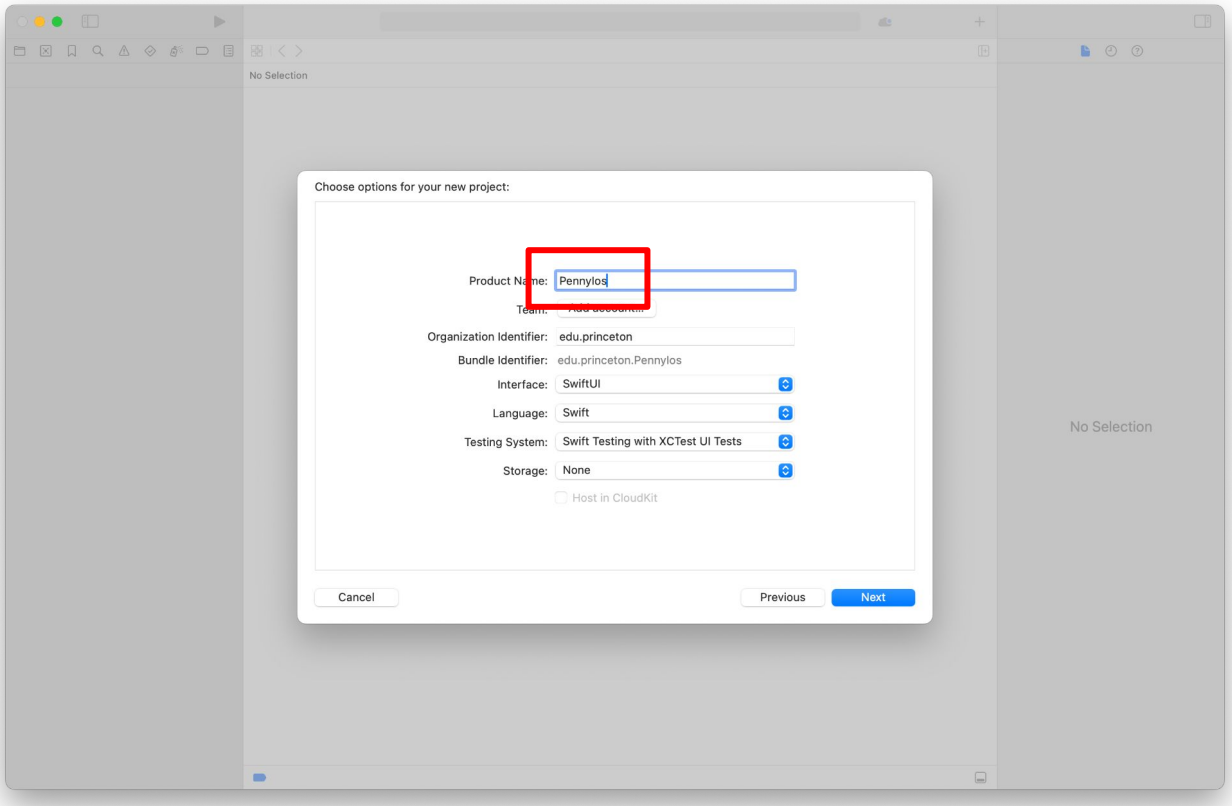
Pennylos App: Defining

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



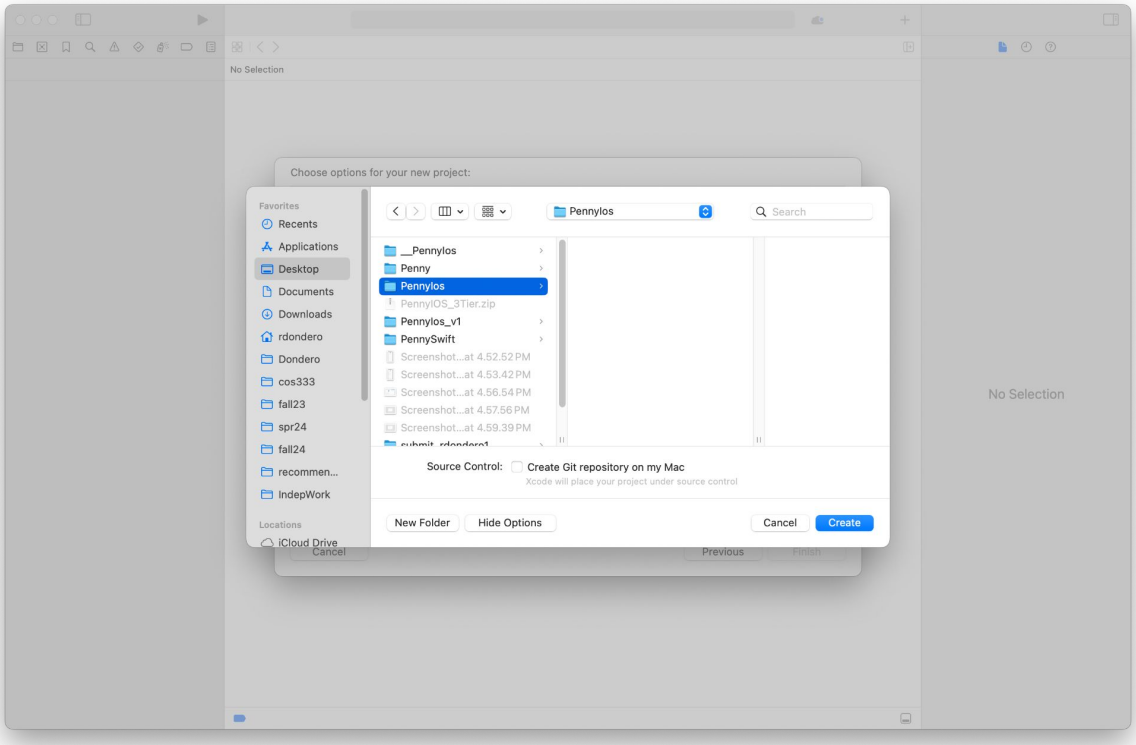
Pennylos App: Defining

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



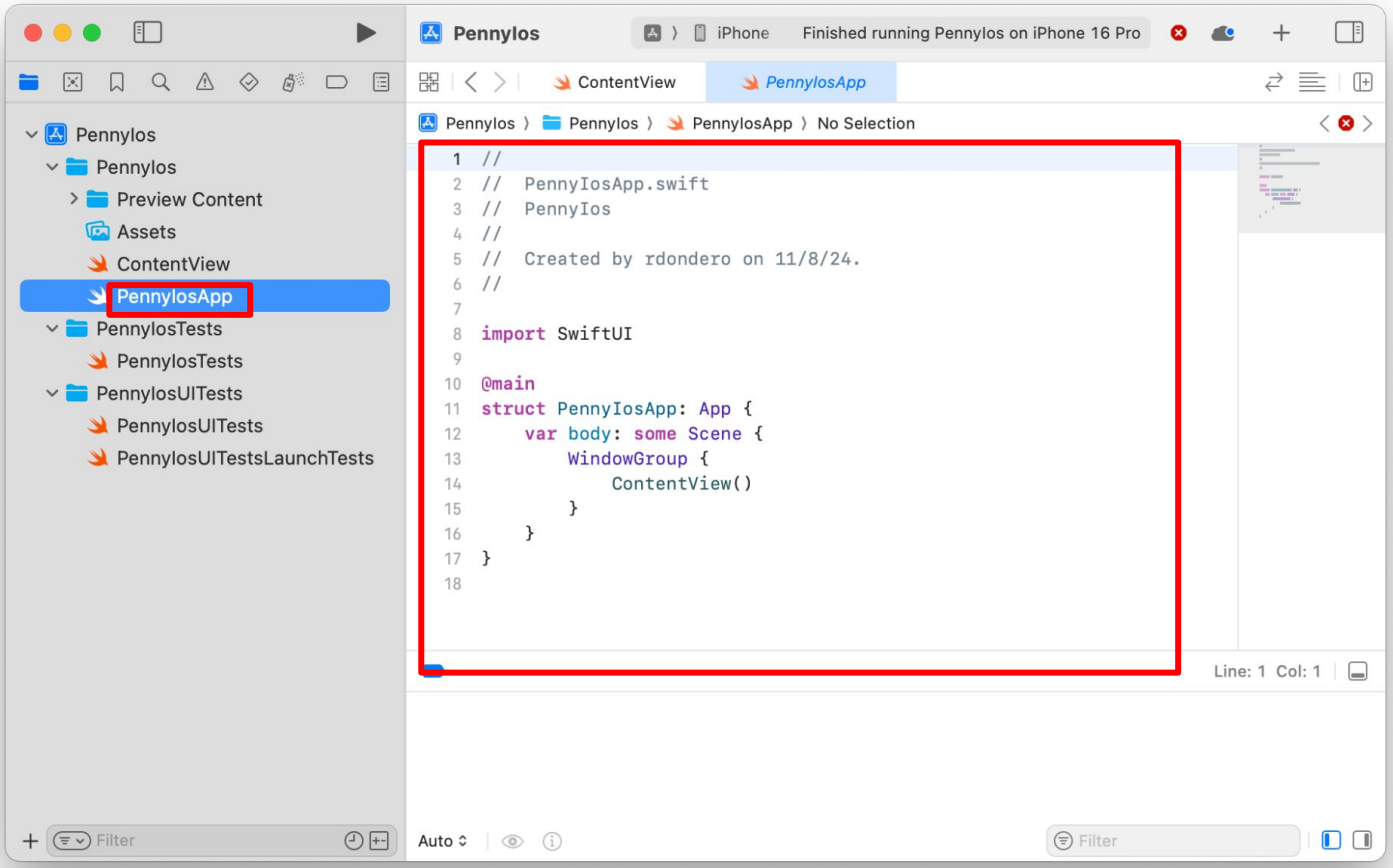
Pennylos App: Defining

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



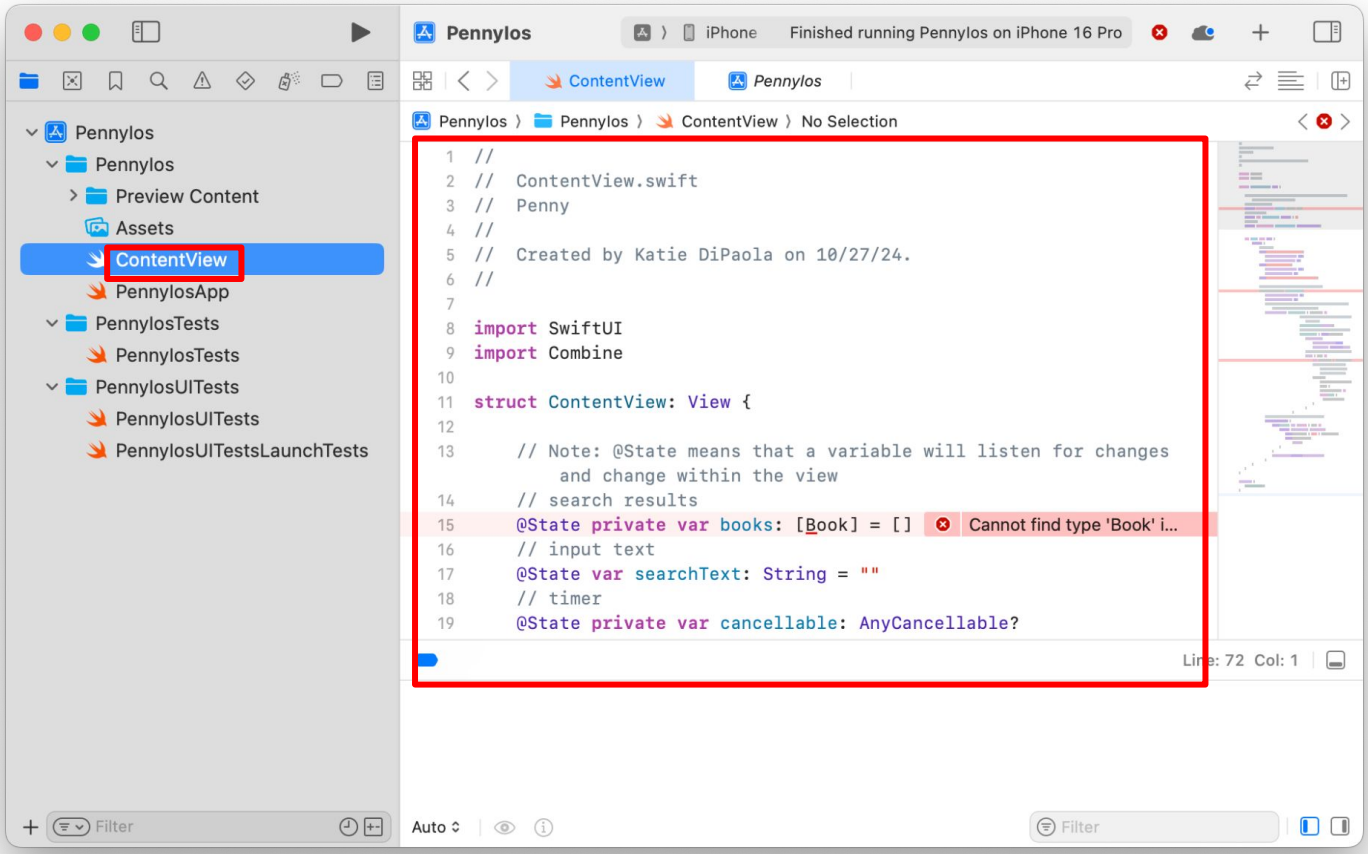
Pennylos App: Defining

Click on *PennylosApp*; examine the generated code



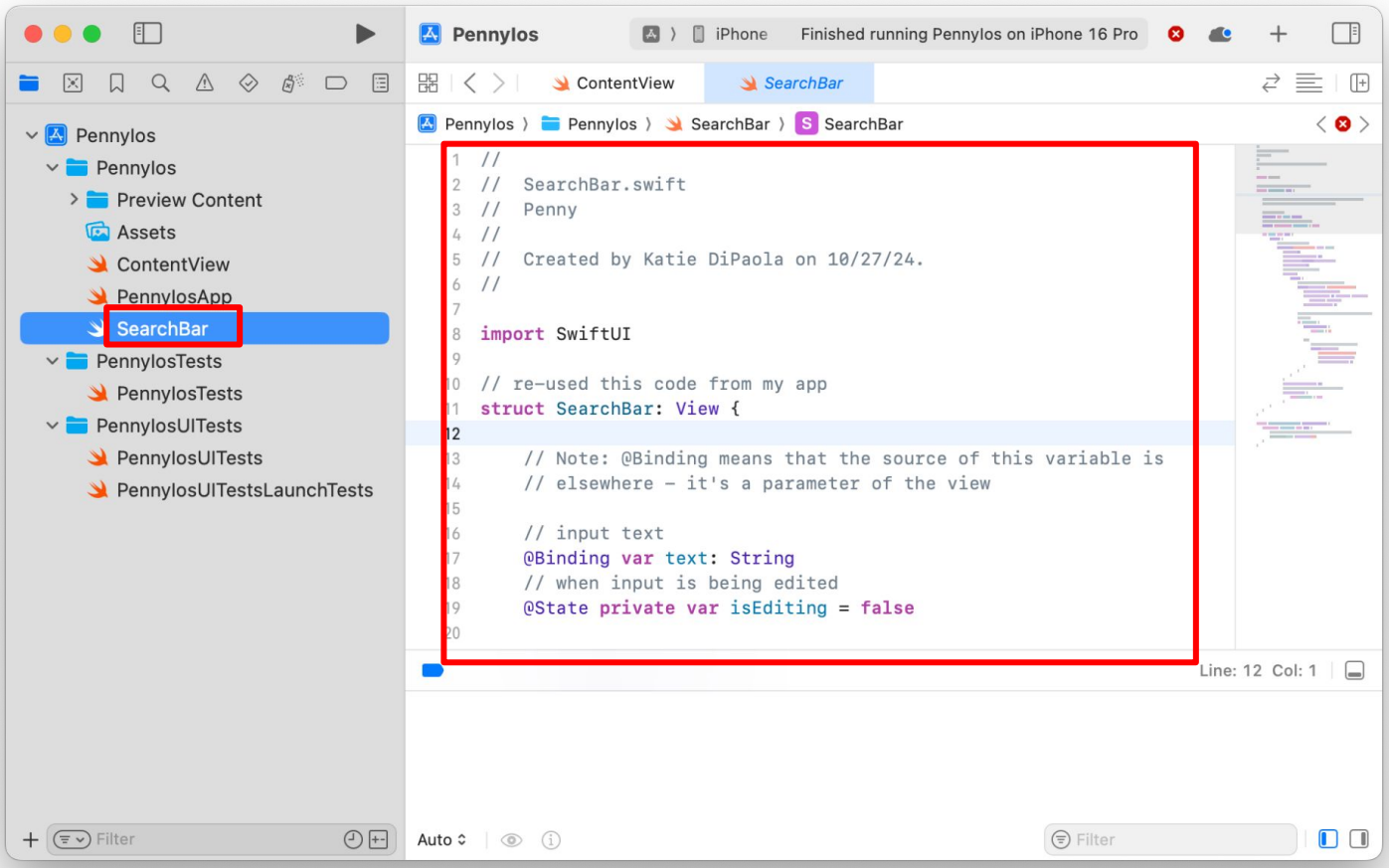
Pennylos App: Defining

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



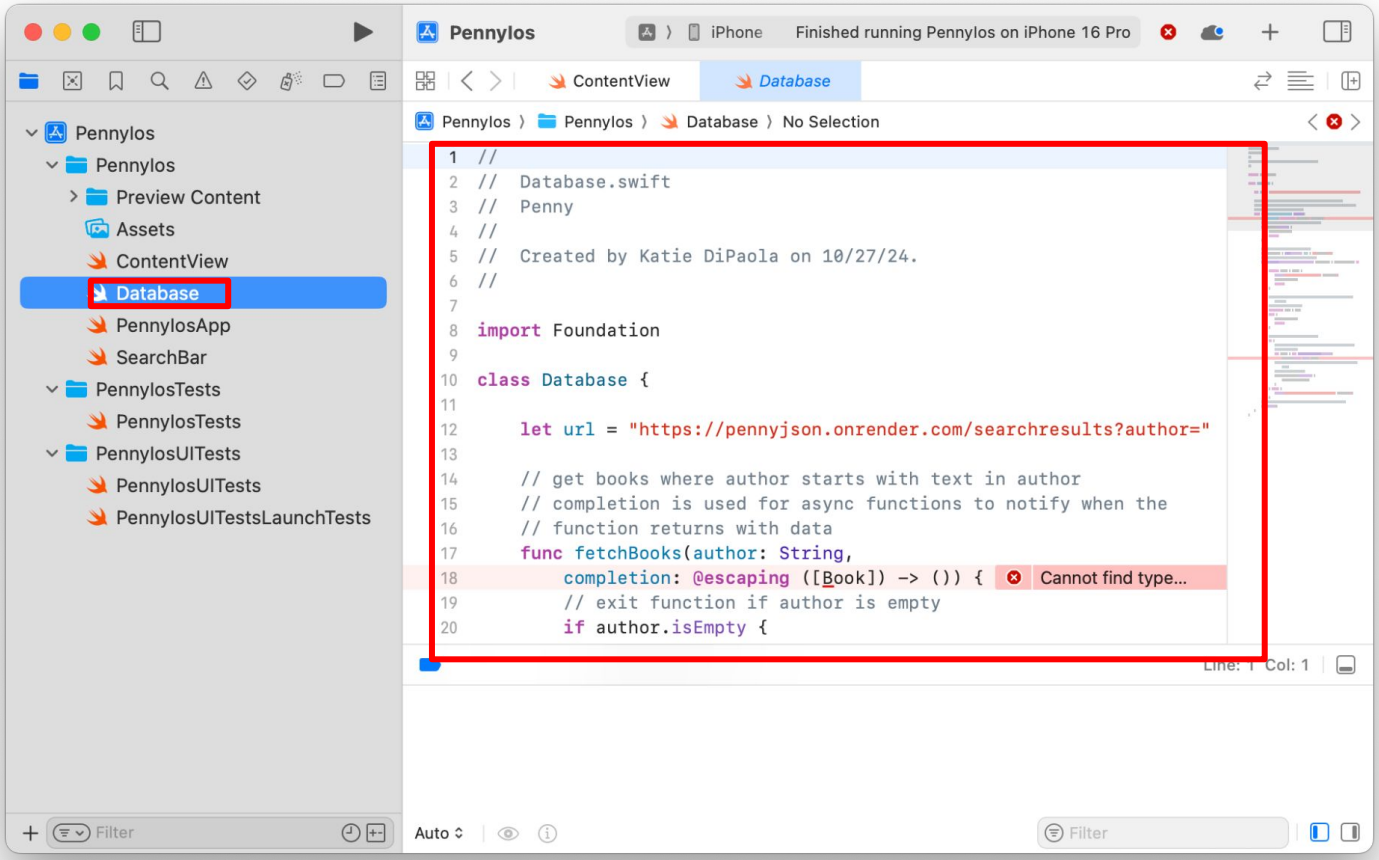
Pennylos App: Defining

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



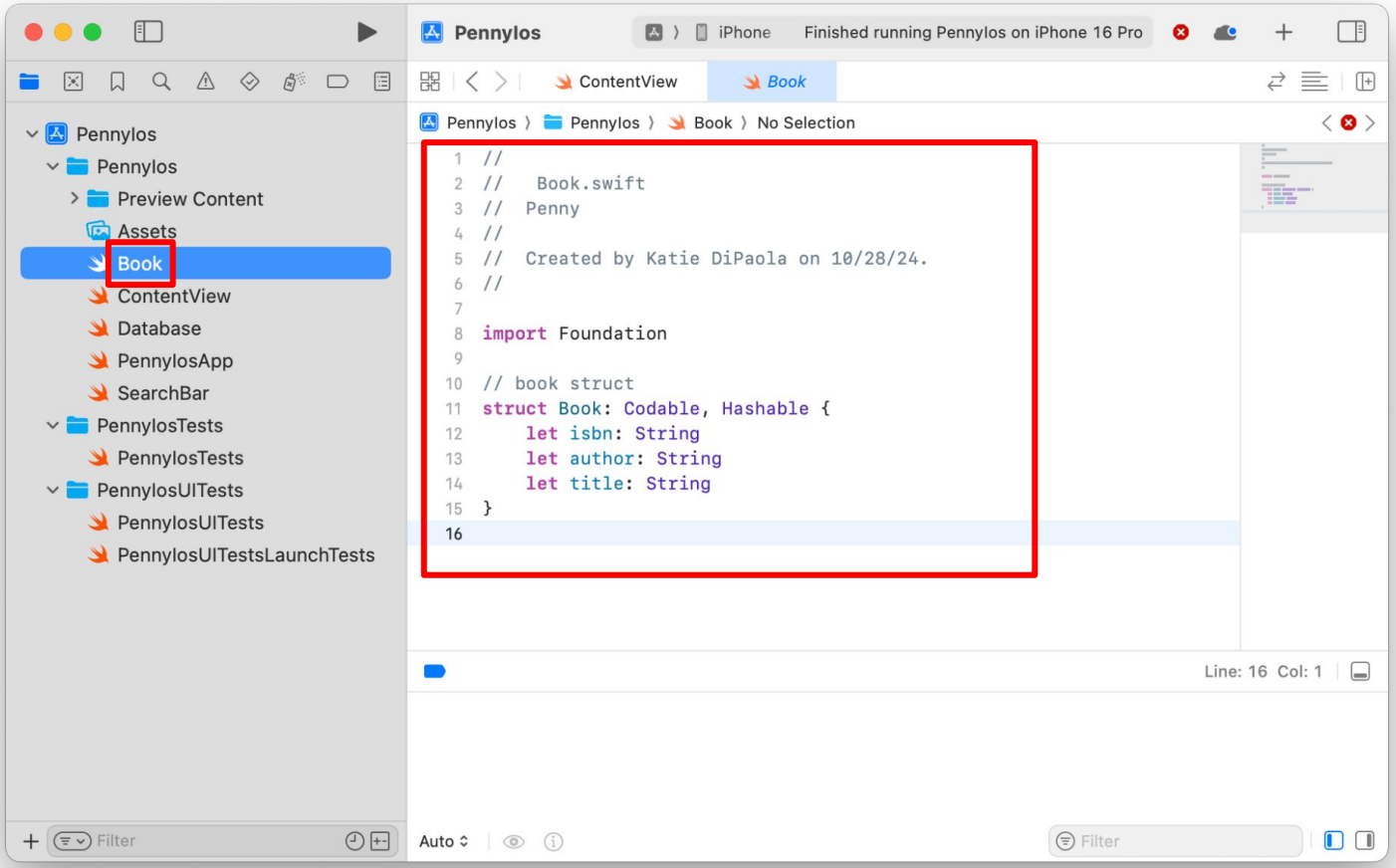
Pennylos App: Defining

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



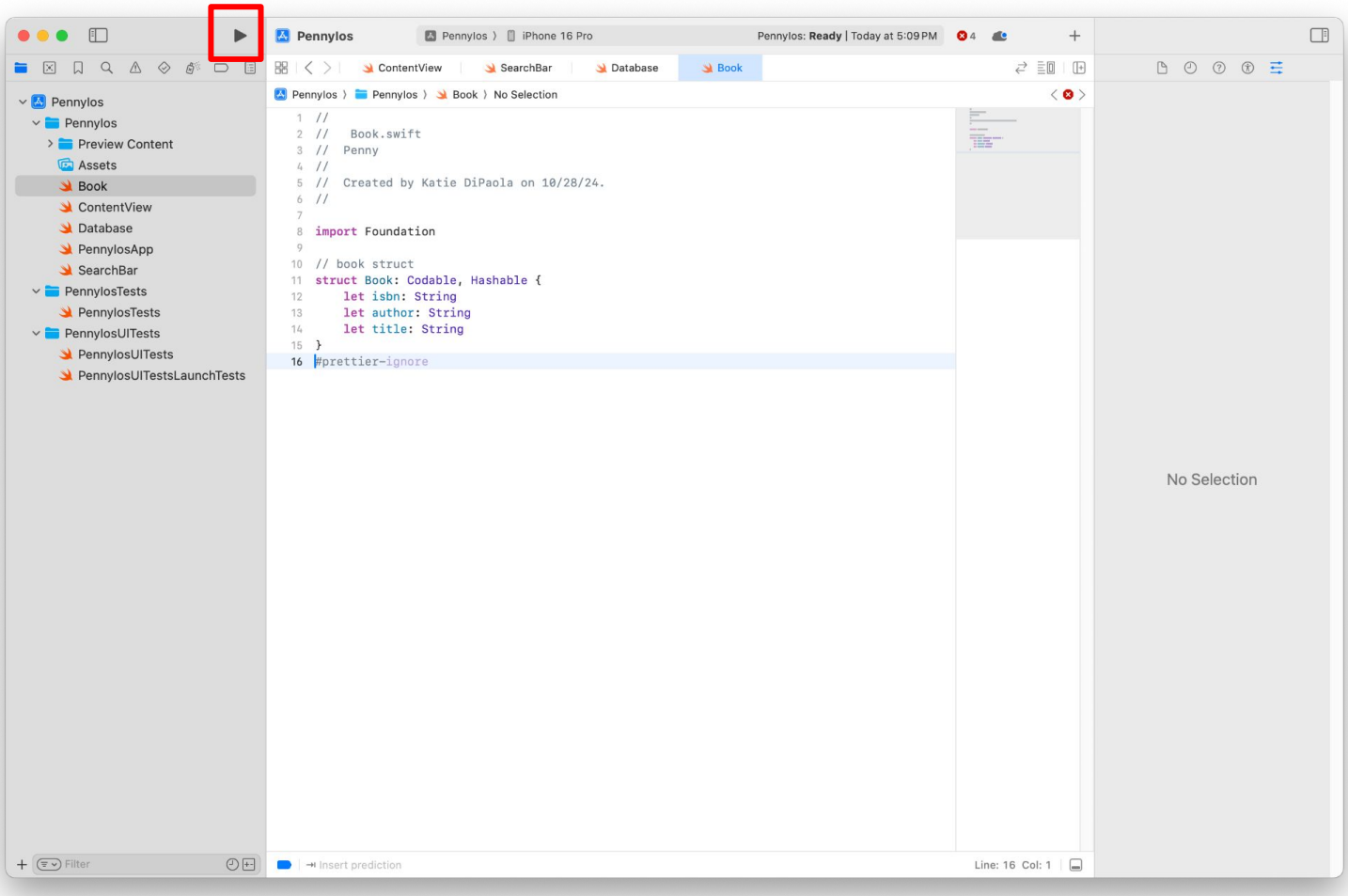
Pennylos App: Defining

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

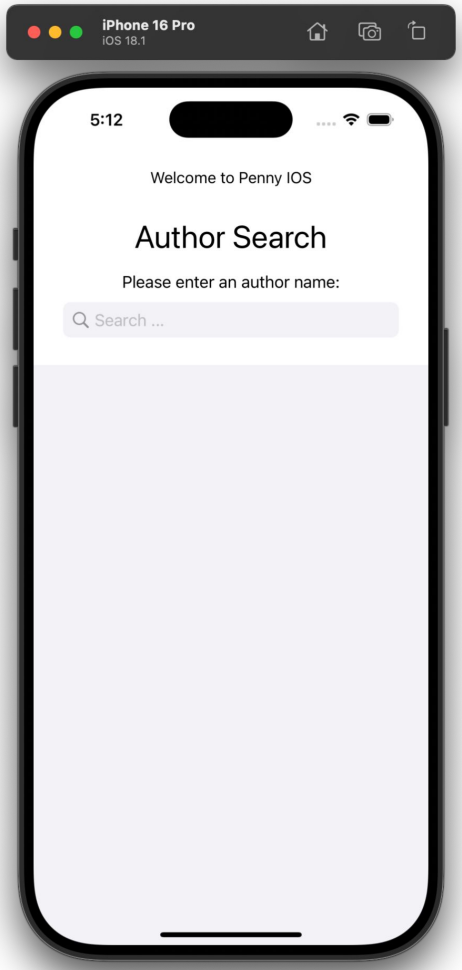


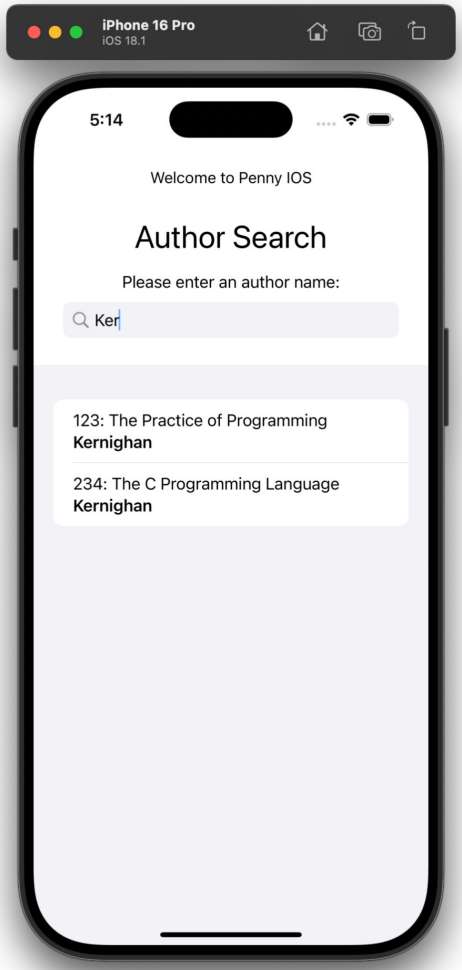
Pennylos App: Running

Click on the *Build/Run* button



Pennylos App: Running





Additional Resources

- Swift Basics Guide
  - https://guides.codepath.com/ios/Swift-Basics

- Uploading to the iOS App Store
  - https://christinesun.notion.site/christinesun/How-to-get-your-app-onto-the-iOS-app-store-018bcd0de6434878acb81c527f2efcb7

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

Objectives

So far:

Client	Browser + HTML + JavaScript Java + Swing Java + Android Swift + iOS
Server	Python + Flask Java + Servlets Java + Spring JavaScript + Express

Appendix 2:
Cross-Platform Frameworks

Cross-Platform Frameworks

- **Observations**:
  - A native **Android** app works only on **Android** devices
  - A native **iOS** app works only on **iOS** devices
- **Problem**:
  - Develop for one kind of device => limit users
  - Develop both kinds of devices => develop *& maintain* two code bases
- **Solution**:
  - Cross-platform frameworks

Cross-Platform Frameworks

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 is not **native**!