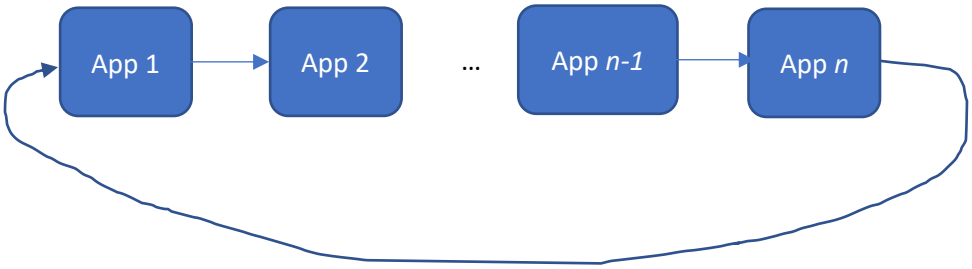
CS 1699 Special Topics in Computer Science
Introduction to Android Programming
(Section 1115)
Spring 2018

Team Project – The Big App

The idea of the team project is to build a big application by connecting apps that are built by the individual teams. While each team builds their own application. The team apps form a cycle, based on team number. Each team's app receives *triggering events* from the previous app in the cycle and delivers triggering events to the next app. Each team defines the triggering events of their app, commits to them, and publishes them so they become available to the previous team.

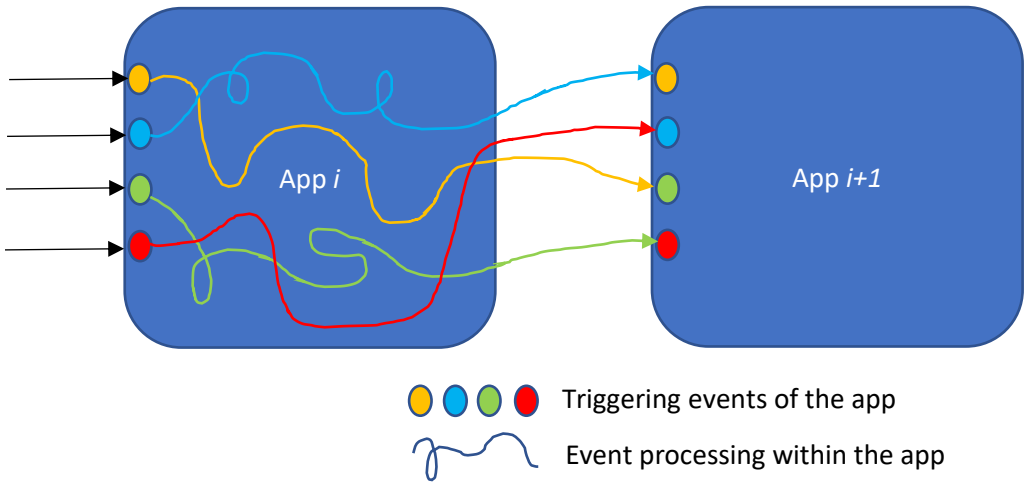


Each team is free to choose the idea of their app under the following constraints:

1. The app has a front-end that interacts with the user(s) and a back-end that manages the app data.
2. The app has a set of at least four well-defined triggering events. These events are external to the app, in the sense that they are not caused by direct user interaction or direct processing by the app. These events are triggered by the previous app in the cycle.
   Example 1 (app 1): Let's take as an example an app that helps its users navigate the public bus system. External triggering events for this app may be:
   a. Adding or changing a bus route.
   b. The arrival of a bus at a bus stop.
   c. The arrival of a passenger at a bus stop.
   d. Severe weather conditions at some geographical area.
   Example 2 (app 2): Let's have another example. Consider an app that helps its users order some sort of merchandise from a store chain. External triggering events for this app may be:
   e. Changing the list of merchandise (e.g., the menu).
   f. The arrival of the user to or near by the store.
   g. The opening of a new store branch.
   h. Modifying of user preferences that are related to the merchandise.

As an example of connecting the two example apps, consider the processing of event d by the first app. The severe weather conditions event leads app 1 to ask its user if she wants to get some hot coffee while waiting for the delayed bus, which would then trigger the modification of the user's preferences (event h of app 2). Of course, the processing of event d in app 1 should include other tasks as deemed appropriate by the team developing app 1. Similarly, you may think of scenarios that connect event c of app 1 to event f of app 2, event a of app 1 to event g of app 2, and event b of app 1 to event e of app 2. (The last connection may need some more imagination though. If App 1 maintains information about the Ads posted around bus stops, it may assume that the user will see a specific Ad and will then trigger an event to change the menu to reflect a potential user interest.) App 1 is cooperating with and is explicitly aware of the existence of app 2.

Although these events may be generated by the app itself, the app outsources the triggering of these events to the previous app in the cycle. The app has, however, to respond to each of these events. The processing of each event has to lead into the triggering of an event to the next app. **An 1-1 mapping between input events and output events has to be defined and implemented.** Negotiation between the teams is allowed and encouraged to reach such a mapping. The processing of each event has to end within reasonable time so that control is handed over to the next app by triggering one of its events.



3. At least one of each of the following components has to be implemented in each app:
    a. Activity
    b. Service
    c. Broadcast Receiver
    d. Content Provider
4. Each application has to control access to its data and events as reasonably needed. Access control can depend on user login and API keys. The API keys have to be provided to the team developing the previous app.
5. Each processing path has to touch at least two different app components. This is to be shown in the sequence diagrams delivered in the APIs and Mock-ups phase (please see the table below).
6. Each app is allowed to interact with the user only within the processing of its triggering events. Background services may be continuously run though. The only chance an app can bring one of its activities to the device foreground is when the app is processing an event. The app assumes

that it will lose foreground status once it hands over to the next app by triggering one of the next app's events.

7. Each app has to define the mechanisms of the triggering of its input events. Each event has exactly one triggering mechanism and overall these mechanisms have to include all of the following:
    a. An Intent sent to an Activity
    b. An IPC with a Service
    c. An event sent to a Broadcast Receiver
8. Event triggering mechanisms have to specify any needed **data** that need to be sent along with the trigger. The data schema and semantics have to be defined clearly. For example, an JSON schema of user preferences along with the meaning of each element would have to be specified along with event d. of Example 2 above.
9. App data are managed both locally and using a public cloud.
10. Appropriate privacy preservation should be implemented; for example, before "leaking" user's data from one app to the next, user permission is to be requested.
11. Each team has to provide a testing tool that triggers the input events based on user selection.

Project Deliverables and Deadlines

Deliverable/Phase	Date and time	Grade Percentage
Response to **Team Formation** Survey	02/05 @11:59pm	5%
Teams are formed ***by the instructor*** and announced	02/07 @11:59pm	
APIs and Mock-ups (on CourseWeb) • Complete specification of the triggering events and their triggering mechanisms • Screen mock-ups and sequence diagrams of the event processing paths	03/12 @11:59pm	20%
First team presentation • Presentation of the APIs and mockups	03/13 in class	10%
First app submission (on CourseWeb) • Source code and signed APK file of the app • Team-built testing tool for triggering the input app events	~~03/26~~ 4/2 @ 11:59pm	10%
First integration party • Each team presents a demo of their app • All apps running on the same device • App 1 triggered by Team 1's testing tool then a single cycle through the apps is executed • Testing tools are used when an app fails to deliver events to the next app	~~03/27~~ 4/3 in class	5%
Final app submission (on CourseWeb) • Source code and signed APK file • Team-built testing tool for triggering the input app events	04/16 @11:59pm	10%

Final integration party • First, each team presents their app separately using their testing tool • Then, all apps are tested by running them on the same device for at least four cycles through all the apps • In the first cycle, App 1 is triggered by Team 1's testing tool then no further intervention from the testing tools unless an app fails to deliver events to the next app.	04/17 in class	• 30% on separate app • 10% on app integration

Code submissions will be graded based on successful execution of the app and adherence to the app restrictions. Presentations are graded based on clarity, adherence to time limits (to be announced) and overall organization. Document deliverables (mock-ups, event specifications, and diagrams) are graded based on clarity and adherence to the app restrictions. 5% of the grade will be based on peer ranking surveys that will be sent during the term.