ANDROID PROGRAMMING: INTRODUCTION

Originals of Slides and Source Code for Examples:
http://www.coreservlets.com/android-tutorial/

OUTLINE

Approaches to develop mobile applications

- Browser Based
- Mobile Framework based
- Hybrid of above
- Cross Platform (Flutter, React Native, Kotlin Multi platform..)

Major OS’s

- Android
- iPhone

BROWSER BASED

- Pro
  - Universal access – just need browser
  - Always up to date - Content controlled by server
  - Many tools and technologies(advantage and disadvantage)
  - APIs for gaming which open up many app types (AR, training, etc)
- Con
  - Weak GUI widget set
  - Can’t interact with many local resources (accelerometer, gps, etc) or other devices
  - Can’t receive system notifications
  - Optimized for large screen and mouse – will work on smartphone but not well
  - Cant put on Appstores

MOBILE FRAMEWORK BASED - PRO

Many GUI controls

- Textfield, text area, button, checkbox, radio, list box, combo box, clock, calendar, date picker, dialog box, image gallery, etc.
    - Comparable to options in desktop programming
- Supports direct drawing
    - Animated games

Can interact with local resources

- Can read files (e.g., contacts list), have local database, access GPS, initiate phone calls, get input from microphone, create voice output, read screen orientation, etc.

MOBILE FRAMEWORK BASED - PRO

Efficient communication

- Can use any networking protocols you want

Easier (?) to write

- Requires knowledge of one language only
  - Kotlin/Java for Android
  - Swift for iPhone

Designed for small displays with touch screen

- So, many apps and GUI controls are optimized for this environment

MOBILE FRAMEWORK BASED - CON

No universal access

- Apps must be manually installed on each device
- An Android app cannot run on iPhone, PC, Mac, or Linux box

Difficult to manage updates

- User must intervene to get latest versions

MUST DEVELOP SAME APP FOR EVERY OS

at least 2 dev environments (android, IOS) ,

multiple codebases to maintain

(Could try Flutter or React Native)

HYBRID

Most of functionality hosted in web pages on web server

Build minimal native apps that host web browser views

Have minimal native code to create for each platform

Web pages are updated instantly

This really eases multiplatform dev cycle.

Cross Platform (Flutter, React Native, Kotlin Multi platform..)

Cross Platform

- Write once deploy to many

New Language to learn (check Tiobe software index)

- Flutter – Dart
- React Native – Javascript
- Kotlin Multiplatform (KMP) - Kotlin

But...

Flutter – Big layoffs in Team last May

Kotlin Multiplatform (KMP) – Still evolving

You still need knowledge of the underlying architecture to effectively use any of these.

SUMMARY

Web apps vs. Android apps

- Web apps can run on Android, iPhone and regular computers. But, they have weaker GUIs, cannot use some local resources (files, databases, GPS, camera), are often ill-suited to small screens, require learning many technologies
- Native apps can access local resources, are optimized for small screens, have richer GUIs, but are not cross platform so require multiple solution codebases. Also difficult to update
- Hybrids are good compromise if you need multiplatform support, simplifies and reduces development cycle. Get auto updates for web based portion.