#WWDC18

CarPlay Audio and Navigation Apps

Tunes and turns

Jonathan Hersh, iOS Car Experience
Albert Wan, iOS Car Experience
Mike Knippers, iOS Car Experience

Agenda

CarPlay audio app improvements

Navigating with the new CarPlay framework

CarPlay navigation demo


9:41
LTE
Phone
Music
Maps
Messages
Now Playing
Podcasts
Audiobooks
WhatsApp
BACK

What's Special About CarPlay?

Touch screens, rotary knob, and touchpad inputs

Left and right hand drive

Night interface style

Screen sizes

Automaker

Messaging

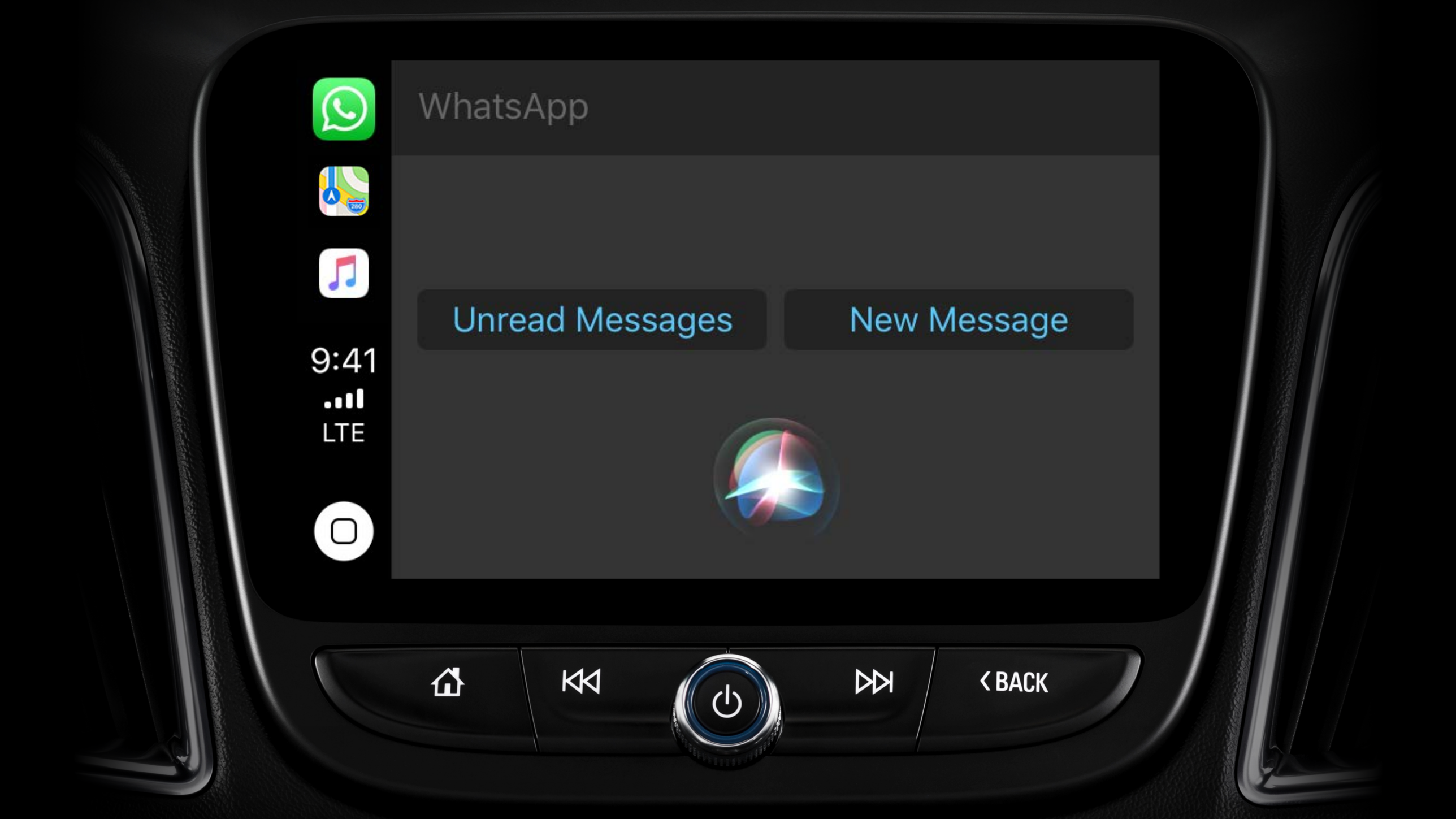
VoIP calling

Audio

Navigation

https://developer.apple.com/carplay


Climate
Radio
Assistance
AUTO
DUAL
9:41
LTE
65°
82°
70°
OUTSIDE
-2
1
BACK


WhatsApp
Unread Messages
New Message
9:41
LTE
BACK


Back
Srirocka — 9 of 24
Dragon's Breath
E
Five Alarm Chilis — 3 Million Scoville
9:41
LTE
1:00
-2:39
BACK


250 ft
S Market St
and then
Balbach St
Discovery Meadow
S Almaden Ave
W William St
S Market St
S 1st St
9:41
LTE
9:45
arrival
4
min
0.7
mi
10
MPH
BACK

Automaker

Messaging

VoIP calling

Audio

Navigation

Automaker

Messaging

VoIP calling

Audio

Navigation

Audio Apps in CarPlay

Albert Wan, iOS Car Experience


Back
Srirocka — 1 of 12
Nacho Spices
E
Five Alarm Chilis — Admiral Pepper
9:41
LTE
1:30
-4:30
BACK

Srirocka

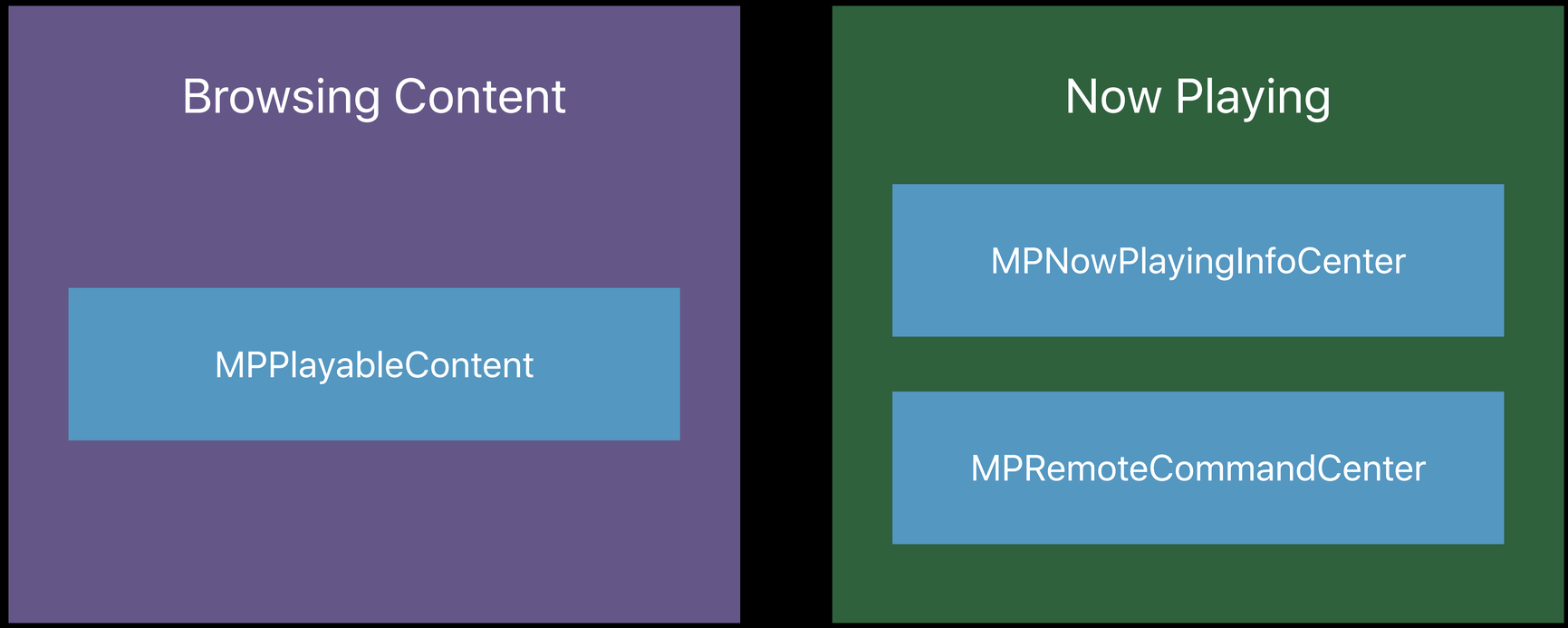
Audio Apps in CarPlay

Template based

Works with all CarPlay systems

Uses existing MediaPlayer APIs

CarPlay Audio App APIs



```swift
func application(
_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {

    // Set up data source and delegate
    MPPlayableContentManager.shared().dataSource = SrirockaContentManager.shared
    MPPlayableContentManager.shared().delegate = SrirockaContentManager.shared

    // Set Now Playing metadata in MPNowPlayingInfoCenter
    let nowPlayingInfo: [String: Any] = [:]
    MPNowPlayingInfoCenter.default().nowPlayingInfo = nowPlayingInfo

    // Respond to remote command events
    let commandCenter = MPRemoteCommandCenter.shared()
    commandCenter.playCommand.isEnabled = true
    commandCenter.playCommand.addTarget { _ in .success }
}
```

```
func application(
_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {

    // Set up data source and delegate
    MPPlayableContentManager.shared().dataSource = SrirockaContentManager.shared
    MPPlayableContentManager.shared().delegate = SrirockaContentManager.shared

    // Set Now Playing metadata in MPNowPlayingInfoCenter
    let nowPlayingInfo: [String: Any] = [:]
    MPNowPlayingInfoCenter.default().nowPlayingInfo = nowPlayingInfo

    // Respond to remote command events
    let commandCenter = MPRemoteCommandCenter.shared()
    commandCenter.playCommand.isEnabled = true
    commandCenter.playCommand.addTarget { _ in .success }
}
```

```swift
func application(
_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {

    // Set up data source and delegate
    MPPlayableContentManager.shared().dataSource = SrirockaContentManager.shared
    MPPlayableContentManager.shared().delegate = SrirockaContentManager.shared

    // Set Now Playing metadata in MPNowPlayingInfoCenter
    let nowPlayingInfo: [String: Any] = [:]
    MPNowPlayingInfoCenter.default().nowPlayingInfo = nowPlayingInfo

    // Respond to remote command events
    let commandCenter = MPRemoteCommandCenter.shared()
    commandCenter.playCommand.isEnabled = true
    commandCenter.playCommand.addTarget { _ in .success }
}
```

```swift
func application(
_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {

    // Set up data source and delegate
    MPPlayableContentManager.shared().dataSource = SrirockaContentManager.shared
    MPPlayableContentManager.shared().delegate = SrirockaContentManager.shared

    // Set Now Playing metadata in MPNowPlayingInfoCenter
    let nowPlayingInfo: [String: Any] = [:]
    MPNowPlayingInfoCenter.default().nowPlayingInfo = nowPlayingInfo

    // Respond to remote command events
    let commandCenter = MPRemoteCommandCenter.shared()
    commandCenter.playCommand.isEnabled = true
    commandCenter.playCommand.addTarget { _ in .success }
}
```

MPPlayableContent

MPPlayableContent

Remastered

Improvements in iOS 12

NEW

Improved performance in `MPPlayableContent`

Faster startup sequence

Smoother animations

Better communication to your app

Best Practices

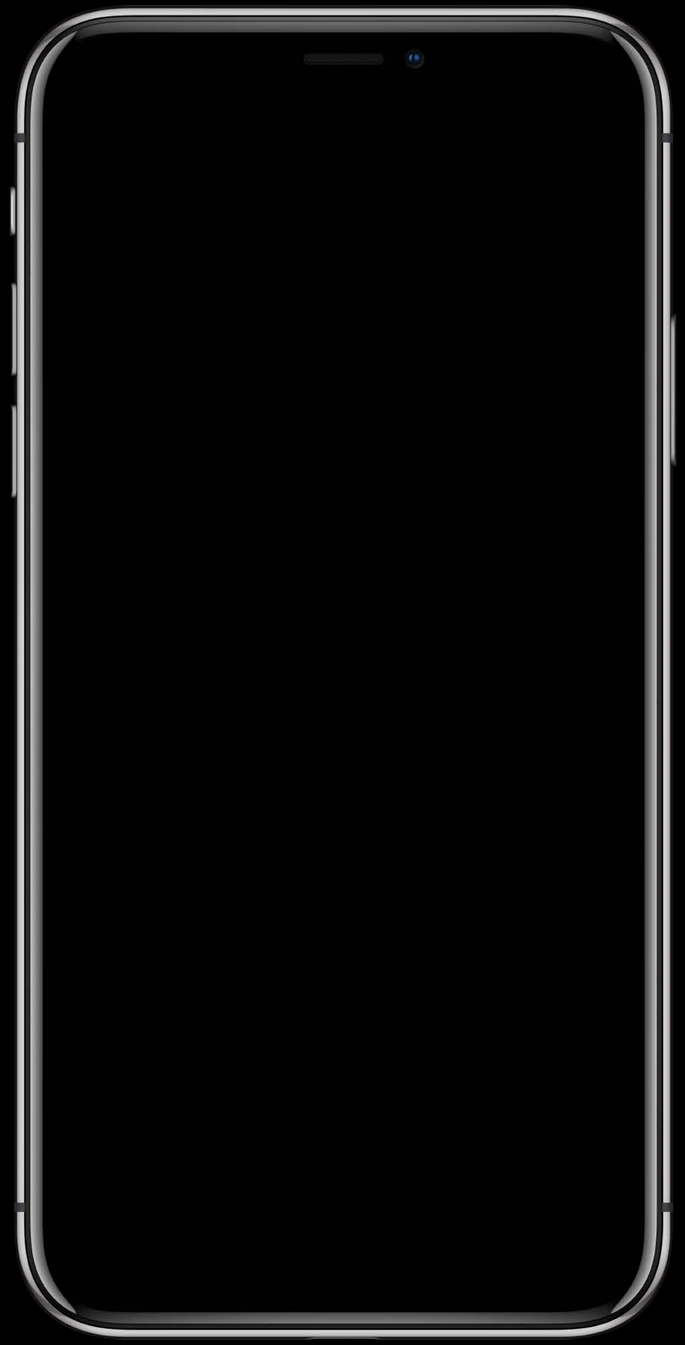
Call `reloadData()` only when needed

Use `beginUpdates()` and `endUpdates()`

Keep an internal representation of the data source to optimize performance


Back
Sizzling Sensations
Heating Habaneros
Jalapeño Hits
Cooling Cayennes
9:41
LTE
BACK


Back
Sizzling Sensations
Heating Habaneros
Jalapeño Hits
Cooling Cayennes
9:41
LTE
BACK



Don’t Miss a Beat!

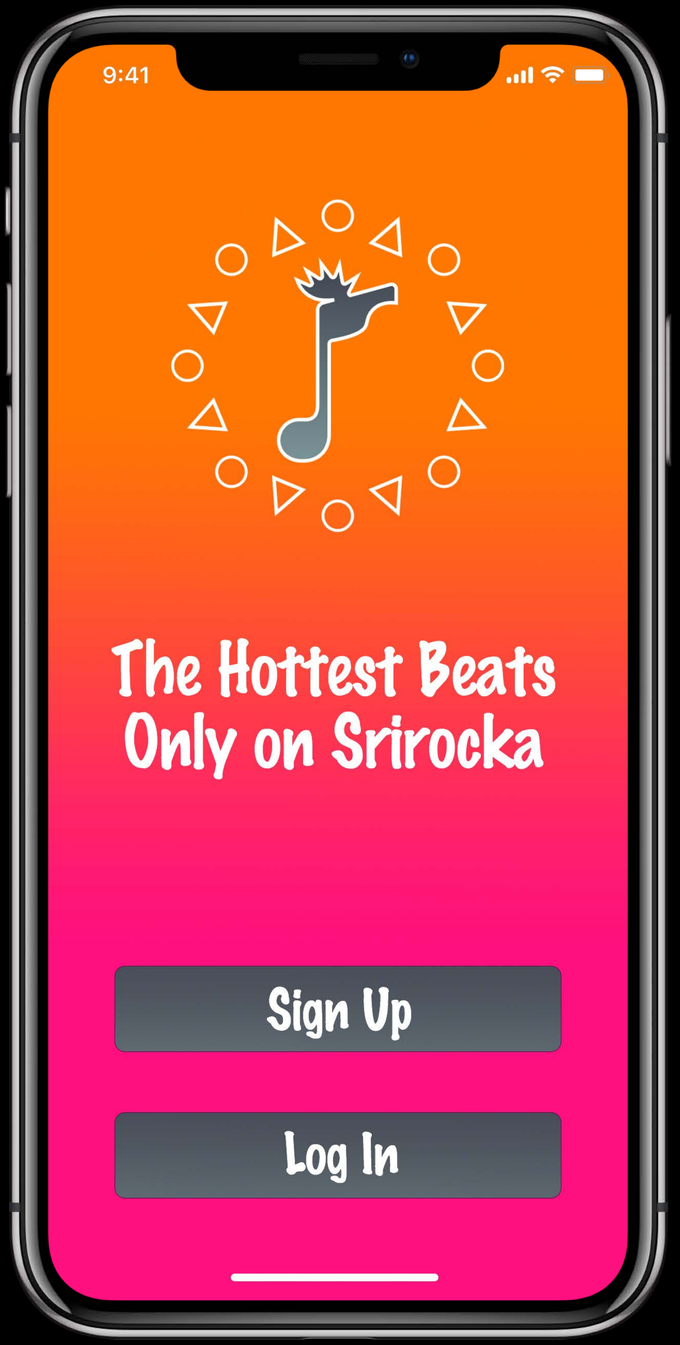
Account for these common scenarios

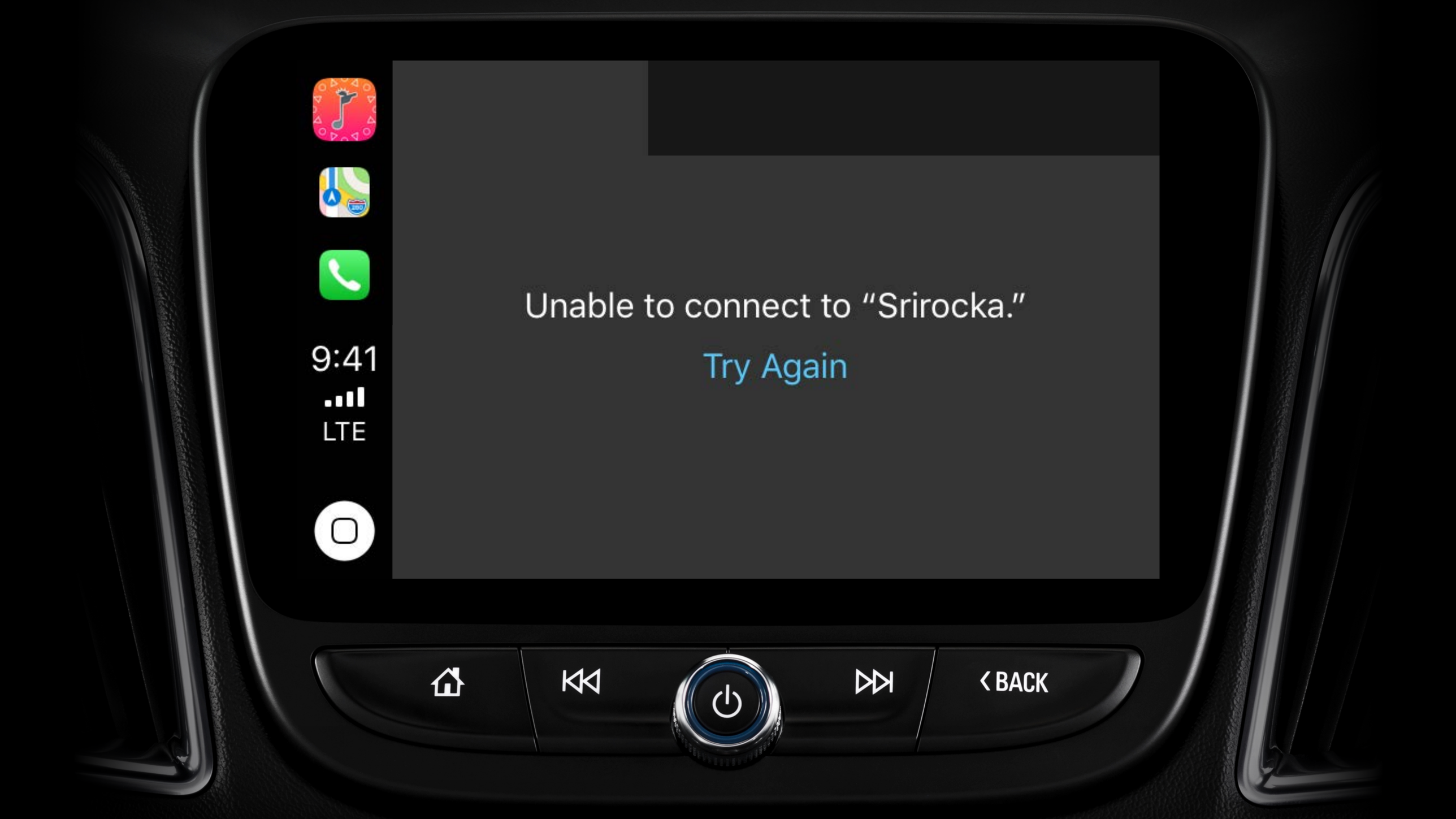
- Screen locked with passcode
- Unreliable network connectivity

Anticipate the Hits

Utilize `beginLoadingChildItems()` to initiate fetching content

```
func beginLoadingChildItems(at indexPath: IndexPath,
                            completionHandler: @escaping (Error?) -> Void) {
    if indexPath.indexAtPosition(2) == 0 {
        // Start fetching content that requires a network connection or needs some setup
        startProcessingHeatingHabaneros()
    }
    ...
    completionHandler(nil)
}
```


9:41
The Hottest Beats
Only on Srirocka
Sign Up
Log In


9:41
LTE
Unable to connect to “Srirocka.”
Try Again
BACK


Recommended
Browse
Atomic Awakenings
Mild Medleys
Sizzling Sensations
Tangy Tunes
9:41
LTE
BACK

Greatest Hits

Use `MPPlayableContent` to populate the CarPlay screen

Anticipate user scenarios while driving

Run your audio apps in CarPlay!

Navigating with CarPlay Framework

Mike Knippers, iOS Car Experience


Back
Srirocka — 15 of 24
Peter Piper's Pickled Peppers
E
Five Alarm Chilis — Pickled Peppers
9:41
LTE
0:34
-3:25
BACK

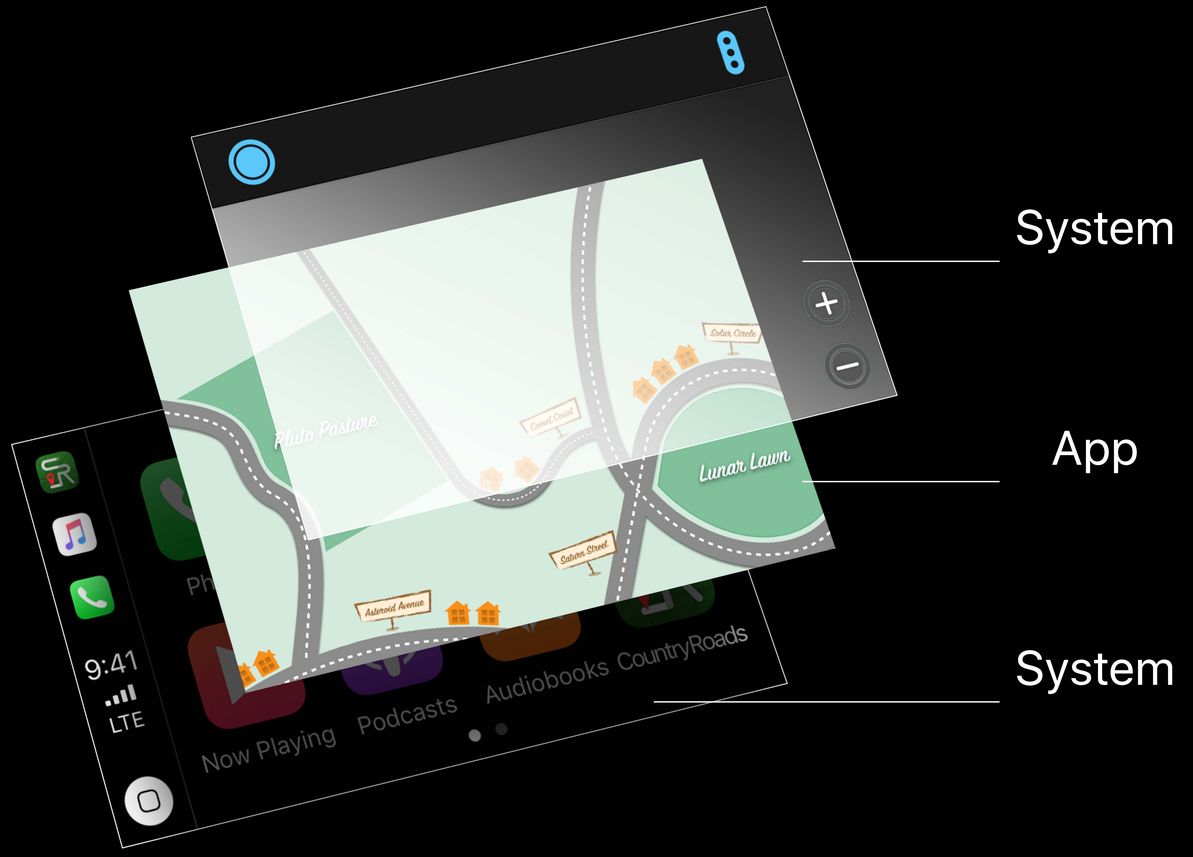
Navigation Apps in CarPlay

NEW

Built with CarPlay framework

Template based

Supports all CarPlay systems




9:41
LTE
Phone
Music
Maps
Messages
Now Playing
Podcasts
Audiobooks
CountryRoads
BACK


9:41
LTE
Phone
Music
Maps
Messages
Now Playing
Podcasts
Audiobooks
CountryRoads
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar Lawn
Asteroid Avenue
Saturn Street
BACK

```swift
func application(_ application: UIApplication,
                 didConnectCarInterfaceController interfaceController: CPInterfaceController,
                 to window: UIWindow) {

    self.interfaceController = interfaceController
    self.carWindow = window

    let rootViewController = MyRootViewController()
    window.rootViewController = rootViewController

    let rootTemplate: CPMapTemplate = createRootTemplate()
    self.interfaceController?.setRootTemplate(rootTemplate, animated: false)
}
```

```
func application(_ application: UIApplication,
                 didConnectCarInterfaceController interfaceController: CPInterfaceController,
                 to window: UIWindow) {

    self.interfaceController = interfaceController
    self.carWindow = window

    let rootViewController = MyRootViewController()
    window.rootViewController = rootViewController

    let rootTemplate: CPMapTemplate = createRootTemplate()
    self.interfaceController?.setRootTemplate(rootTemplate, animated: false)
}
```

```
func application(_ application: UIApplication,
                 didConnectCarInterfaceController interfaceController: CPInterfaceController,
                 to window: UIWindow) {

    self.interfaceController = interfaceController
    self.carWindow = window

    let rootViewController = MyRootViewController()
    window.rootViewController = rootViewController

    let rootTemplate: CPMapTemplate = createRootTemplate()
    self.interfaceController?.setRootTemplate(rootTemplate, animated: false)
}
```

```
func application(_ application: UIApplication,
                 didConnectCarInterfaceController interfaceController: CPInterfaceController,
                 to window: UIWindow) {

    self.interfaceController = interfaceController
    self.carWindow = window

    let rootViewController = MyRootViewController()
    window.rootViewController = rootViewController

    let rootTemplate: CPMapTemplate = createRootTemplate()
    self.interfaceController?.setRootTemplate(rootTemplate, animated: false)
}
```

```
func application(_ application: UIApplication,
                 didConnectCarInterfaceController interfaceController: CPInterfaceController,
                 to window: UIWindow) {

    self.interfaceController = interfaceController
    self.carWindow = window

    let rootViewController = MyRootViewController()
    window.rootViewController = rootViewController

    let rootTemplate: CPMapTemplate = createRootTemplate()
    self.interfaceController?.setRootTemplate(rootTemplate, animated: false)
}
```


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Asteroid Avenue
Saturn Street
BACK

Map Template

Key responsibilities

Panning

Navigation alerts

Guidance

Map Template

Key responsibilities

Panning

Navigation alerts

Guidance


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Asteroid Avenue
Saturn Street
BACK


Done
9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar Lawn
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar Lawn
Asteroid Avenue
Saturn Street
BACK


9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar Lawn
Asteroid Avenue
Saturn Street
BACK

```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```

```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```

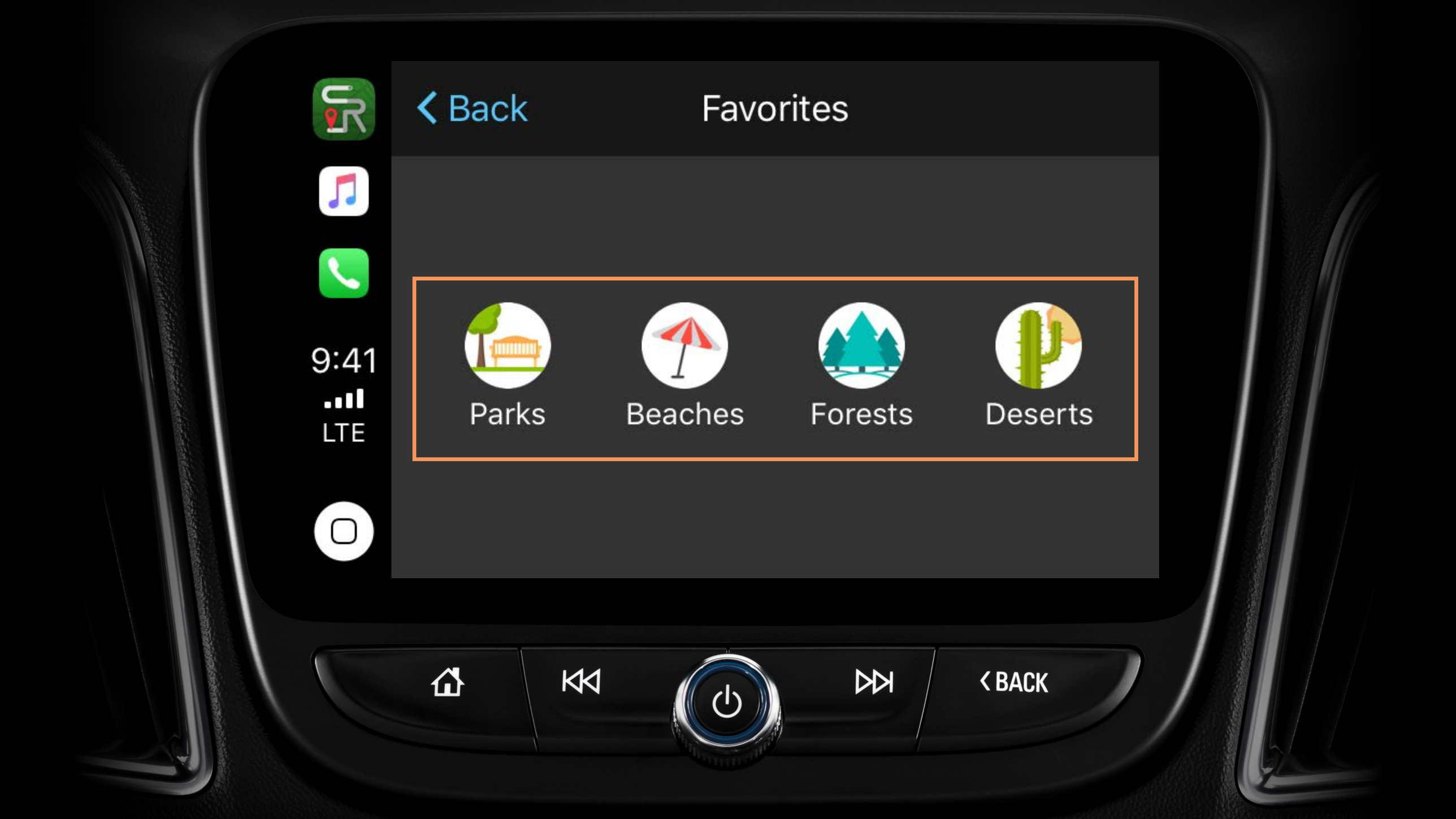
```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```

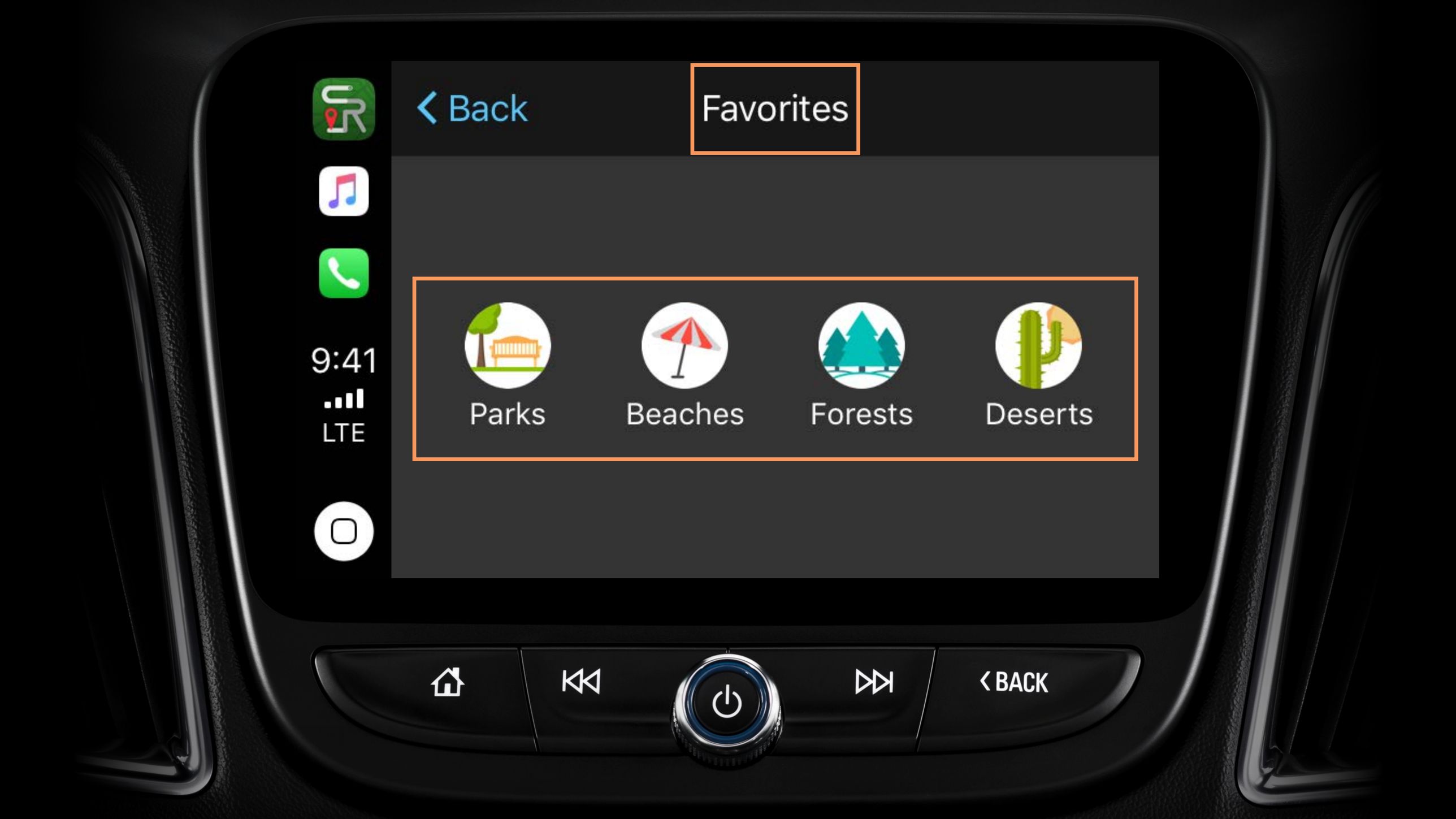
```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```

```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```

```
func createRootTemplate() -> CPMapTemplate {
    let mapTemplate = CPMapTemplate()
    ...
    let categorySearchButton = CPBarButton(type: .image) { [weak self] _ in
        self?.displayFavoriteCategories()
    }

    guard let favoritesImage = self.getButtonImage(buttonName: "Favorites") else {
        return
    }
    categorySearchButton.image = favoritesImage

    mapTemplate.trailingNavigationBarButtons = [trafficButton, categorySearchButton]

    return mapTemplate
}
```


Back
Favorites
9:41
LTE
Parks
Beaches
Forests
Deserts
BACK


Back
Favorites
9:41
LTE
Parks
Beaches
Forests
Deserts
BACK


Back
Favorites
Parks
Beaches
Forests
Deserts
9:41
LTE
BACK

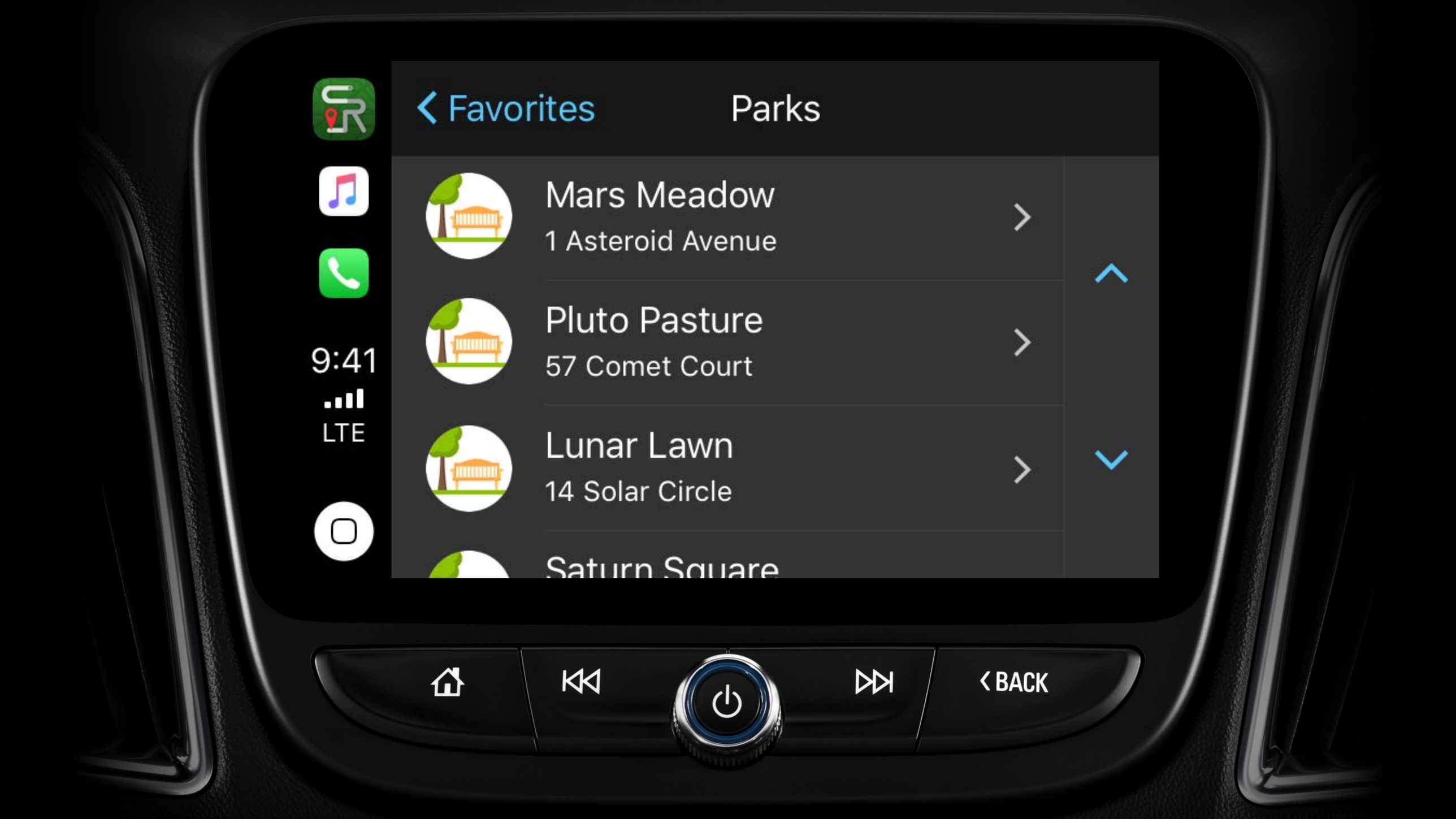
```
func displayFavoriteCategories() {
    guard let image = self.getCategoryImage(categoryName: "Parks") else {
        return
    }

    let parksButton = CPGridButton(titleVariants: ["Parks"], image: image) { [weak self] _ in
        self?.searchForNearbyParks()
    }

    ...

    let buttons = [parksButton, beachesButton, forestsButton, desertsButton]
    let template = CPGridTemplate(title: "Favorites", gridButtons: buttons)

    interfaceController?.pushTemplate(template, animated: true)
}
```

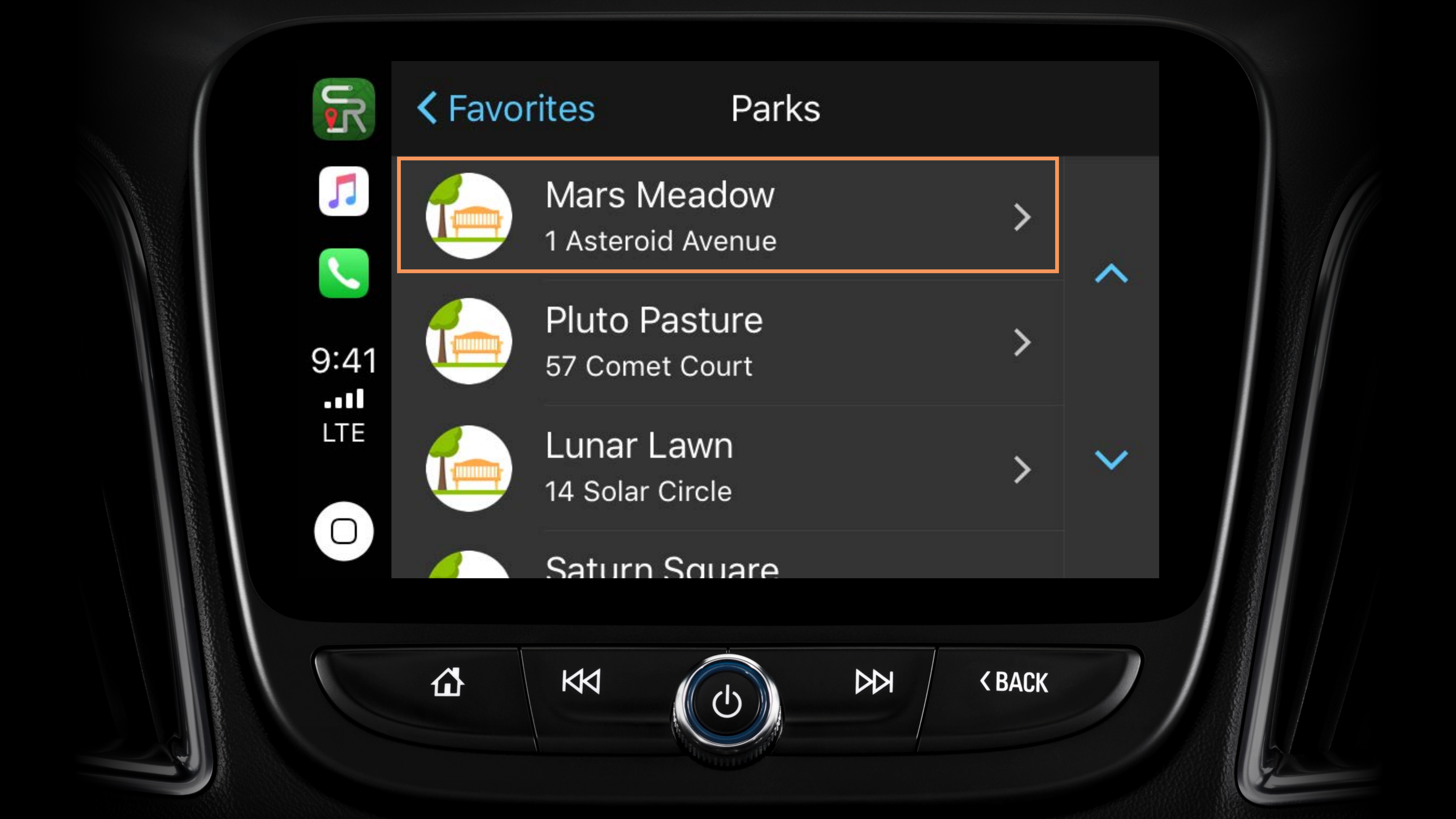
```
func displayFavoriteCategories() {
    guard let image = self.getCategoryImage(categoryName: "Parks") else {
        return
    }

    let parksButton = CPGridButton(titleVariants: ["Parks"], image: image) { [weak self] _ in
        self?.searchForNearbyParks()
    }

    ...

    let buttons = [parksButton, beachesButton, forestsButton, desertsButton]
    let template = CPGridTemplate(title: "Favorites", gridButtons: buttons)

    interfaceController?.pushTemplate(template, animated: true)
}
```

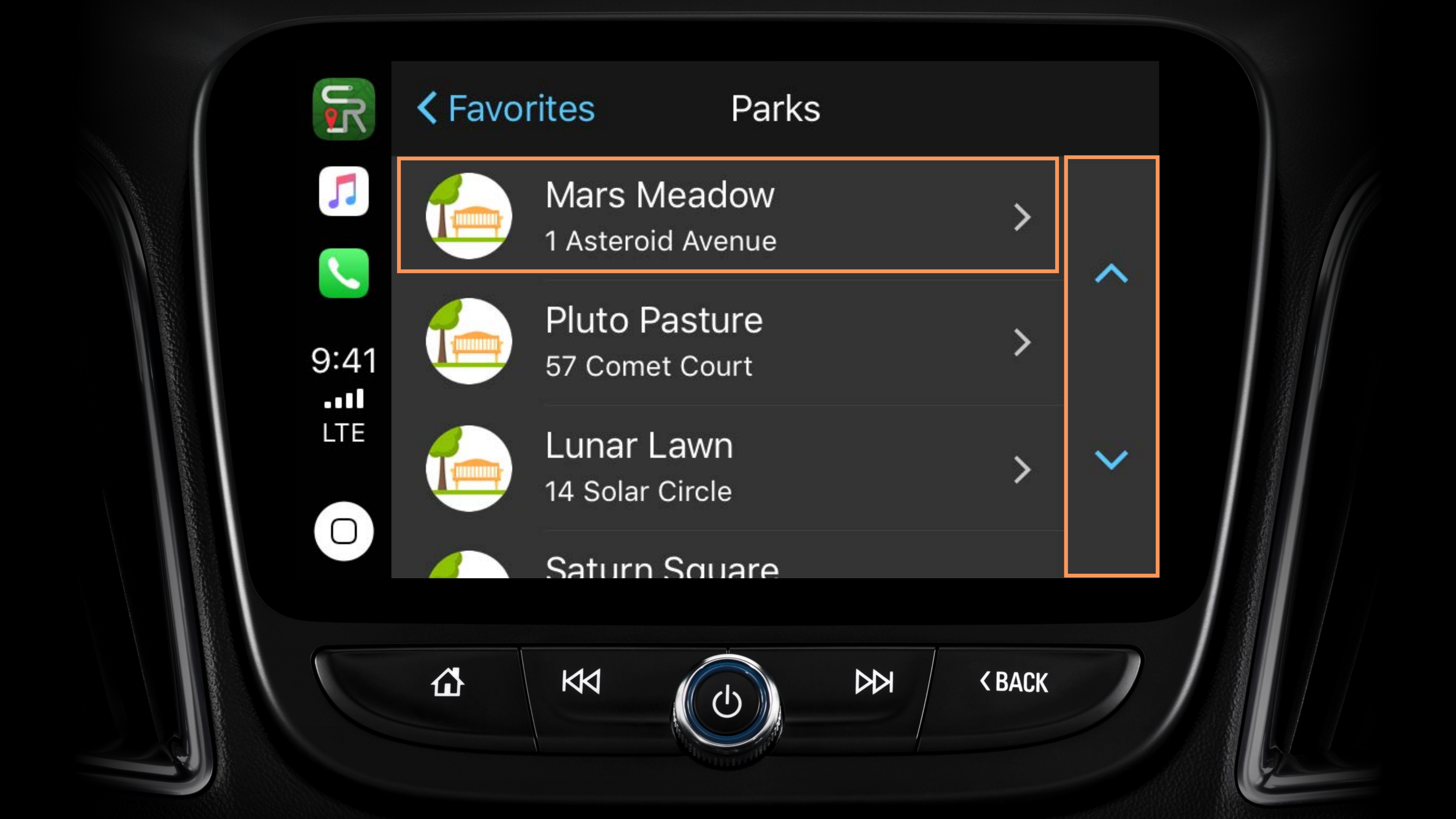
```
func displayFavoriteCategories() {
    guard let image = self.getCategoryImage(categoryName: "Parks") else {
        return
    }

    let parksButton = CPGridButton(titleVariants: ["Parks"], image: image) { [weak self] _ in
        self?.searchForNearbyParks()
    }

    ...

    let buttons = [parksButton, beachesButton, forestsButton, desertsButton]
    let template = CPGridTemplate(title: "Favorites", gridButtons: buttons)

    interfaceController?.pushTemplate(template, animated: true)
}
```

```
func displayFavoriteCategories() {
    guard let image = self.getCategoryImage(categoryName: "Parks") else {
        return
    }

    let parksButton = CPGridButton(titleVariants: ["Parks"], image: image) { [weak self] _ in
        self?.searchForNearbyParks()
    }

    ...

    let buttons = [parksButton, beachesButton, forestsButton, desertsButton]
    let template = CPGridTemplate(title: "Favorites", gridButtons: buttons)

    interfaceController?.pushTemplate(template, animated: true)
}
```

```
func displayFavoriteCategories() {
    guard let image = self.getCategoryImage(categoryName: "Parks") else {
        return
    }

    let parksButton = CPGridButton(titleVariants: ["Parks"], image: image) { [weak self] _ in
        self?.searchForNearbyParks()
    }

    ...

    let buttons = [parksButton, beachesButton, forestsButton, desertsButton]
    let template = CPGridTemplate(title: "Favorites", gridButtons: buttons)

    interfaceController?.pushTemplate(template, animated: true)
}
```


Favorites
Parks
Mars Meadow
1 Asteroid Avenue
Pluto Pasture
57 Comet Court
Lunar Lawn
14 Solar Circle
Saturn Square
9:41
LTE
BACK


Favorites
Parks
Mars Meadow
1 Asteroid Avenue
Pluto Pasture
57 Comet Court
Lunar Lawn
14 Solar Circle
Saturn Square
9:41
LTE
BACK


Favorites
Parks
Mars Meadow
1 Asteroid Avenue
Pluto Pasture
57 Comet Court
Lunar Lawn
14 Solar Circle
Saturn Square
9:41
LTE
BACK

```
func displayNearbyParks(with results: [SearchResult]) {
    let listItems = results.map { result in
        return CPListItem(text: result.name, detailText: result.address, image: result.image)
    }
    let section = CPListSection(items: listItems)

    let listTemplate = CPListTemplate(sections: [section])
    listTemplate.title = "Parks"
    listTemplate.delegate = self

    interfaceController?.pushTemplate(listTemplate, animated: true)
}

func listTemplate(_ listTemplate: CPListTemplate, didSelect item: CPListItem,
                  completionHandler: @escaping () -> Void) {
   ...
}
```

```
func displayNearbyParks(with results: [SearchResult]) {
    let listItems = results.map { result in
        return CPListItem(text: result.name, detailText: result.address, image: result.image)
    }
    let section = CPListSection(items: listItems)

    let listTemplate = CPListTemplate(sections: [section])
    listTemplate.title = "Parks"
    listTemplate.delegate = self

    interfaceController?.pushTemplate(listTemplate, animated: true)
}

func listTemplate(_ listTemplate: CPListTemplate, didSelect item: CPListItem,
                  completionHandler: @escaping () -> Void) {
   ...
}
```

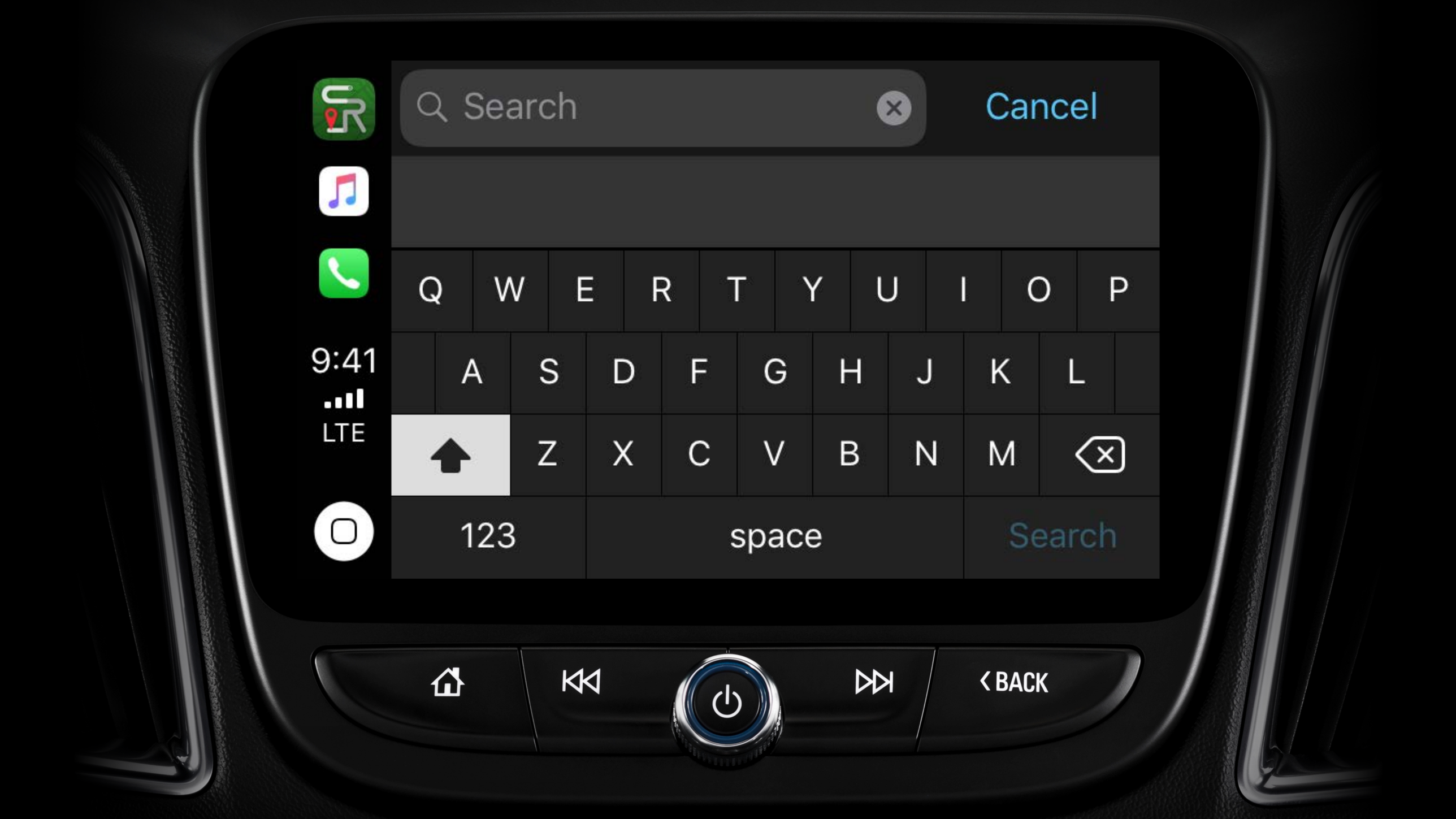
```
func displayNearbyParks(with results: [SearchResult]) {
    let listItems = results.map { result in
        return CPListItem(text: result.name, detailText: result.address, image: result.image)
    }
    let section = CPListSection(items: listItems)

    let listTemplate = CPListTemplate(sections: [section])
    listTemplate.title = "Parks"
    listTemplate.delegate = self

    interfaceController?.pushTemplate(listTemplate, animated: true)
}

func listTemplate(_ listTemplate: CPListTemplate, didSelect item: CPListItem,
                  completionHandler: @escaping () -> Void) {
   ...
}
```

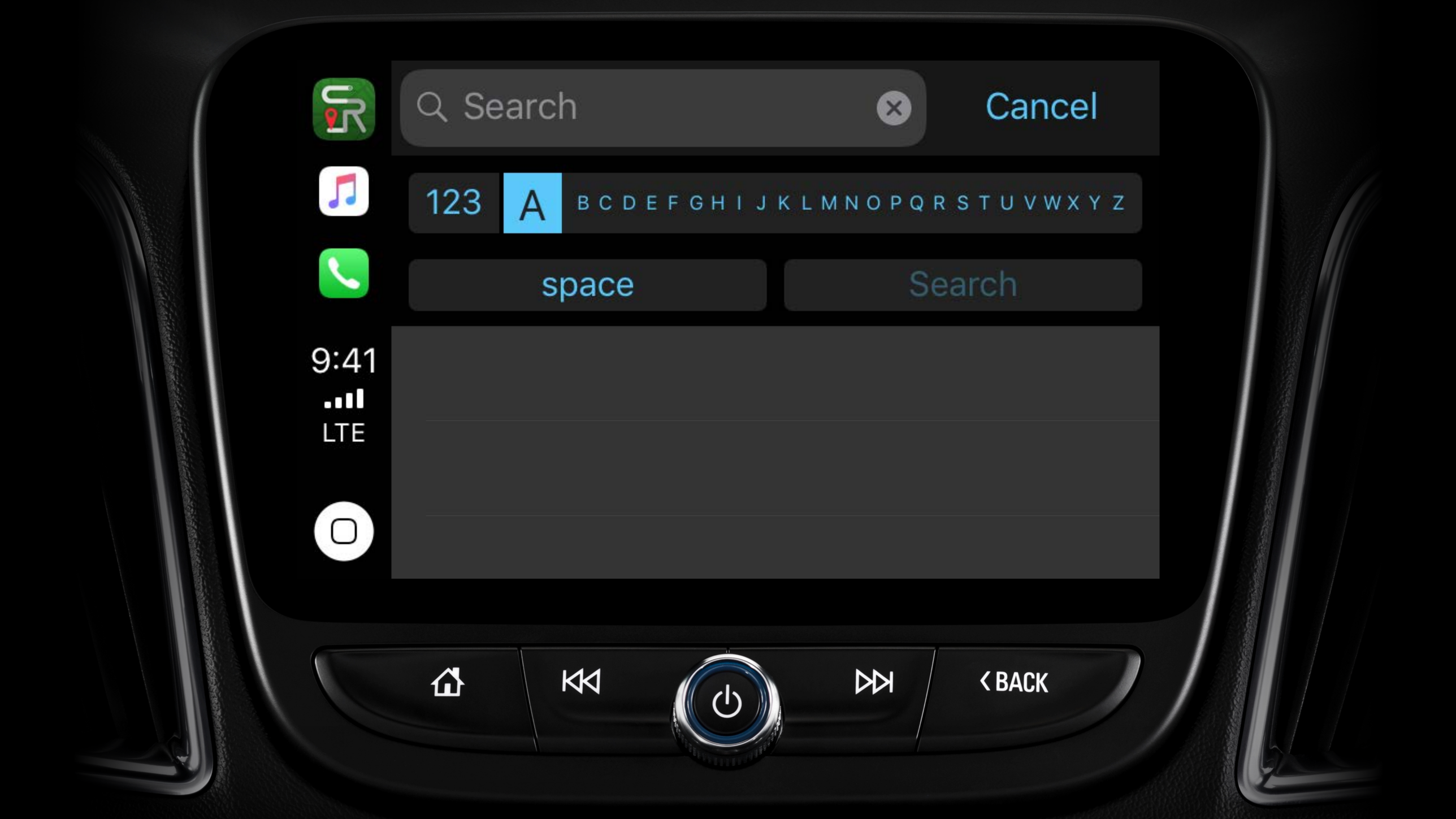
```
func displayNearbyParks(with results: [SearchResult]) {
    let listItems = results.map { result in
        return CPListItem(text: result.name, detailText: result.address, image: result.image)
    }
    let section = CPListSection(items: listItems)

    let listTemplate = CPListTemplate(sections: [section])
    listTemplate.title = "Parks"
    listTemplate.delegate = self

    interfaceController?.pushTemplate(listTemplate, animated: true)
}

func listTemplate(_ listTemplate: CPListTemplate, didSelect item: CPListItem,
                  completionHandler: @escaping () -> Void) {
   ...
}
```

```
func displayNearbyParks(with results: [SearchResult]) {
    let listItems = results.map { result in
        return CPListItem(text: result.name, detailText: result.address, image: result.image)
    }
    let section = CPListSection(items: listItems)

    let listTemplate = CPListTemplate(sections: [section])
    listTemplate.title = "Parks"
    listTemplate.delegate = self

    interfaceController?.pushTemplate(listTemplate, animated: true)
}
```

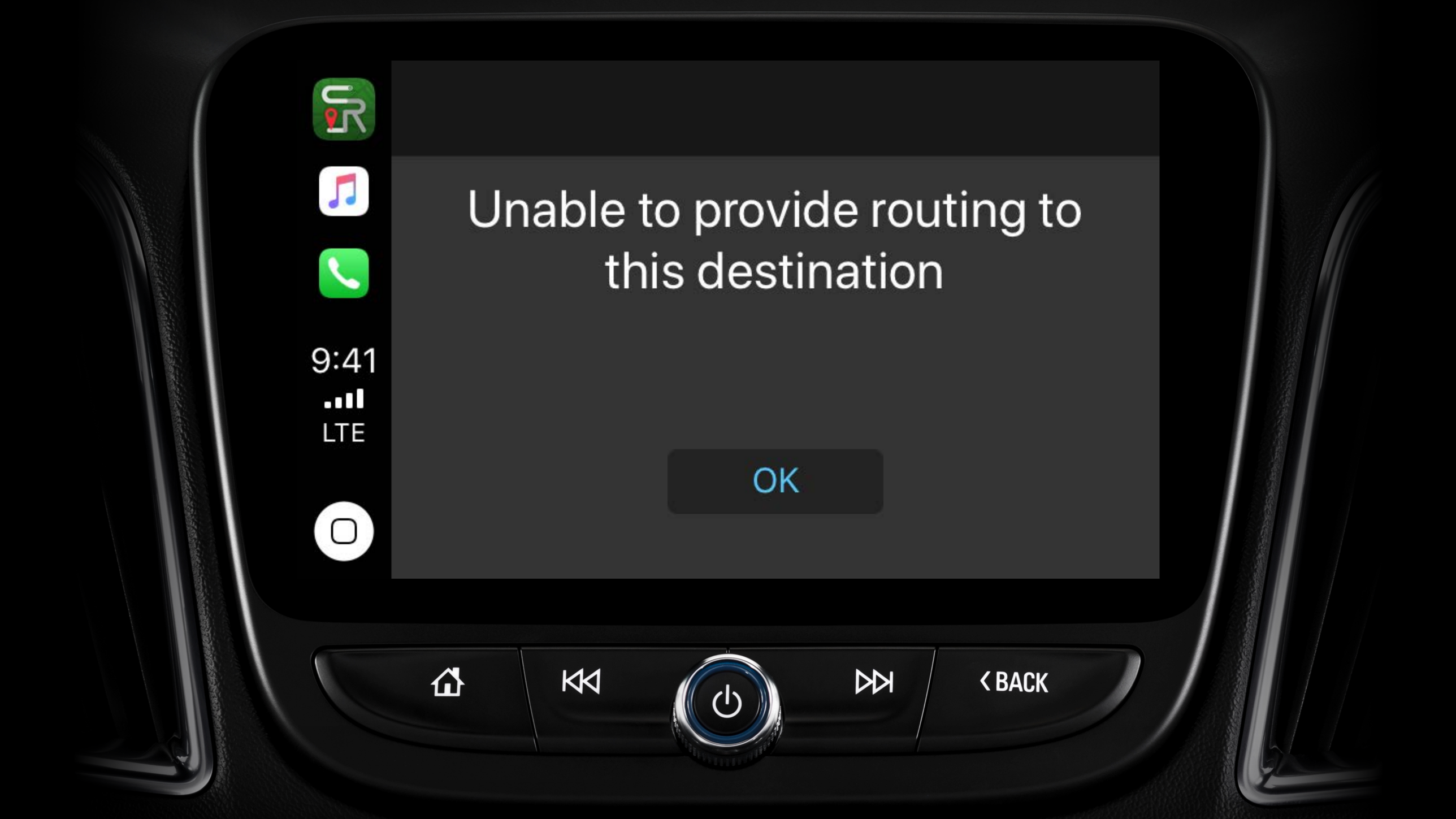
```
func listTemplate(_ listTemplate: CPListTemplate, didSelect item: CPListItem,
                  completionHandler: @escaping () -> Void) {
   ...
}
```

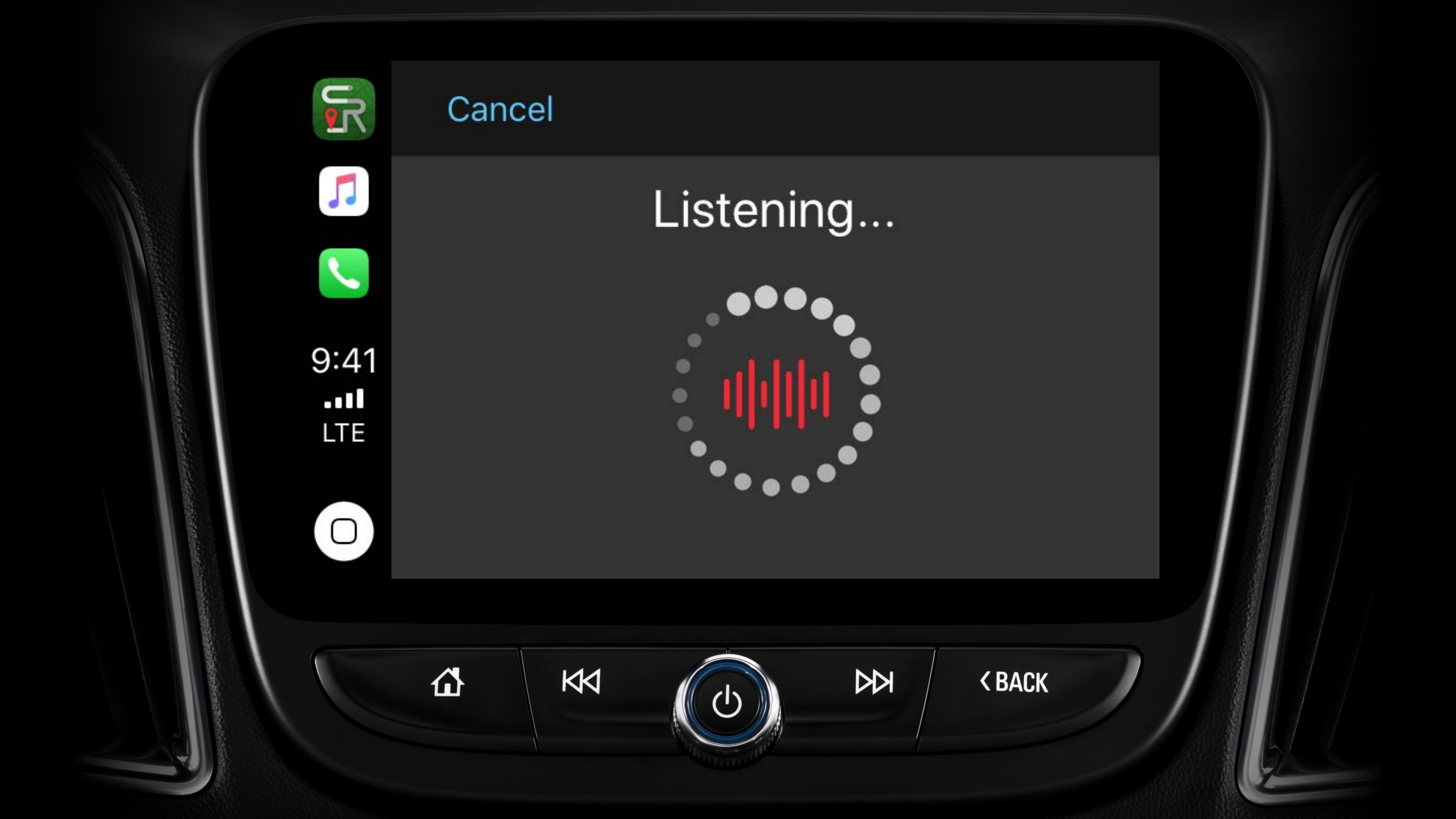

9:41
LTE
Pluto Pasture
Comet Court
Solar Circle
Lunar Lawn
Asteroid Avenue
Saturn Street
BACK


Search
Cancel
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
Search
9:41
LTE
BACK


Search
Cancel
123
A
B C D E F G H I J K L M N O P Q R S T U V W X Y Z
space
Search
9:41
LTE
BACK


9:41
LTE
Do you want to resume navigation?
Yes
No
BACK


Unable to provide routing to this destination
OK
9:41
LTE
BACK


Cancel
Listening...
9:41
LTE
BACK

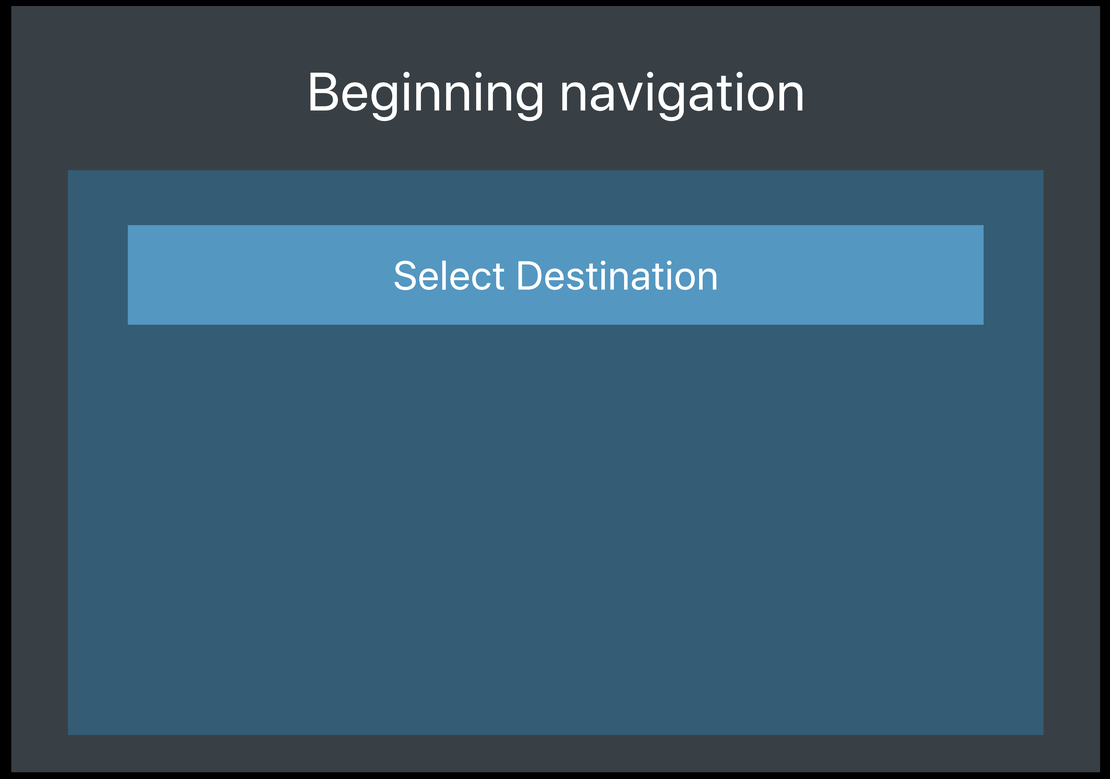
Demo

Jonathan Hersh, iOS Car Experience

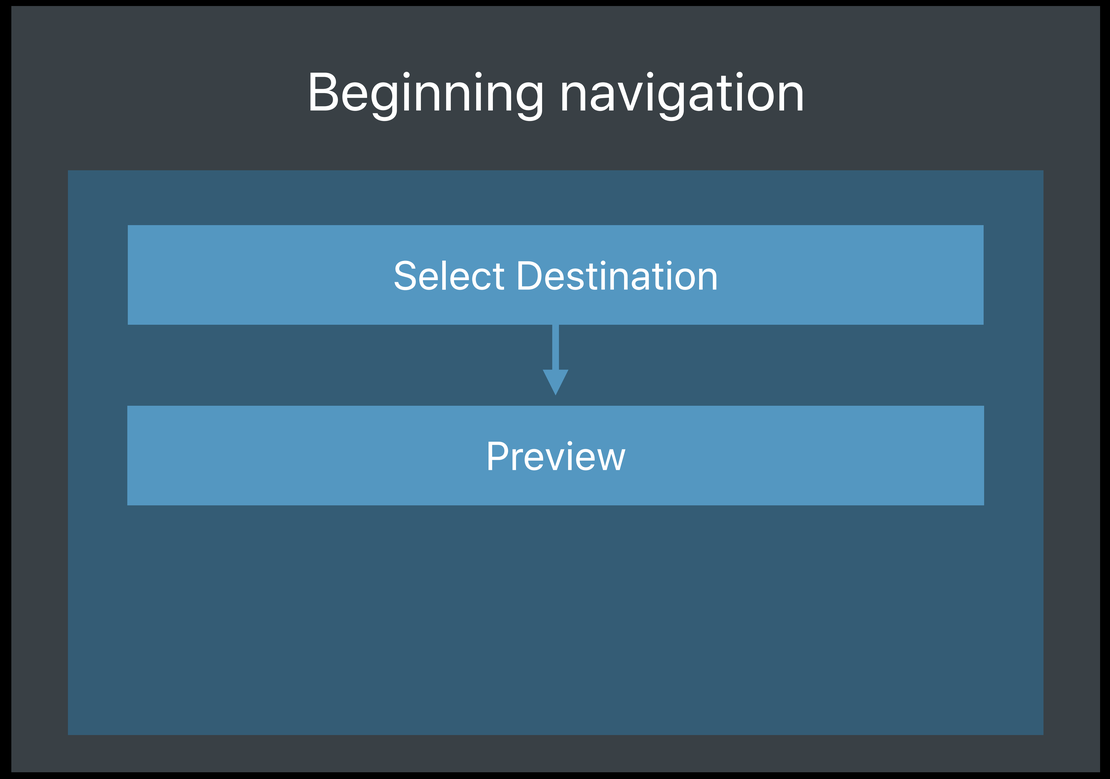
Guidance

Mike Knippers, iOS Car Experience

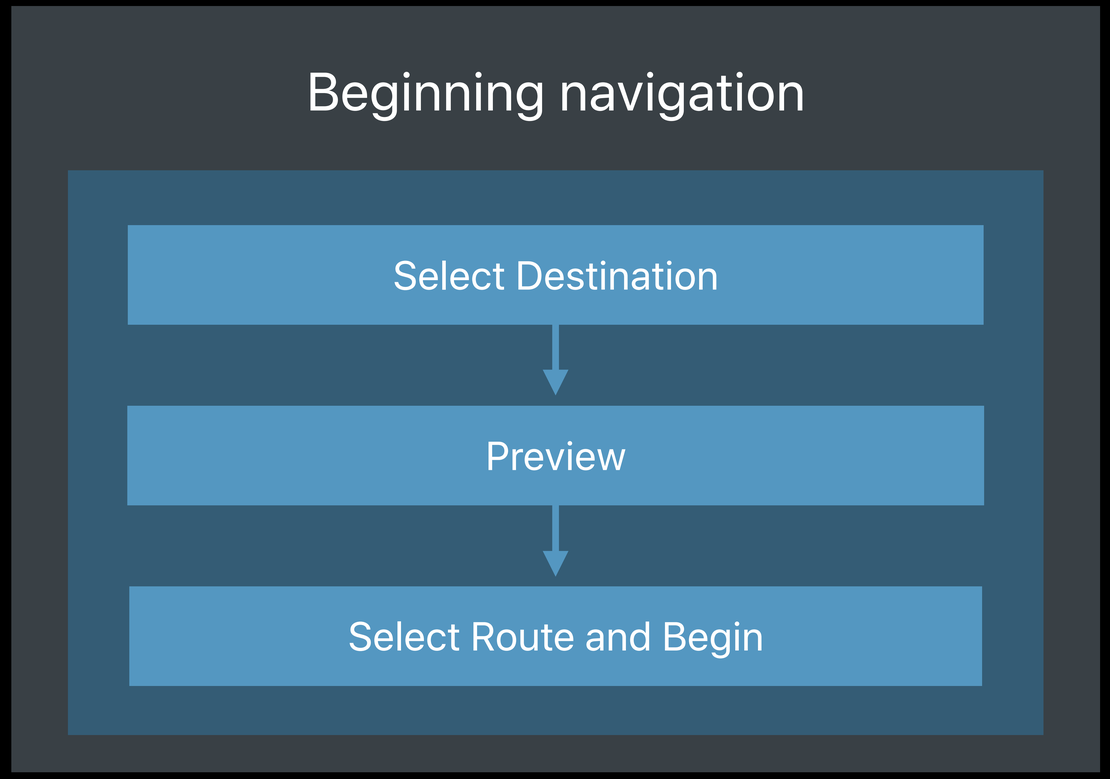
Guidance



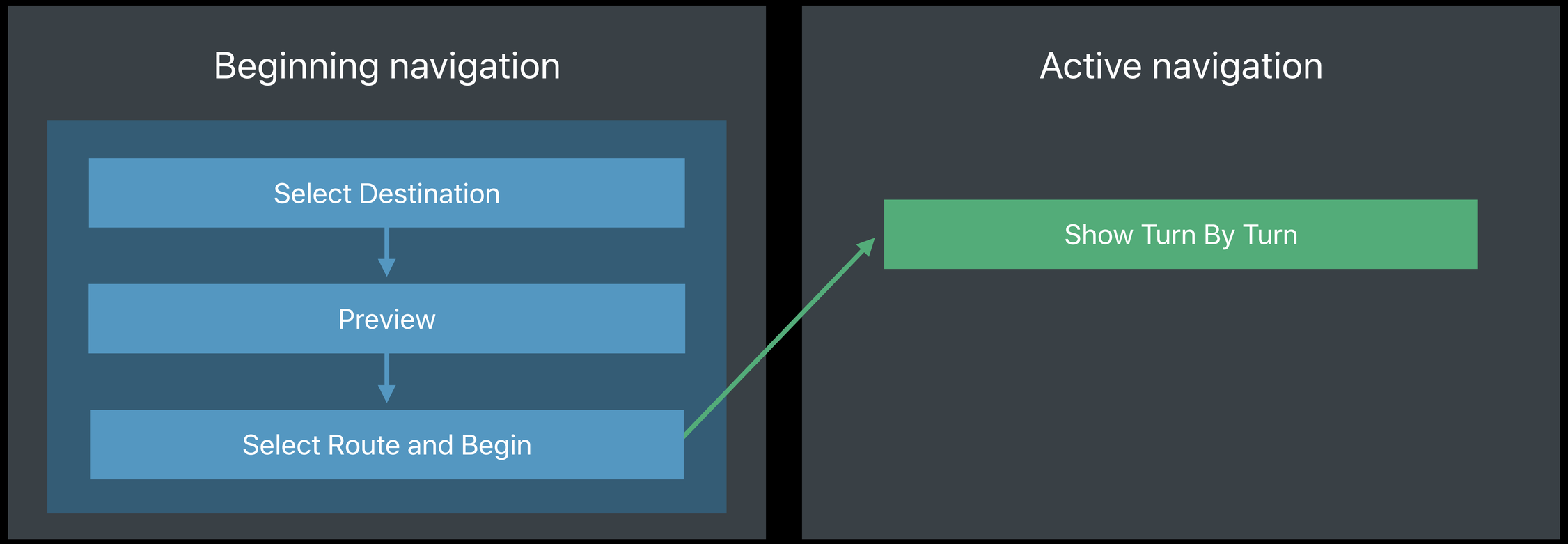
Guidance



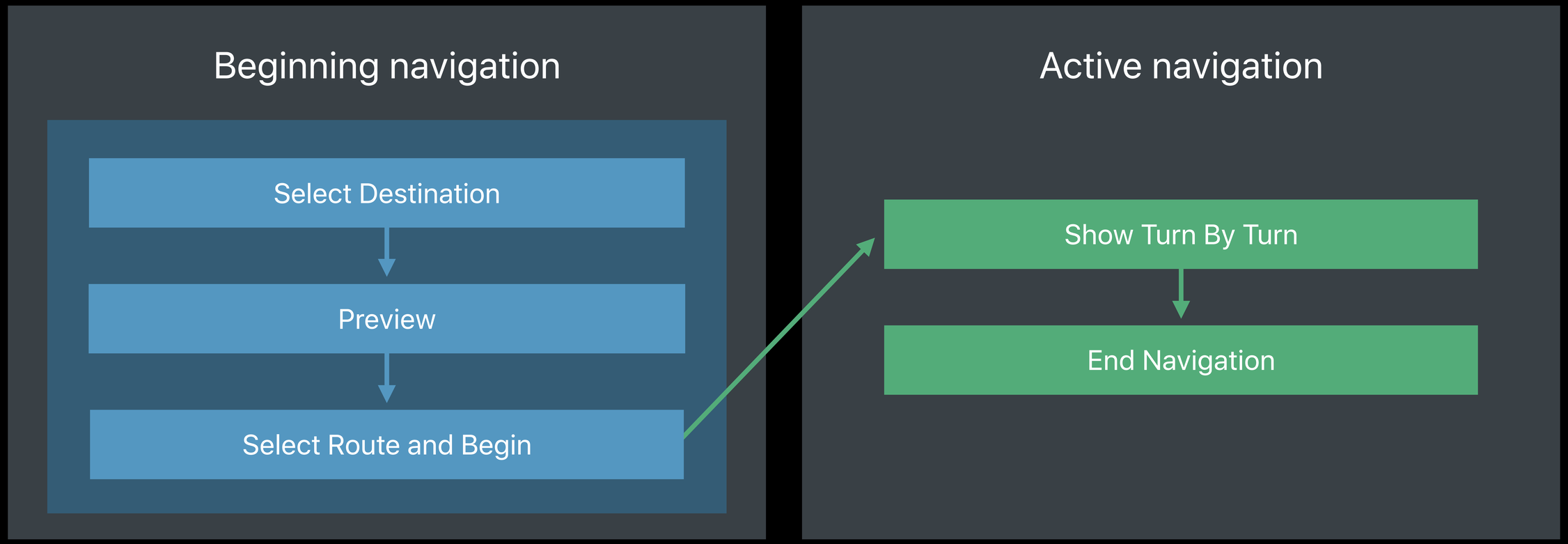
Guidance


Beginning navigation
Select Destination
Preview
Select Route and Begin

Guidance


Beginning navigation
Select Destination
Preview
Select Route and Begin
Active navigation
Show Turn By Turn

Guidance


Beginning navigation
Select Destination
Preview
Select Route and Begin
Active navigation
Show Turn By Turn
End Navigation


Cancel
Mars Meadow
Possible meteor shower.
9:47
arrival
6
min
2.8
mi
More Routes
Go
Comet Court
Solar Circle
Lunar Lawn
Saturn Street
9:41
LTE
BACK


Cancel
Mars Meadow
Possible meteor shower.
9:47
arrival
6
min
2.8
mi
More Routes
Go
Comet Court
Solar Circle
Lunar Lawn
Saturn Street
9:41
LTE
BACK

Route Preview

Classes and methods

```
class CPTrip, class CPRouteChoice, class CPTravelEstimates
// CPMapTemplate
func showTripPreviews(_ tripPreviews: [CPTrip])

// CPMapTemplateDelegate
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 selectedPreviewFor trip: CPTrip,
                 using routeChoice: CPRouteChoice)
// UIView
func safeAreaInsetsDidChange()
```


0.4 mi
Turn Right on
Solar Circle
Comet Court
9:41
LTE
9:47
arrival
6
min
1.4
mi
BACK

Guidance

Classes and methods

```
protocol CPMapTemplateDelegate, class CPMapTemplate, class CPNavigationSession
// CPMapTemplateDelegate
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip,
                 using routeChoice: CPRouteChoice)
// CPMapTemplate
func startNavigationSession(for trip: CPTrip) -> CPNavigationSession
```

Guidance

Classes and methods

```
class CPNavigationSession, class CPManeuver, class CPTravelEstimates

// CPNavigationSession
var upcomingManeuvers: [CPManeuver]
func update(_ estimates: CPTravelEstimates, for maneuver: CPManeuver)
// AVAudioSession
let AVAudioSessionModeVoicePrompt: String
static var duckOthers: AVAudioSessionCategoryOptions
static var interruptSpokenAudioAndMixWithOthers: AVAudioSessionCategoryOptions
```

Guidance

What happens next?

Set upcoming maneuvers and travel estimates

Set pause reason

Navigation alerts

End Navigation


0.4 mi
Turn Right on Solar Circle
Phone
Music
Maps
Messages
9:41
LTE
Now Playing
Podcasts
Audiobooks
CountryRoads
BACK

Notifications

```
// CPMapTemplateDelegate
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 shouldShowNotificationFor
                 maneuver: CPManeuver) -> Bool

func mapTemplate(_ mapTemplate: CPMapTemplate,
                 shouldUpdateNotificationFor maneuver: CPManeuver,
                 with travelEstimates: CPTravelEstimates) -> Bool

func mapTemplate(_ mapTemplate: CPMapTemplate,
                 shouldShowNotificationFor
                 navigationAlert: CPNavigationAlert) -> Bool
```

Demo

Jonathan Hersh, iOS Car Experience

```
guard let destination = item.userInfo as? MKMapItem else { completionHandler(); return }

let currentLocation = MKMapItem.forCurrentLocation()
let routeChoice = CPRouteChoice(summaryVariants: ["Take Solar Circle."],
                                additionalInformationVariants: ["Traffic is light."])
let trip = CPTrip(origin: currentLocation,
                  destination: destination,
                  routeChoices: [routeChoice])

mapTemplate.showTripPreviews([trip])
let estimates = CPTravelEstimates(distanceRemaining: Measurement(value: 1500,
                                                                 unit: .meters),
                                  timeRemaining: 300)

mapTemplate.updateEstimates(estimates, for: trip)
```

```
guard let destination = item.userInfo as? MKMapItem else { completionHandler(); return }

let currentLocation = MKMapItem.forCurrentLocation()
let routeChoice = CPRouteChoice(summaryVariants: ["Take Solar Circle."],
                                additionalInformationVariants: ["Traffic is light."])
let trip = CPTrip(origin: currentLocation,
                  destination: destination,
                  routeChoices: [routeChoice])

mapTemplate.showTripPreviews([trip])
let estimates = CPTravelEstimates(distanceRemaining: Measurement(value: 1500,
                                                                 unit: .meters),
                                  timeRemaining: 300)

mapTemplate.updateEstimates(estimates, for: trip)
```

```
guard let destination = item.userInfo as? MKMapItem else { completionHandler(); return }

let currentLocation = MKMapItem.forCurrentLocation()
let routeChoice = CPRouteChoice(summaryVariants: ["Take Solar Circle."],
                                additionalInformationVariants: ["Traffic is light."])
let trip = CPTrip(origin: currentLocation,
                  destination: destination,
                  routeChoices: [routeChoice])

mapTemplate.showTripPreviews([trip])
let estimates = CPTravelEstimates(distanceRemaining: Measurement(value: 1500,
                                                                 unit: .meters),
                                  timeRemaining: 300)

mapTemplate.updateEstimates(estimates, for: trip)
```

```
guard let destination = item.userInfo as? MKMapItem else { completionHandler(); return }

let currentLocation = MKMapItem.forCurrentLocation()
let routeChoice = CPRouteChoice(summaryVariants: ["Take Solar Circle."],
                                additionalInformationVariants: ["Traffic is light."])
let trip = CPTrip(origin: currentLocation,
                  destination: destination,
                  routeChoices: [routeChoice])

mapTemplate.showTripPreviews([trip])
let estimates = CPTravelEstimates(distanceRemaining: Measurement(value: 1500,
                                                                 unit: .meters),
                                  timeRemaining: 300)

mapTemplate.updateEstimates(estimates, for: trip)
```

```
guard let destination = item.userInfo as? MKMapItem else { completionHandler(); return }

let currentLocation = MKMapItem.forCurrentLocation()
let routeChoice = CPRouteChoice(summaryVariants: ["Take Solar Circle."],
                                additionalInformationVariants: ["Traffic is light."])
let trip = CPTrip(origin: currentLocation,
                  destination: destination,
                  routeChoices: [routeChoice])

mapTemplate.showTripPreviews([trip])
let estimates = CPTravelEstimates(distanceRemaining: Measurement(value: 1500,
                                                                 unit: .meters),
                                  timeRemaining: 300)

mapTemplate.updateEstimates(estimates, for: trip)
```

```
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip, using routeChoice: CPRouteChoice) {

    mapTemplate.hideTripPreviews()
    let session = mapTemplate.startNavigationSession(for: trip)
    session.pauseTrip(for: .loading)

    guard let selectedRoute = routeChoice.userInfo as? CountryRoute else { return }

    let maneuvers: [CPManeuver] = selectedRoute.steps.map { step in
        let maneuver = CPManeuver()
        maneuver.instructionVariants = [step.instructions]
        maneuver.distanceFromPreviousManeuver = Measurement(value: step.distance,
                                                            unit: .feet)
        return maneuver
    }
    session.upcomingManeuvers = [maneuvers.first]
}
```

```
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip, using routeChoice: CPRouteChoice) {

    mapTemplate.hideTripPreviews()
    let session = mapTemplate.startNavigationSession(for: trip)
    session.pauseTrip(for: .loading)

    guard let selectedRoute = routeChoice.userInfo as? CountryRoute else { return }

    let maneuvers: [CPManeuver] = selectedRoute.steps.map { step in
        let maneuver = CPManeuver()
        maneuver.instructionVariants = [step.instructions]
        maneuver.distanceFromPreviousManeuver = Measurement(value: step.distance,
                                                            unit: .feet)
        return maneuver
    }
    session.upcomingManeuvers = [maneuvers.first]
}
```

```
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip, using routeChoice: CPRouteChoice) {

    mapTemplate.hideTripPreviews()
    let session = mapTemplate.startNavigationSession(for: trip)
    session.pauseTrip(for: .loading)

    guard let selectedRoute = routeChoice.userInfo as? CountryRoute else { return }

    let maneuvers: [CPManeuver] = selectedRoute.steps.map { step in
        let maneuver = CPManeuver()
        maneuver.instructionVariants = [step.instructions]
        maneuver.distanceFromPreviousManeuver = Measurement(value: step.distance,
                                                            unit: .feet)
        return maneuver
    }
    session.upcomingManeuvers = [maneuvers.first]
}
```

```
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip, using routeChoice: CPRouteChoice) {

    mapTemplate.hideTripPreviews()
    let session = mapTemplate.startNavigationSession(for: trip)
    session.pauseTrip(for: .loading)

    guard let selectedRoute = routeChoice.userInfo as? CountryRoute else { return }

    let maneuvers: [CPManeuver] = selectedRoute.steps.map { step in
        let maneuver = CPManeuver()
        maneuver.instructionVariants = [step.instructions]
        maneuver.distanceFromPreviousManeuver = Measurement(value: step.distance,
                                                            unit: .feet)
        return maneuver
    }
    session.upcomingManeuvers = [maneuvers.first]
}
```

```
func mapTemplate(_ mapTemplate: CPMapTemplate,
                 startedTrip trip: CPTrip, using routeChoice: CPRouteChoice) {

    mapTemplate.hideTripPreviews()
    let session = mapTemplate.startNavigationSession(for: trip)
    session.pauseTrip(for: .loading)

    guard let selectedRoute = routeChoice.userInfo as? CountryRoute else { return }

    let maneuvers: [CPManeuver] = selectedRoute.steps.map { step in
        let maneuver = CPManeuver()
        maneuver.instructionVariants = [step.instructions]
        maneuver.distanceFromPreviousManeuver = Measurement(value: step.distance,
                                                            unit: .feet)
        return maneuver
    }
    session.upcomingManeuvers = [maneuvers.first]
}
```

CarPlay audio app improvements

Navigating with the new CarPlay framework

CarPlay navigation demo

More Information

https://developer.apple.com/wwdc18/213

CarPlay Lab	Technology Lab 11	Wednesday 2:00PM


WWDC18