GitHub Username: robbiestells

The Commentist

Description

The Commentist is a network of podcasts with tens of thousands of listeners every week. The network currently has four active podcasts:

- Roll to Hit - a live play, 5th edition Dungeons and Dragons podcast.
- The Bearded Vegans - news, reviews and interviews related to all things vegan.
- The Unwind - discussing topics related to technology, games, gadgets, and geek culture.
- Sky on Fire - a live play RPG set in the Star Wars universe.

The Commentist app will allow fans to stay up to date on all the latest episodes of their favorite shows. Users of the app will be able to stream all of the available episodes for these shows with a unique player, as well as view the description for each episode and see the hosts. Users will receive notifications when new episodes are available to stream.

Intended User

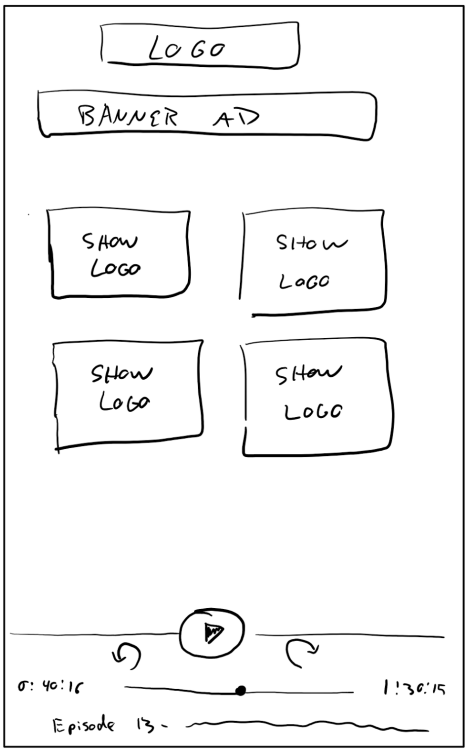
This app is intended for fans and listeners of The Commentist shows or people looking to find a new set of podcasts to listen to.

Features

- View all current shows on The Commentist
- View all episodes of each show
- View a description and the hosts of each show
- Stream episodes of the shows
- View episode descriptions
- Save episode information for quick loading
- Offer free version with ads or paid version without ads
- Receive notifications of new episodes
- Keep listening to the episode in the background when other apps are opened through a player in the notification panel

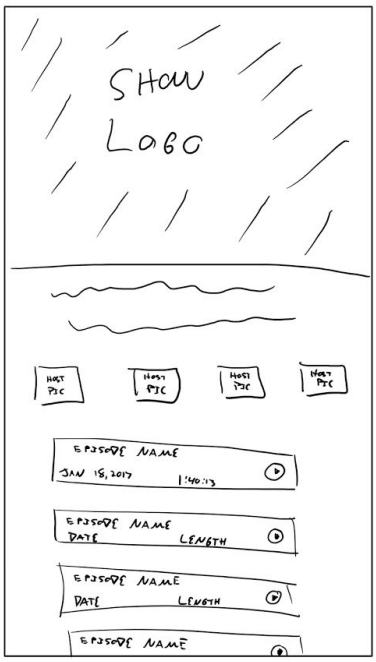
User Interface Mocks

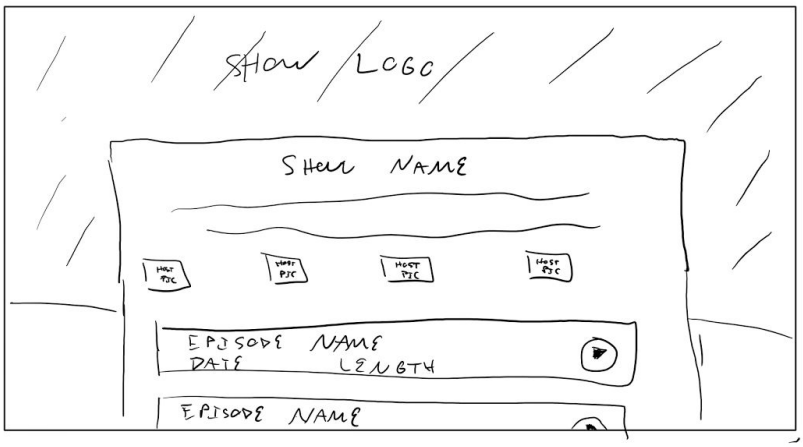
Screen 1 - Show List



Launch screen shows The Commentist logo on top, with banner ad (on the free version) underneath. The show logos will be displayed in a GridView. Once an episode is played, the Player will appear on the bottom of the screen and persist across all screens. The player will include a play/pause button, skip forward/backwards 10 seconds, a seekbar, the current time and total time, and the episode title. Sliding the player up will display the episode description. Clicking on a show logo will take users to the Show Details page.

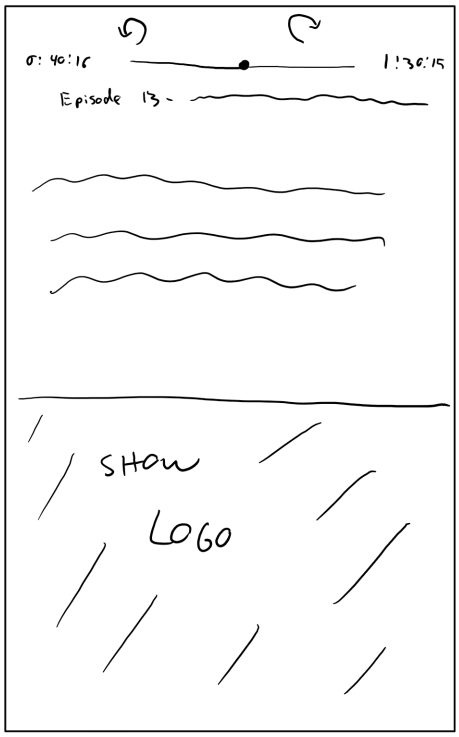
Screen 2 - Show Details





The Show details page will display the show logo, the description of the show, the host images, and a list of all of the episodes available. Each episode will be in a cardView, with the episode title, date published, length, and a button to start playing the episode. Playing the episode will trigger the Player to appear if nothing is playing, otherwise, it will replace the currently loaded episode and start playing the selected episode. On tablets, the main information on the screen will be displayed on a sheet with a max width set so that the content doesn't get too stretched out.

Screen 3 - Player Expanded



Expanding the Player displays the episode information, including the episode name, date published, length, episode description, and the show logo.

Key Considerations

How will your app handle data persistence?

The app will use an AyncTask to retrieve the rss feed for each show and save the data to a ContentProvider, which will store data in a SQLite database. On the first launch of the app on a new device, the app will save all of the available episode data. This data populates the Show page to display the cards for each episode. Everytime the app is launched, the AsyncTask will run to look for new episodes and add them to the local database.

Describe any corner cases in the UX.

Since the MediaPlayer will persist across all screens through the Player in a BottomSheet, the user can return to the current episode information by expanding the Player.

Describe any libraries you'll be using and share your reasoning for including them.
I'll use the ButterKnife library to reduce the amount of code needed to bind views.

Describe how you will implement Google Play Services.
This app will utilize the Admob and Notifications from Google Play Services. Admob will be used to display a banner ad on the home screen of the free version of the app. Notifications will be used to allow the app administrator to send out notifications to users, alerting them when new episodes are released.

Next Steps: Required Tasks

Task 1: Project Setup
During the setup phase, I will create the AsyncTask to retrieve the rss feed episodes and the Content Provider to store that information. I'll be creating objects for Episodes and Shows, so that they can be stored within the ContentProvider. This includes setting up the SQLite database structure, the SQLiteOpenHelper, and the ContentProvider.

Task 2: Implement UI for Each Activity and Fragment
During the UI phase, I'll be creating the UI for the main page which contains the app Logo, a Frame to load the fragments, and a BottomSheet with the Player. I'll also create the fragments for the Show grid and the Show page with a listview of episodes. From initial testing, I determined that the best way to have the player be persistent across all screens was to have the player be contained in a BottomSheet on the MainPage, with the Show grid and Show page loaded as fragments on the MainPage.

Task 3: Create the Player Service
During Task 3, I'll hook up the Player as a service. This will allow me to access it from within the fragments, as well as the Player on the MainPage. The player will be hooked up so that it can play/pause the current track, skip forward/backwards in 10 second intervals, and use the SeekBar to skip throughout the episode.

Task 4: Add Google Play Services, Flavors
I'll set up the Admob and Notification services from Google Play. The Notifications will be used to send out updates to users about new episodes. Two flavors, paid and free, will be created, with Admob being added to display an advertising banner on the MainPage of the free version.

Task 5: Background Audio/Notification Player

The final task will be to allow the Player to continuing playing from the background of the phone if the app is suspended. Player controls will be added to a notification item on the phone so that the user can access them from other apps.