# Enabling Your App for CarPlay

Session 719

Albert Wan, CarPlay Engineering

## Agenda

How apps run in CarPlay

CarPlay app integration

## How Apps Run in CarPlay

## Features Available to All Apps in CarPlay

Now Playing information and playback controls

Audio plays through car’s speakers

SiriKit (for appropriate intents)

## Best Practices for All Apps

Don’t play audio automatically unless requested

Configure the audio session

## Best Practices for All Apps

Don’t play audio automatically unless requested

Configure the audio session

Spoken audio apps

- `AVAudioSessionModeSpokenAudio`

## Best Practices for All Apps

Don’t play audio automatically unless requested

Configure the audio session

Spoken audio apps

- `AVAudioSessionModeSpokenAudio`

Navigation apps

- `AVAudioSessionModeSpokenAudio`
- `AVAudioSessionCategoryOptionInterruptSpokenAudioAndMixWithOthers`
- Set mixable before making audio active

## CarPlay Enabled Apps

Appear on the CarPlay home screen

Run in the foreground on both screens

## CarPlay Enabled Apps

Supported app categories

- Audio apps
- Messaging and VoIP calling apps
- Automaker apps

Request CarPlay entitlement
http://developer.apple.com/carplay

Entitlement setup and provisioning

## Designing for CarPlay

For icon guidance, see “App Icon” in “iOS Human Interface Guidelines”

For design guidelines, see “CarPlay Human Interface Guidelines”

## Xcode and Simulator

Audio limitations (playback state)

Test on device and with an actual head unit

Wireless debugging in Xcode

## Data Protection

Apps can run in CarPlay while the iPhone is passcode locked

Passcode protected data may be unavailable

## Data Protection

Apps can run in CarPlay while the iPhone is passcode locked

Passcode protected data may be unavailable

| | |
|---|---|
| Files | `FileProtectionType.complete`<br>`FileProtectionType.completeUnlessOpen` |

## Data Protection

Apps can run in CarPlay while the iPhone is passcode locked

Passcode protected data may be unavailable

| | |
|---|---|
| Files | `FileProtectionType.complete`<br>`FileProtectionType.completeUnlessOpen` |
| Keychain | `kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly`<br>`kSecAttrAccessibleWhenUnlocked`<br>`kSecAttrAccessibleWhenUnlockedThisDeviceOnly` |

## Data Protection

Apps can run in CarPlay while the iPhone is passcode locked

Passcode protected data may be unavailable

| | |
|---|---|
| Files | `FileProtectionType.complete`<br>`FileProtectionType.completeUnlessOpen` |
| Keychain | `kSecAttrAccessibleWhenPasscodeSetThisDeviceOnly`<br>`kSecAttrAccessibleWhenUnlocked`<br>`kSecAttrAccessibleWhenUnlockedThisDeviceOnly` |
| SQLite | `SQLITE_OPEN_FILEPROTECTION_COMPLETE`<br>`SQLITE_OPEN_FILEPROTECTION_COMPLETEUNLESSOPEN` |

## CarPlay App Integration

## CarPlay App Integration

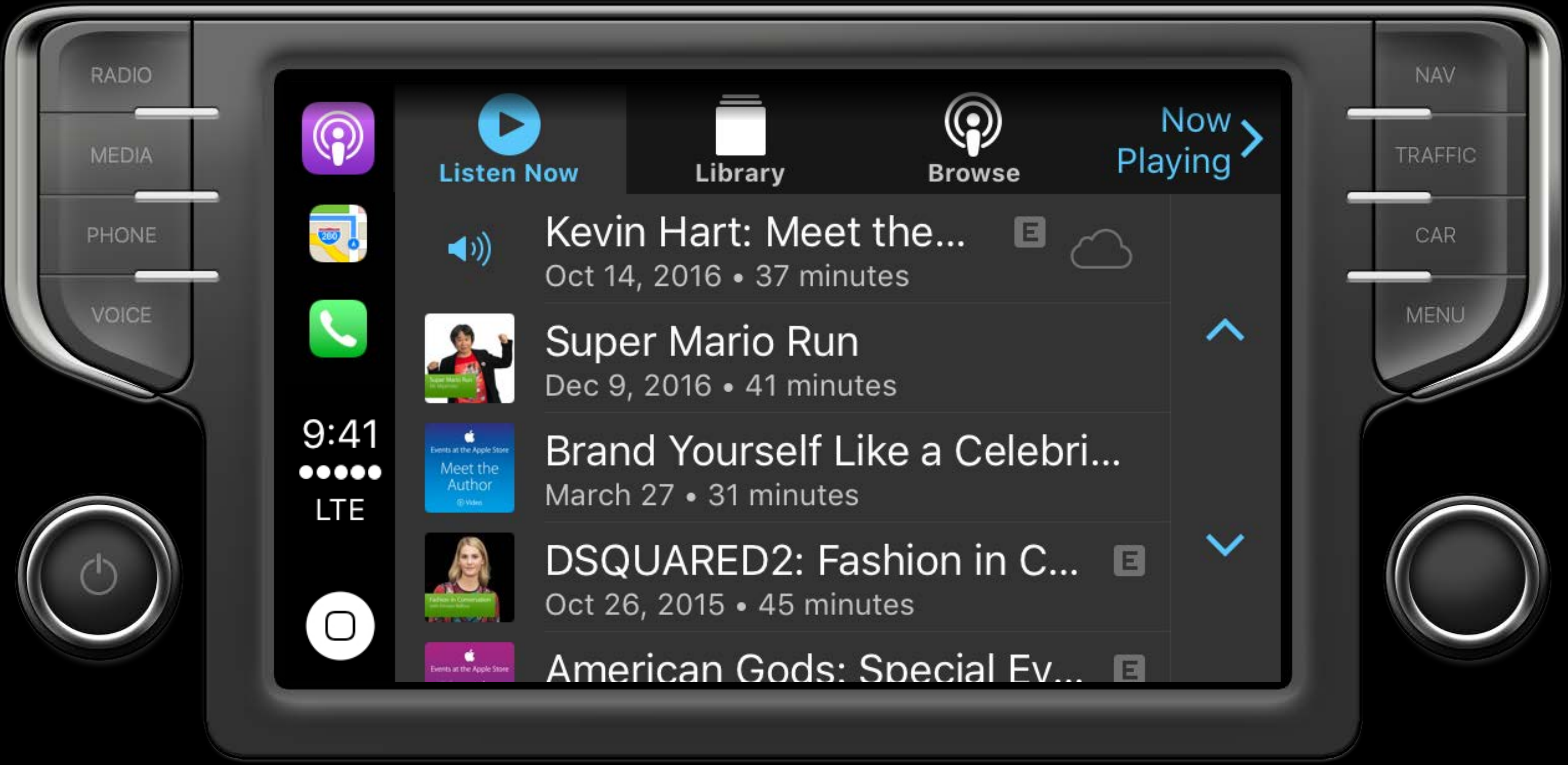
Audio Apps

Messaging and VoIP Calling Apps

Automaker Apps

## Audio Apps

# Audio Apps



Kevin Hart: Meet the... E
Oct 14, 2016 • 37 minutes

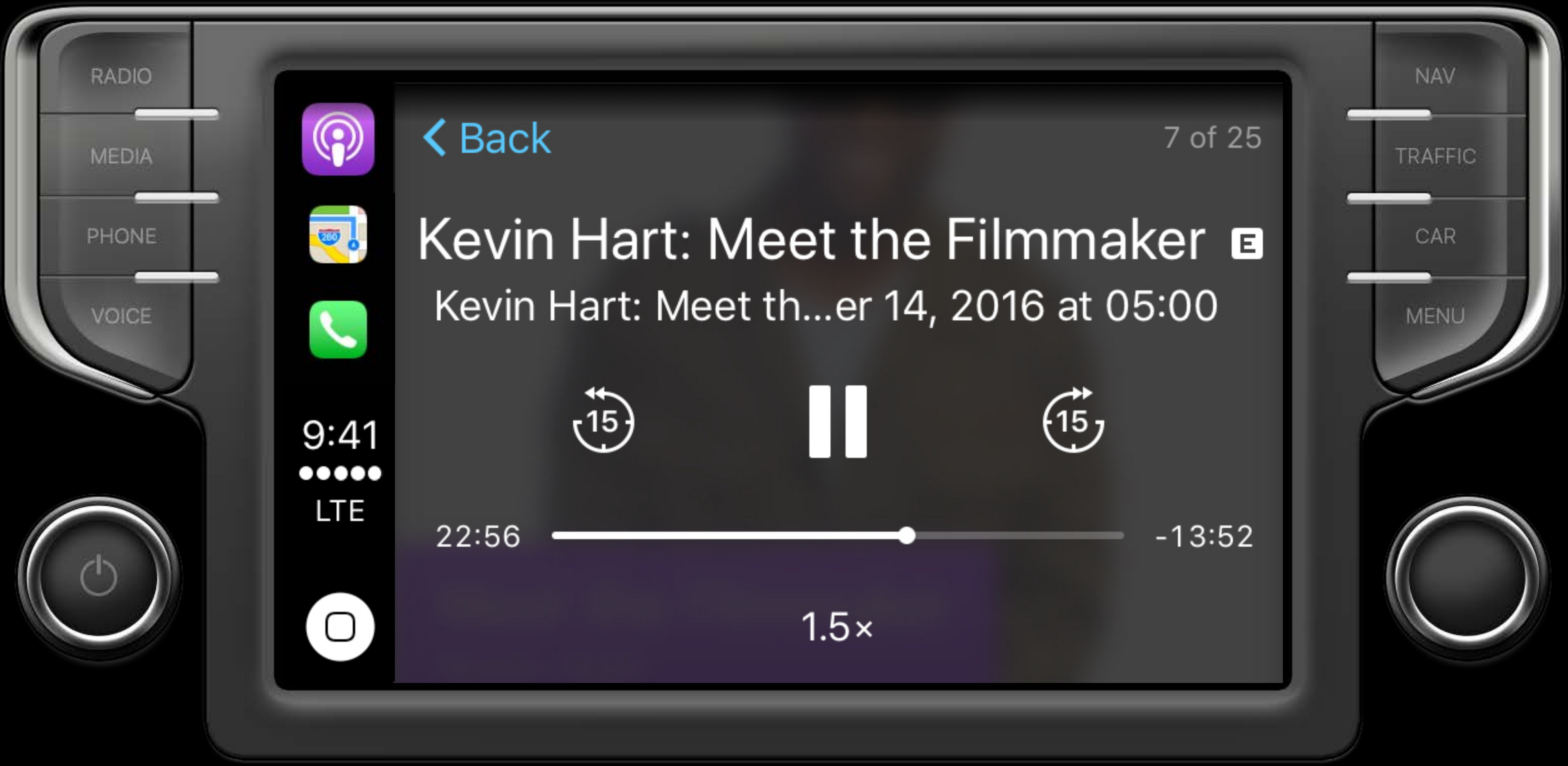
Super Mario Run
Dec 9, 2016 • 41 minutes

Brand Yourself Like a Celebri...
March 27 • 31 minutes

DSQUARED2: Fashion in C... E
Oct 26, 2015 • 45 minutes

American Gods: Special Ev... E

# Audio Apps



Kevin Hart: Meet the Filmmaker E

Kevin Hart: Meet th...er 14, 2016 at 05:00

## Requirements for Audio Apps

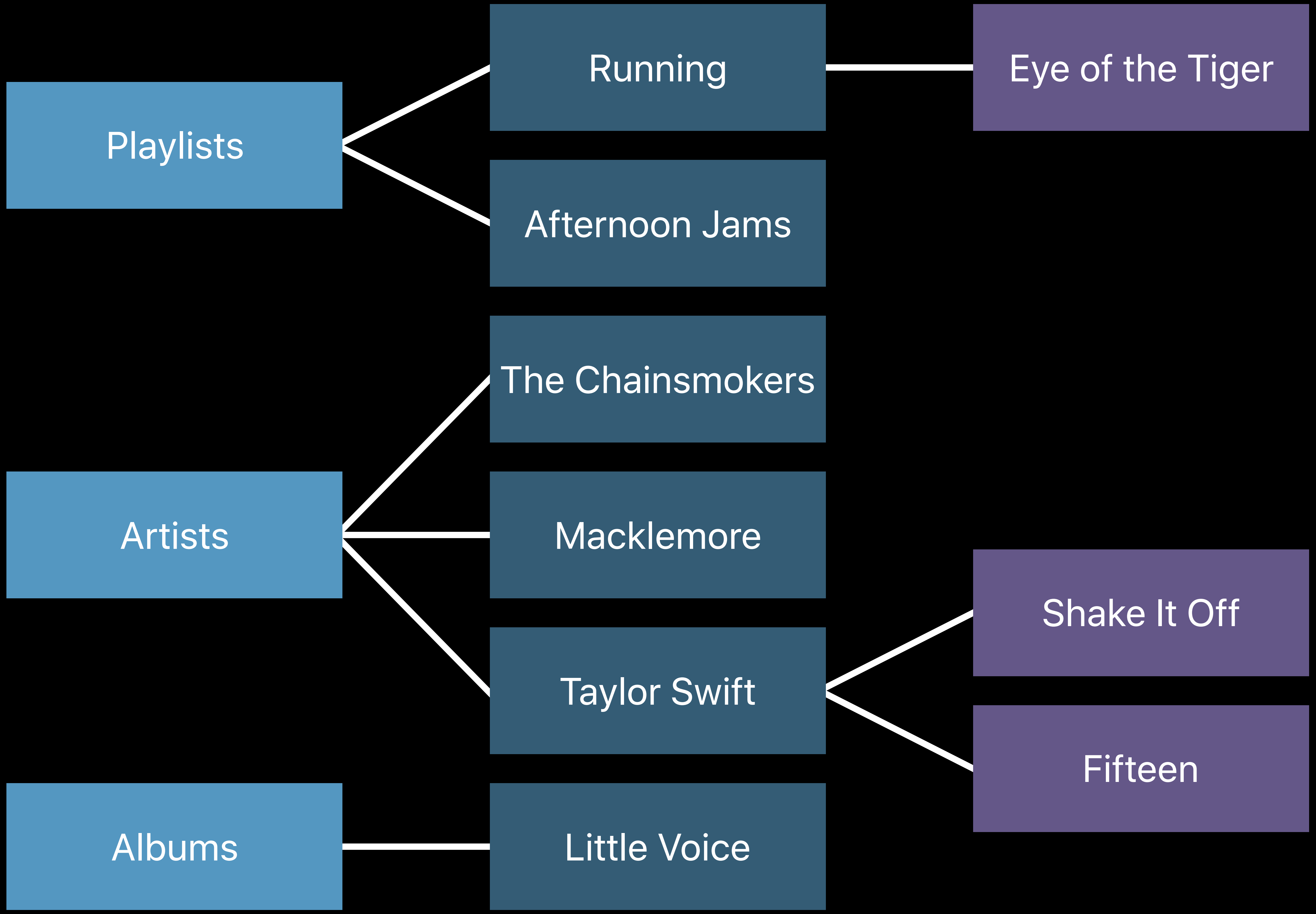
Implement the `MPPlayableContent` API

- `MPPlayableContentDataSource, MPPlayableContentDelegate`

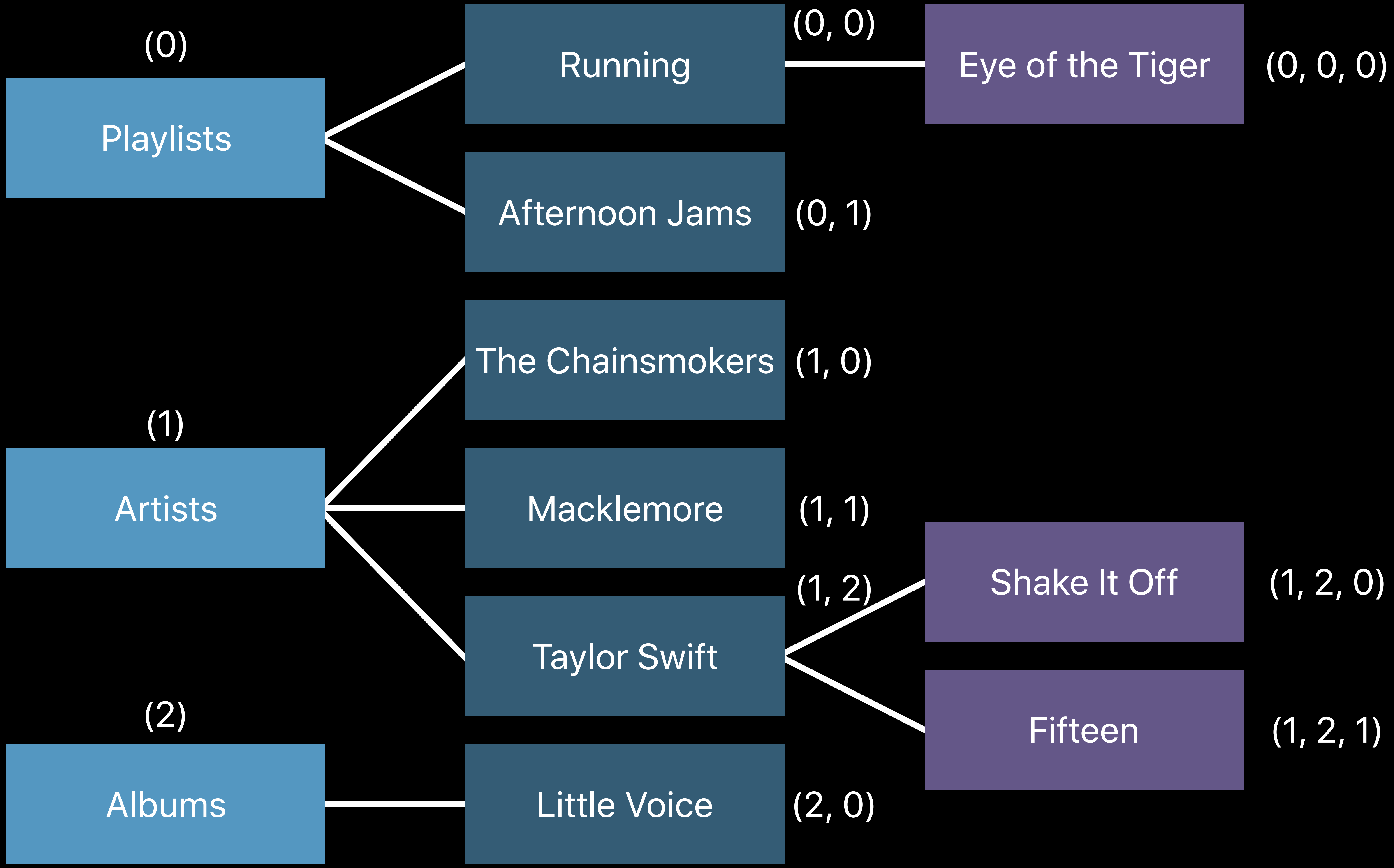
Respond to `MPRemoteCommandCenter` events

Update `MPNowPlayingInfoCenter`

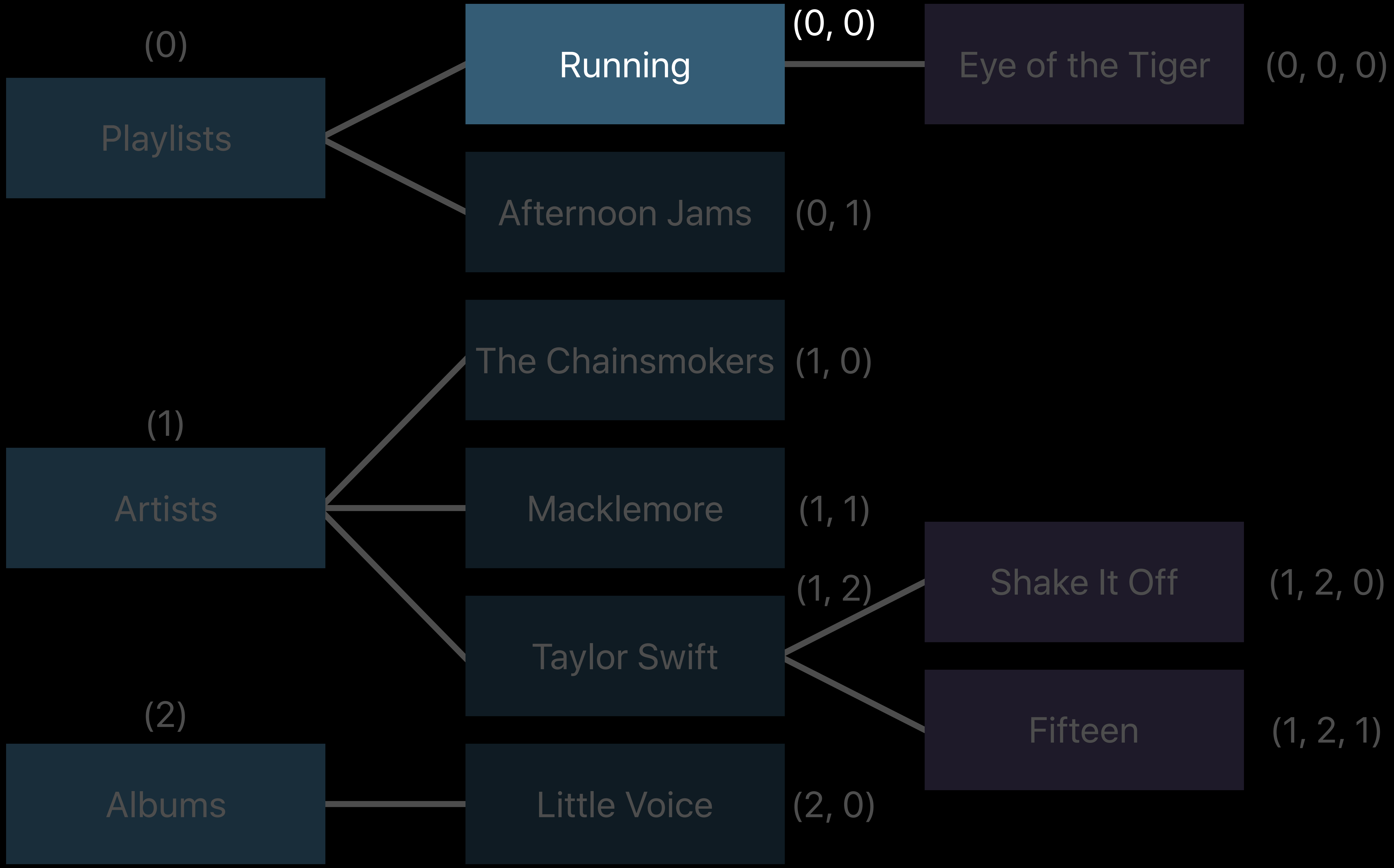
# Building the Data Source



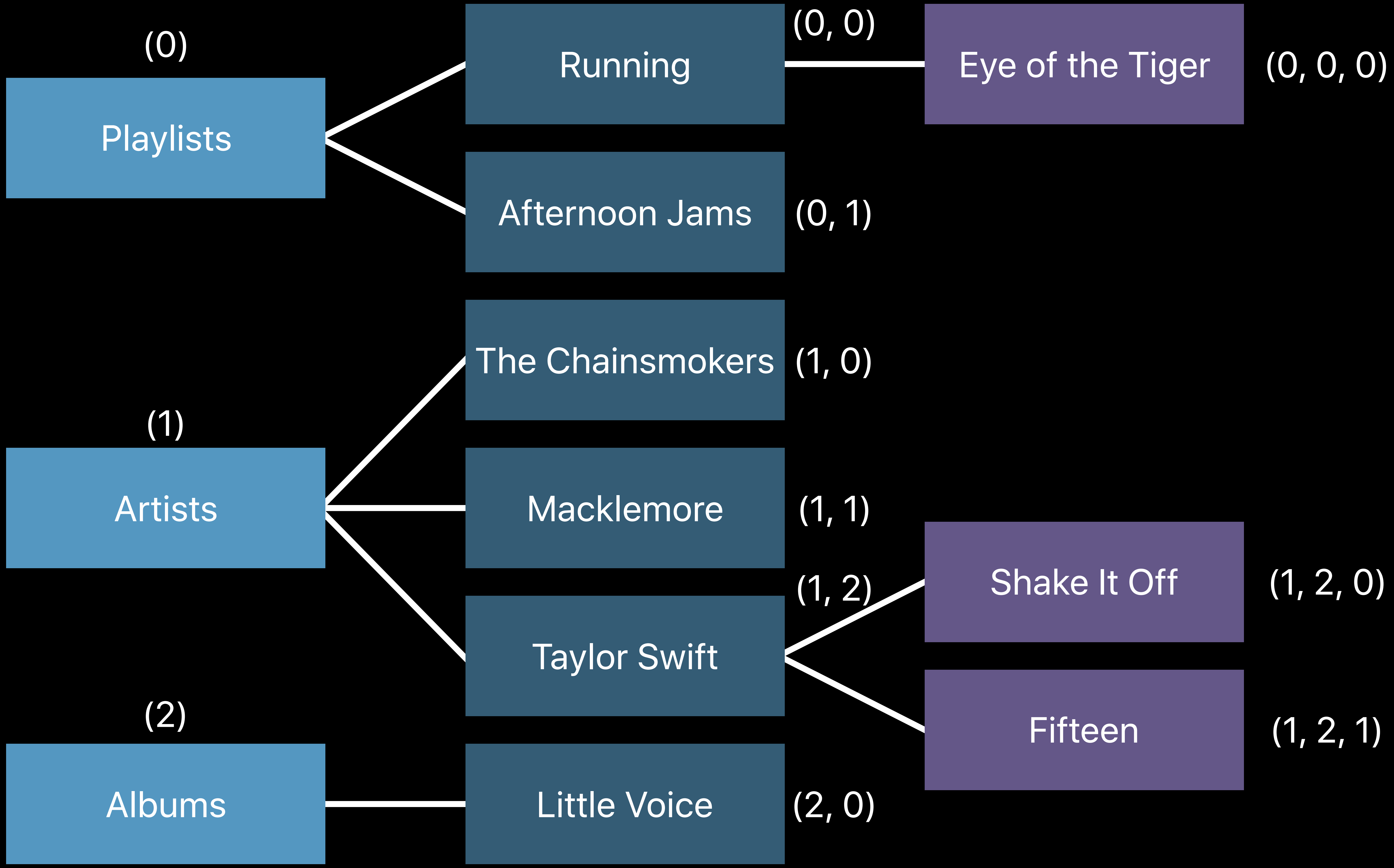
# `contentItem` at (0, 0)



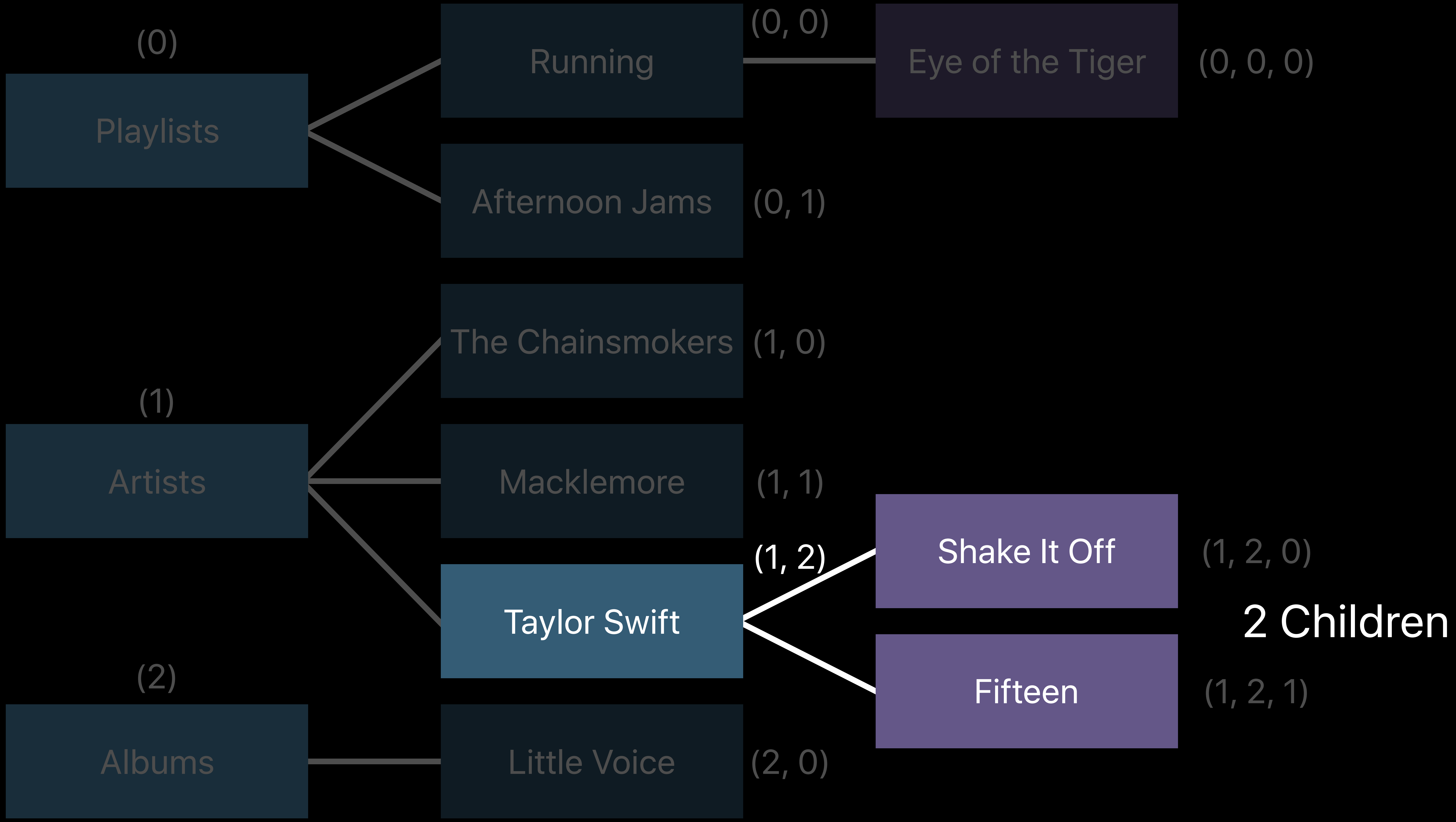
# `contentItem` at (0, 0)



# `numberOfChildItems` at (1,2)



# `numberOfChildItems` at (1,2)



## Content Limits

Content limits may be enforced that can truncate the number of rows and container depth

Take changes into account with `MPPlayableContentManager`

```swift
extension YourAppContentManager : MPPlayableContentDelegate {
    // called whenever CarPlay state changes
    func playableContentManager(
                _ contentManager: MPPlayableContentManager,
               didUpdate context: MPPlayableManagerContext) {
        // check to see if content limits are enforced
        let contentLimitsEnforced = context.contentLimitsEnforced
        if contentLimitsEnforced {
            // the maximum number of items shown in a list when content limits are enforced
            let contentLimitItemCount = context.enforcedContentItemsCount
            // the maximum depth in the hierarchy when content limits are enforced
            let contentLimitTreeDepth = context.enforcedContentTreeDepth
        } else {
            …
        }
    }
}
```

## Tabs

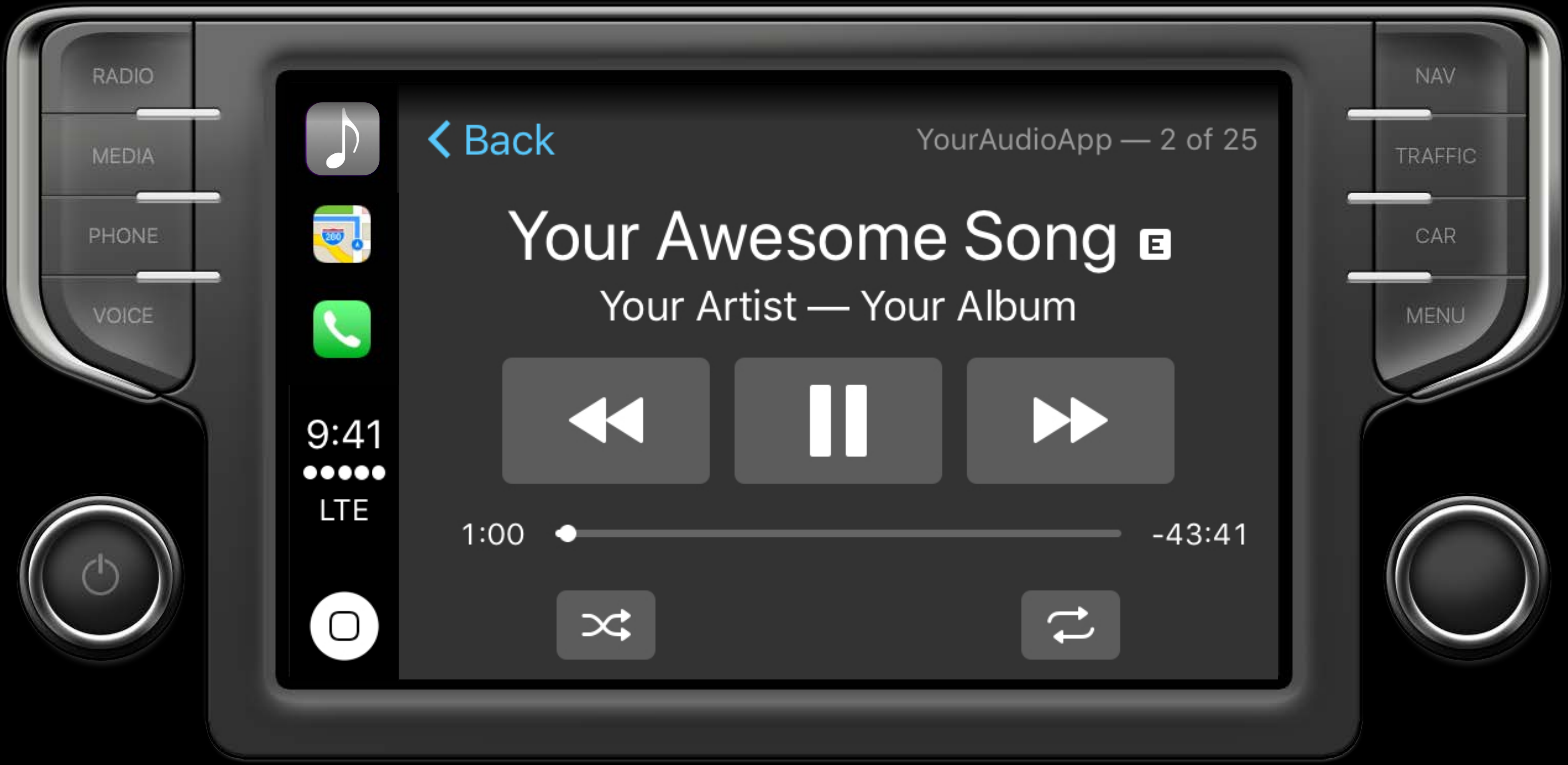
Tabs—Add `UIBrowsableContentSupportsSectionedBrowsing` to Info.plist

Recommended to have at most 4 tabs for space constraints

Keep tab titles as short as possible for narrow screens and Now Playing button

Tab image assets will be rendered as template images

# Now Playing Screen



```
// Set Metadata to be Displayed in Now Playing Info Center

let infoCenter = MPNowPlayingInfoCenter.default()
infoCenter.nowPlayingInfo = [MPMediaItemPropertyTitle: "Style",
                             MPMediaItemPropertyArtist: "Taylor Swift",
                             MPMediaItemPropertyAlbumTitle: "1989",
                             MPMediaItemPropertyGenre: "Pop",
                             MPMediaItemPropertyReleaseDate: "2014",
                             MPMediaItemPropertyPlaybackDuration: 231,
                             MPMediaItemPropertyArtwork: mediaItemArtwork,
                             MPNowPlayingInfoPropertyElapsedPlayback: 53,
                             MPNowPlayingInfoPropertyDefaultPlaybackRate: 1,
                             MPNowPlayingInfoPropertyPlaybackRate: 1,
                             MPNowPlayingInfoPropertyPlaybackQueueCount: 13,
                             MPNowPlayingInfoPropertyPlaybackQueueIndex: 3,
                            … ]
```

## Playback Controls

`MPRemoteCommandCenter`

- Play, pause, stop
- Previous track, next track
- Seek backward, seek forward
- Skip backward, skip forward
- Shuffle, repeat
- Like, dislike, bookmark
- Change playback rate

## Changing Playback Rate

NEW

Add default playback rate to `MPNowPlayingInfoCenter`:

`MPNowPlayingInfoPropertyDefaultPlaybackRate`

Implement `changePlaybackRateCommand` with `supportedPlaybackRates` in `MPRemoteCommandCenter`

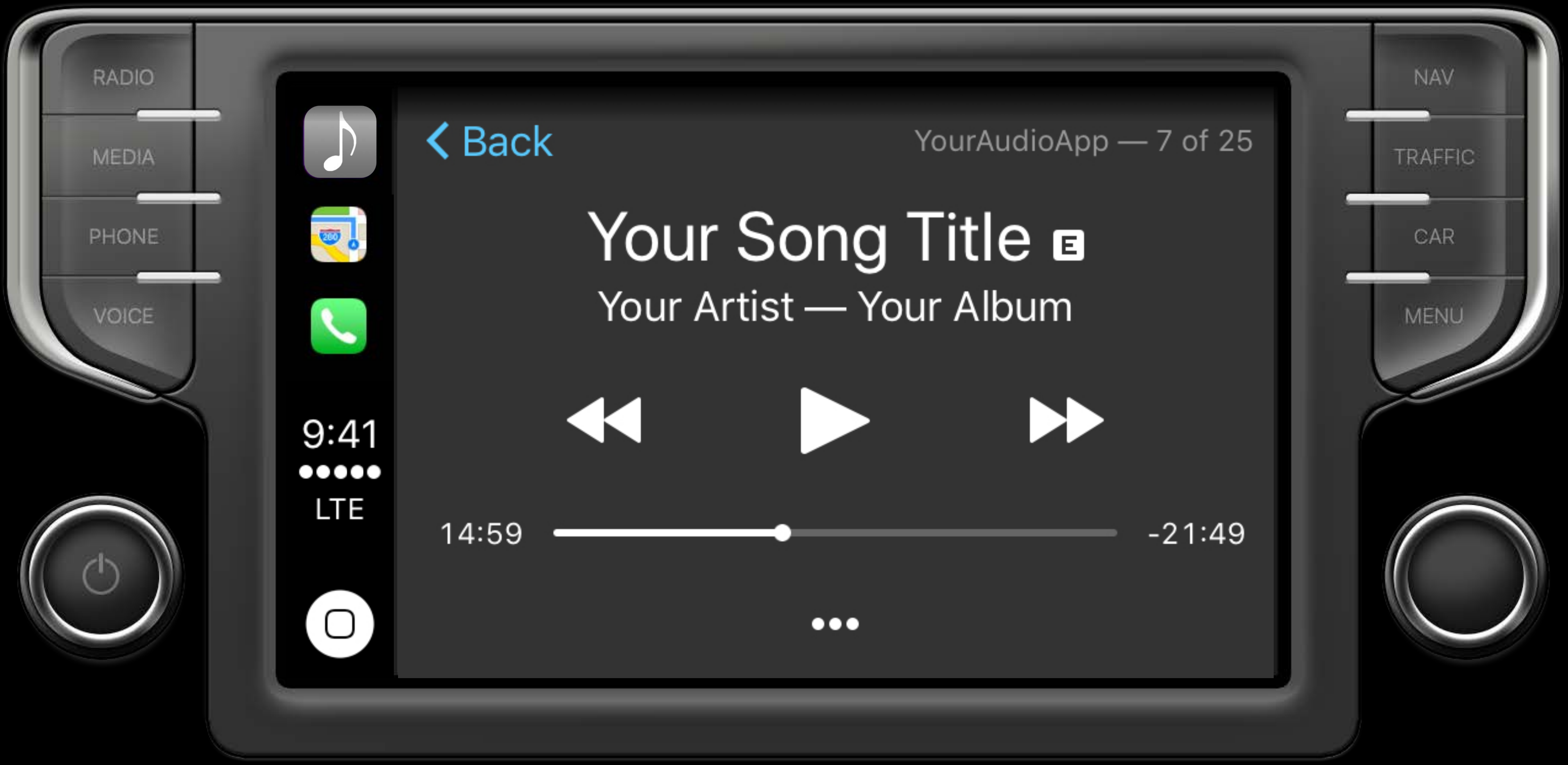
```
// Change Playback Rate

let infoCenter = MPNowPlayingInfoCenter.default()
infoCenter.nowPlayingInfo = [MPNowPlayingInfoPropertyDefaultPlaybackRate: 1.0,
                             … ]

let changePlaybackRateCommand = MPRemoteCommandCenter.shared().changePlaybackRateCommand
changePlaybackRateCommand.supportedPlaybackRates = [0.5, 1.0, 1.5, 2.0]
```

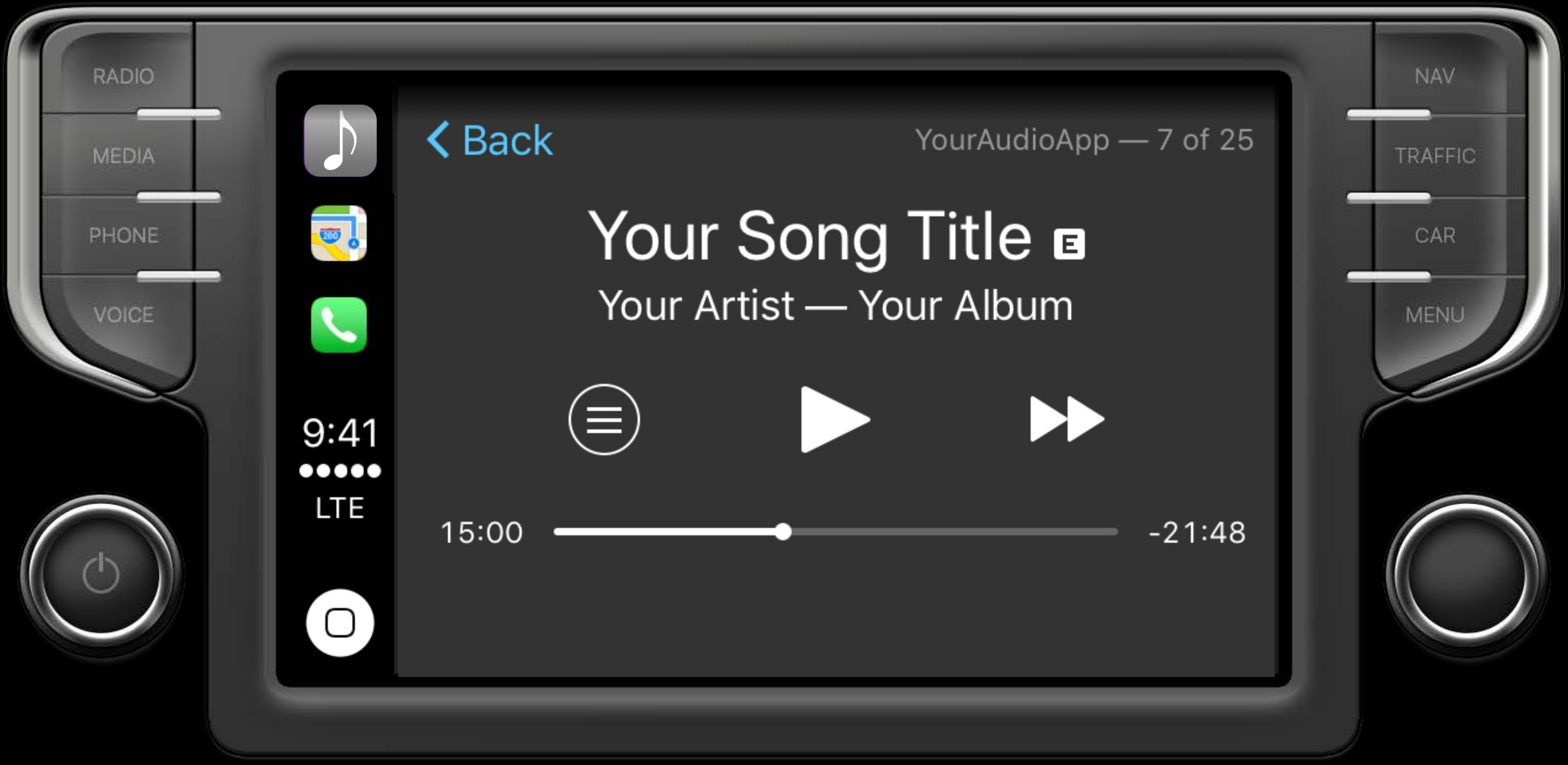
# Playback Controls

If multiple commands are specified, CarPlay may combine them into a single button



## Playback Controls

If multiple commands are specified, CarPlay may combine them into a single button



## Best Practices

Call the completion handler when content is ready to play or be displayed

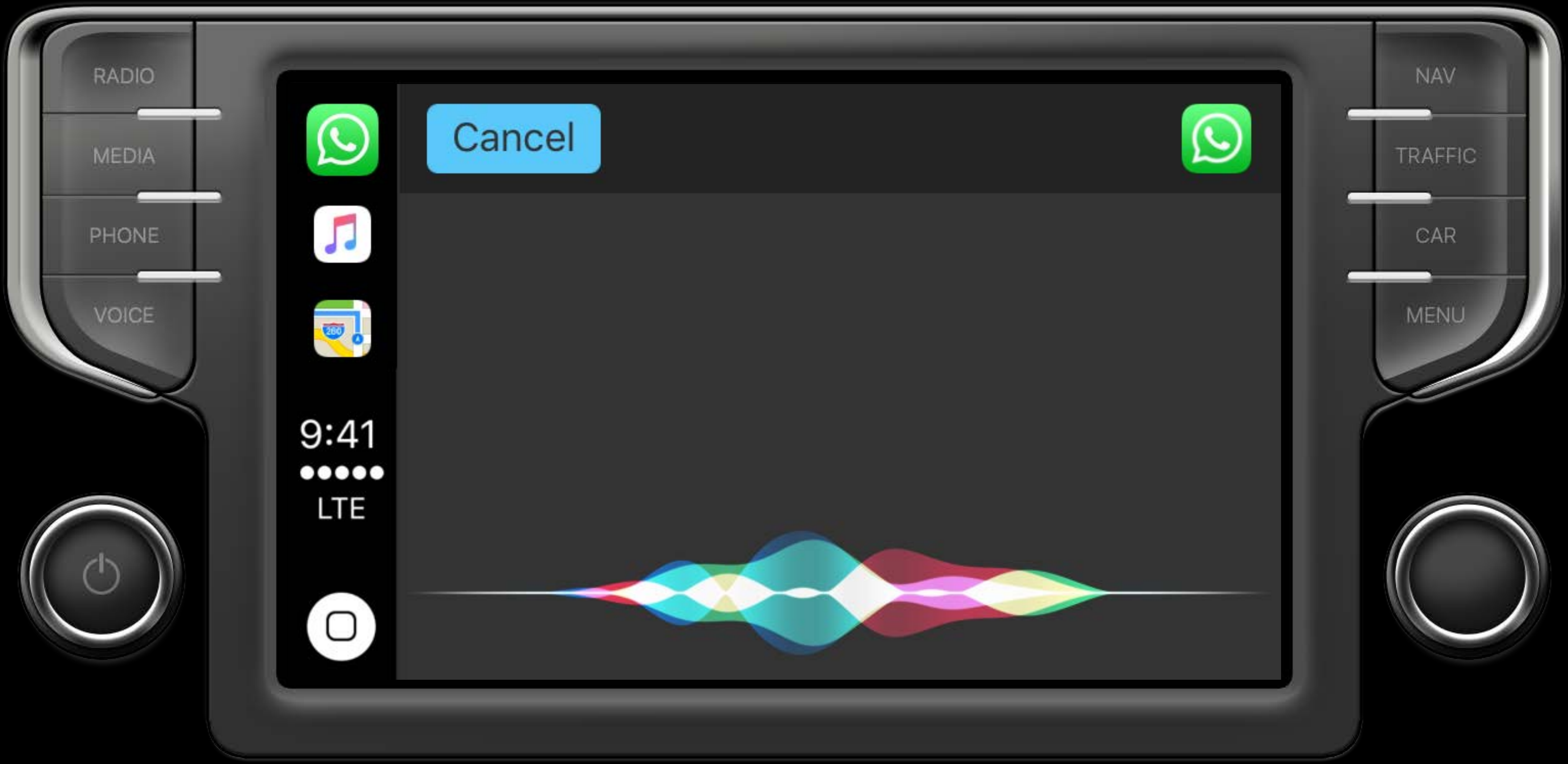
Return at least one row in the root table view for non-tabbed apps

If your app has initial setup, populate the first row with an `MPContentItem` that is neither playable nor a container indicating the state of the app to the user

## Messaging and VoIP Calling Apps

Chris Whitney, CarPlay Engineering

# Messaging and VoIP Calling Apps



## Requirements for Messaging and VoIP Calling Apps

## Requirements for Messaging and VoIP Calling Apps

Messaging

- SiriKit Messaging Intents
- CarPlay Messaging Entitlement

## Requirements for Messaging and VoIP Calling Apps

Messaging

- SiriKit Messaging Intents
- CarPlay Messaging Entitlement

VoIP Calling

- SiriKit Calling Intents
- CarPlay VoIP Calling Entitlement
- CallKit

## Required SiriKit Intents for Messaging Apps

## Required SiriKit Intents for Messaging Apps

Send a message

- `INSendMessageIntent`

## Required SiriKit Intents for Messaging Apps

Send a message

- `INSendMessageIntent`

Search for messages

- `INSearchForMessagesIntent`

## Required SiriKit Intents for Messaging Apps

Send a message

- `INSendMessageIntent`

Search for messages

- `INSearchForMessagesIntent`

Set attributes on a message

- `INSetMessageAttributeIntent`

## Required SiriKit Intents for VoIP Calling Apps

## Required SiriKit Intents for VoIP Calling Apps

Start an audio call

- `INStartAudioCallIntent`

# Required SiriKit Intents for VoIP Calling Apps

Start an audio call

- `INStartAudioCallIntent`

Search call history

- `INSearchCallHistoryIntent`

## Required SiriKit Intents for VoIP Calling Apps

Start an audio call

- `INStartAudioCallIntent`

Search call history

- `INSearchCallHistoryIntent`

| | | |
|---|---|---|
| What's New in SiriKit | Grand Ballroom B | Wednesday 1:50PM |
| Making Great SiriKit Experiences | Grand Ballroom A | Thursday 11:00AM |

# CallKit

## CallKit

Report incoming calls

## CallKit

Report incoming calls

Handle call actions

- Start, answer, end
- Mute, grouping, holding, keypad tones

## CallKit

Report incoming calls

Handle call actions

- Start, answer, end
- Mute, grouping, holding, keypad tones

## Notifications

## Notifications

Request authorization for CarPlay

## Notifications

Request authorization for CarPlay

Separate message notifications into an exclusive notification category

## Notifications

Request authorization for CarPlay

Separate message notifications into an exclusive notification category

Add the CarPlay category option

## Notifications

Request authorization for CarPlay

Separate message notifications into an exclusive notification category

Add the CarPlay category option

Set a SiriKit intent to handle notification selection

```swift
let authorizationOptions : UNAuthorizationOptions = [.badge, .sound, .alert, .carPlay]

let notificationCenter = UNUserNotificationCenter.current()
notificationCenter.requestAuthorization(options: authorizationOptions) { (granted, error) in
    // Enable or disable app features based on authorization
}

let messageCategory = UNNotificationCategory(
                                  identifier: "messages",
                                     actions: […],
                           intentIdentifiers: [INSearchForMessagesIntentIdentifier],
                                     options: .allowInCarPlay)

notificationCenter.setNotificationCategories([messageCategory])
```

## Best Practices

## Best Practices

Don’t show message contents in a notification’s title or subtitle

## Best Practices

Don’t show message contents in a notification’s title or subtitle

Mark content as read in SiriKit

## Best Practices

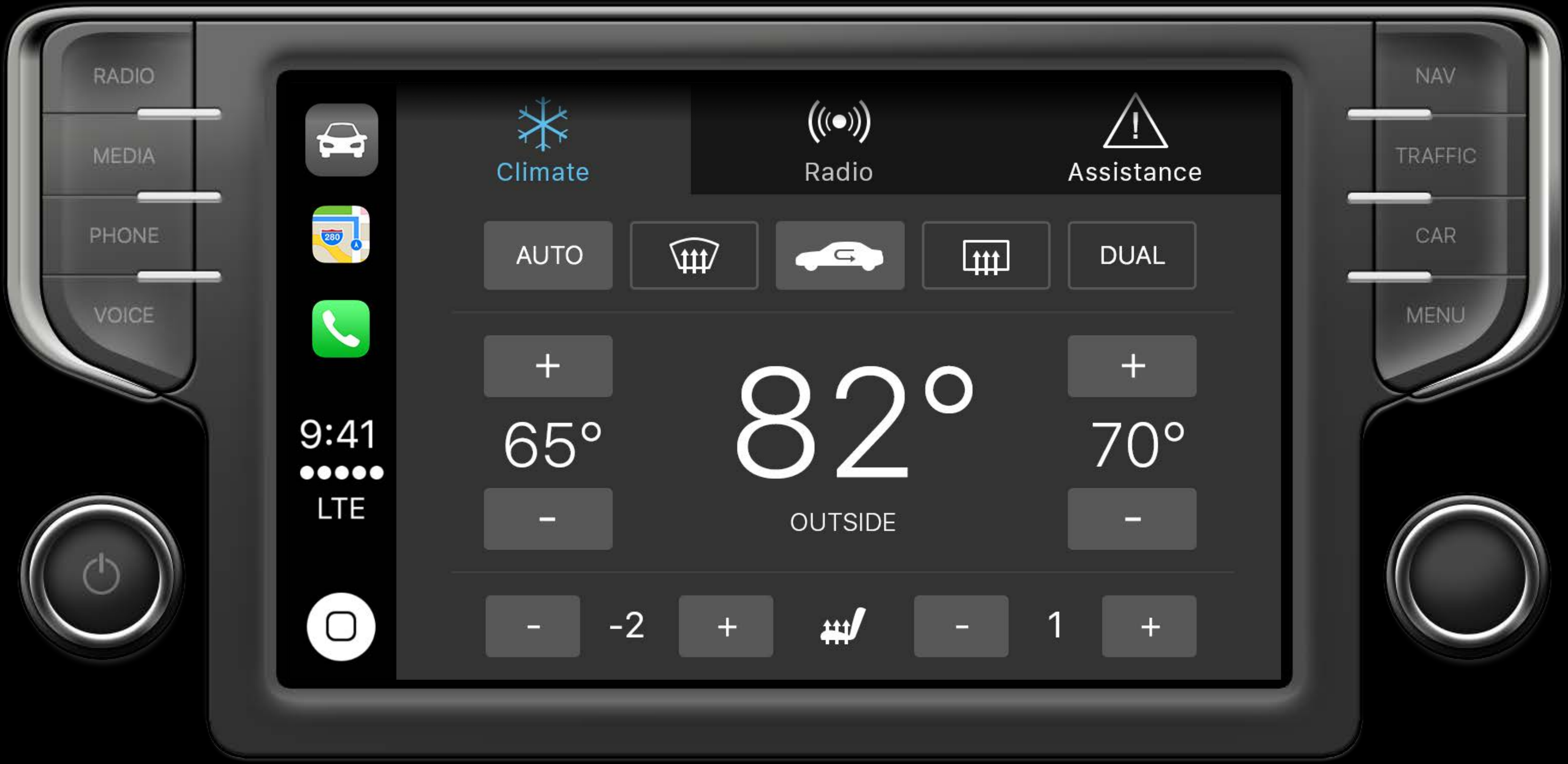
Don’t show message contents in a notification’s title or subtitle

Mark content as read in SiriKit

Show notifications for missed calls and message delivery failures

## Automaker Apps

# Automaker Apps



## Requirements for Automaker Apps

## Requirements for Automaker Apps

Created by the car’s manufacturer to provide information and control features

## Requirements for Automaker Apps

Created by the car’s manufacturer to provide information and control features

Automaker-specific entitlement

## Requirements for Automaker Apps

Created by the car’s manufacturer to provide information and control features

Automaker-specific entitlement

Display a user interface in CarPlay

## Requirements for Automaker Apps

Created by the car’s manufacturer to provide information and control features

Automaker-specific entitlement

Display a user interface in CarPlay

Only appear on supported vehicles from the automaker

## Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

`com.sampleautos.performance`

```
com.sampleautos.performance
com.sampleautos.climate
```

## Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

`com.sampleautos.performance`

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

RADIO
MEDIA
PHONE
VOICE
NAV
TRAFFIC
CAR
MENU
9:41
LTE
Phone
Music
Maps
Messages
Now Playing
Podcasts
Audiobooks
Track Laps

```
com.sampleautos.performance
com.sampleautos.climate
```

## Matching Automaker Apps to Vehicles

### CarPlay Protocols Entitlement

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

```
com.sampleautos.radio
com.sampleautos.climate
```

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

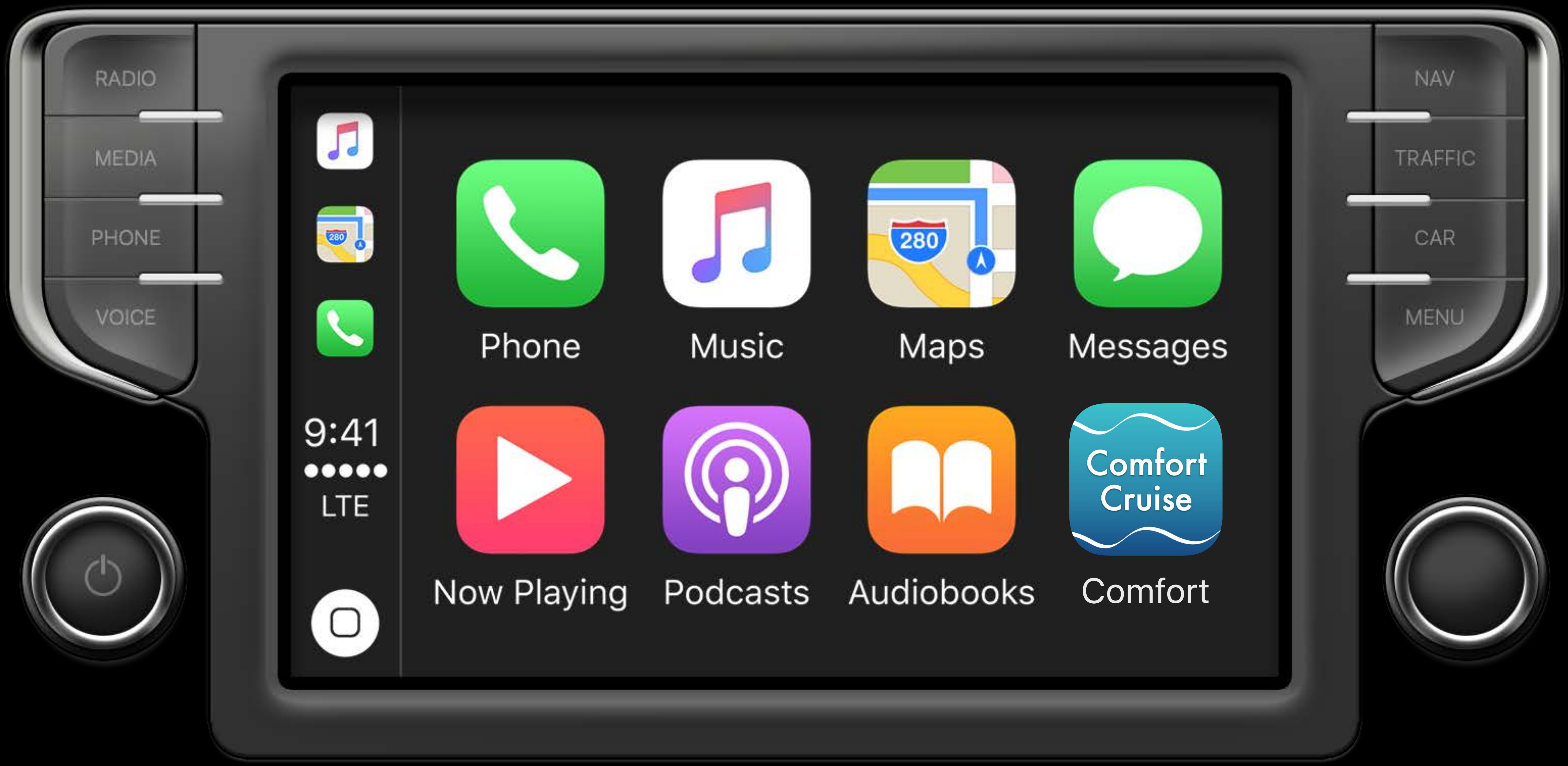
CarPlay Protocols Entitlement

```
com.sampleautos.radio
com.sampleautos.climate
```

```
com.sampleautos.performance
com.sampleautos.climate
```

## Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement



```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

```
com.sampleautos.performance
com.sampleautos.climate
```

# Matching Automaker Apps to Vehicles

## CarPlay Protocols Entitlement

`com.sampleautos.electric`

```
com.sampleautos.performance
com.sampleautos.climate
```

## Matching Automaker Apps to Vehicles

CarPlay Protocols Entitlement

`com.sampleautos.electric`

```
com.sampleautos.performance
com.sampleautos.climate
```

## Communicating with the Vehicle

## Communicating with the Vehicle

Use External Accessory Framework

## Communicating with the Vehicle

Use External Accessory Framework

- Define a custom protocol

## Communicating with the Vehicle

Use External Accessory Framework

- Define a custom protocol
- Implement the protocol in the vehicle’s software

## Communicating with the Vehicle

Use External Accessory Framework

- Define a custom protocol
- Implement the protocol in the vehicle’s software
- Observe accessory connection events

## Communicating with the Vehicle

Use External Accessory Framework

- Define a custom protocol
- Implement the protocol in the vehicle’s software
- Observe accessory connection events
- Initialize an EASession for the protocol

## Adding a CarPlay User Interface

## Adding a CarPlay User Interface

Observe `UIScreen` connection events

## Adding a CarPlay User Interface

Observe `UIScreen` connection events

Verify the screen’s traits contain `UIUserInterfaceIdiomCarPlay`

## Adding a CarPlay User Interface

Observe `UIScreen` connection events

Verify the screen’s traits contain `UIUserInterfaceIdiomCarPlay`

Create a `UIWindow` with a root view controller

```swift
var carWindow: UIWindow?

func updateCarWindow() {
    guard let screen = UIScreen.screens.first(where:
        { $0.traitCollection.userInterfaceIdiom == .carPlay })
        else {
            // CarPlay is not connected
            self.carWindow = nil;
            return
    }

    // CarPlay is connected
    let carWindow = UIWindow(frame: screen.bounds)
    carWindow.screen = screen
    carWindow.makeKeyAndVisible()
    carWindow.rootViewController = CarViewController(nibName: nil, bundle: nil)
    self.carWindow = carWindow
}
```

```swift
updateCarWindow()

let notificationCenter = NotificationCenter.default
notificationCenter.addObserver(self,
                               selector: #selector(screenDidUpdate(notification:)),
                                   name: .UIScreenDidConnect,
                                 object: nil)
notificationCenter.addObserver(self,
                               selector: #selector(screenDidUpdate(notification:)),
                                   name: .UIScreenDidDisconnect,
                                 object: nil)

@objc func screenDidUpdate(notification: Notification) {
    updateCarWindow()
}
```

## UIKit in CarPlay

## UIKit in CarPlay

`UIButtonTypeSystem` displays with a CarPlay style

## UIKit in CarPlay

`UIButtonTypeSystem` displays with a CarPlay style

`UITableViewController` may limit table length

## UIKit in CarPlay

`UIButtonTypeSystem` displays with a CarPlay style

`UITableViewController` may limit table length

`UIFocusEnvironment` responds to input device events

## UIKit in CarPlay

`UIButtonTypeSystem` displays with a CarPlay style

`UITableViewController` may limit table length

`UIFocusEnvironment` responds to input device events

Limited availability of system user interface elements

## SiriKit for Automaker Apps

## SiriKit for Automaker Apps

Car Commands Intents

- Lock and unlock
- Fuel or charge level
- Signal tone and lights

## SiriKit for Automaker Apps

Car Commands Intents

- Lock and unlock
- Fuel or charge level
- Signal tone and lights

CarPlay Intents

- Climate, defroster, and seat heater settings
- Radio and audio source selection

## Best Practices

## Best Practices

Make your app useful even when outside of the car

## Best Practices

Make your app useful even when outside of the car

Consider forwards and backwards compatibility

## Best Practices

Make your app useful even when outside of the car

Consider forwards and backwards compatibility

Simplify the user interface – buttons, labels, tables, navigation, tabs

## Best Practices

Make your app useful even when outside of the car

Consider forwards and backwards compatibility

Simplify the user interface – buttons, labels, tables, navigation, tabs

Determine how focus should move between UI elements

## More Information

https://developer.apple.com/wwdc17/719

# Related Sessions

| | | |
|---|---|---|
| Debugging with Xcode 9 | Hall 2 | Wednesday 10:00AM |
| What's New in SiriKit | Grand Ballroom B | Wednesday 1:50PM |
| Making Great SiriKit Experiences | Grand Ballroom A | Thursday 11:00AM |
| Developing Wireless CarPlay Systems | | WWDC 2017 Video |
| Enhancing VoIP Apps with CallKit | | WWDC 2016 |
| Developing CarPlay Systems, Part 1 | | WWDC 2016 |
| Developing CarPlay Systems, Part 2 | | WWDC 2016 |

# Labs

| | | |
|---|---|---|
| CarPlay Lab | Technology Lab D | Wednesday 4:10PM–6:00PM |
| SiriKit Lab | Technology Lab B | Wednesday 3:10PM–5:00PM |
| SiriKit Lab | Technology Lab C | Friday 9:00AM–12:00PM |

WWDC17