Structuring Screen Transitions with Parent-Child View States

In this page, you define NavigationView as a parent Composite View, and inside it, you build a structure that switches between NavigationMain (a Scene with IsView=true) and RouteDetailsView (a Composite View) as child View States. What you learn here is that, unlike the sibling relationship in the previous page, a parent-child relationship is suited for "switching only a local state while maintaining the shared display." By switching between the normal display and the detail display while maintaining the map and shared buttons inside NavigationView, you understand the practical use of the parent-child structure in the View Tree. You also revisit the composition that combines a Composite View with a Scene that has IsView=true.

Creating Shared Scenes for Navigation

First, you prepare the elements that you want to display continuously while NavigationView is active. Here, you handle the map display and the Home button as shared parts, and incorporate them into the parent View definition later.

NavigationMap Scene

1. Create a new 2D scene and name it NavigationMap.

2. Select the camera in the Scene Tree and configure the following properties in the Properties panel:

Property Category	Item	Value
Camera	Sequence Number	1
	Clear Color Buffer	False
	Enable Swapping	False

3. Place a 2D Text Node from the Toolbox onto the Scene Tree of the NavigationMap scene.

Property Category	Item	Value
Transform	Position X	250
	Position Y	200
2D Text Node	Text	Navigation Map

NavigationViewButtons Scene

1. Create a new 2D scene and name it NavigationViewButtons.

2. Select the camera in the Scene Tree and configure the following properties in the Properties panel:

Property Category	Item	Value
Camera	Sequence Number	2
	Clear Color Buffer	False
	Enable Swapping	False

3. Place a TextButton control from the Toolbox onto the Scene Tree of the NavigationViewButtons scene.

Property Category	Item	Value
General	Name	JumpToHomeViewButton
Transform	Position X	350
	Position Y	300
Control	Text	Home

NavigationMap and NavigationViewButtons are elements that you want to remain visible while in NavigationView. Therefore, you incorporate them into the parent View definition later.

Creating Transition Scenes for Navigation

Next, you prepare the targets to switch between inside NavigationView. Here, you separate the normal display and the route details display so that you can treat them as child states of the parent View.

NavigationMain Scene

1. Create a new 2D scene and name it NavigationMain.

2. Select the camera in the Scene Tree and configure the following properties in the Properties panel:

Property Category	Item	Value
Camera	Sequence Number	2
	Clear Color Buffer	False
	Enable Swapping	False

3. Place a 2D Text Node from the Toolbox onto the Scene Tree of the NavigationMain scene.

Property Category	Item	Value
Transform	Position X	50
	Position Y	200
2D Text Node	Text	Navigation Main

4. Next, place a TextButton control and configure the following properties:

Property Category	Item	Value
General	Name	ShowDetailsButton
Transform	Position X	50
	Position Y	300
Control	Text	Show Details

RouteDetailsContent Scene

1. Create a new 2D scene and name it RouteDetailsContent.

2. Select the camera in the Scene Tree and configure the following properties in the Properties panel:

Property Category	Item	Value
Camera	Sequence Number	2
	Clear Color Buffer	False
	Enable Swapping	False

3. Place a 2D Text Node from the Toolbox onto the Scene Tree of the RouteDetailsContent scene.

Property Category	Item	Value
Transform	Position X	450
	Position Y	200
2D Text Node	Text	Route Details

NavigationBack Scene

1. Create a new 2D scene and name it NavigationBack.

2. Select the camera in the Scene Tree and configure the following properties in the Properties panel:

Property Category	Item	Value
Camera	Sequence Number	2
	Clear Color Buffer	False
	Enable Swapping	False

3. Place a TextButton control from the Toolbox onto the Scene Tree of the NavigationBack scene and configure the following properties:

Property Category	Item	Value
General	Name	BackToNavigationMainButton

Transform	Position X	50
	Position Y	300
Control	Text	Back

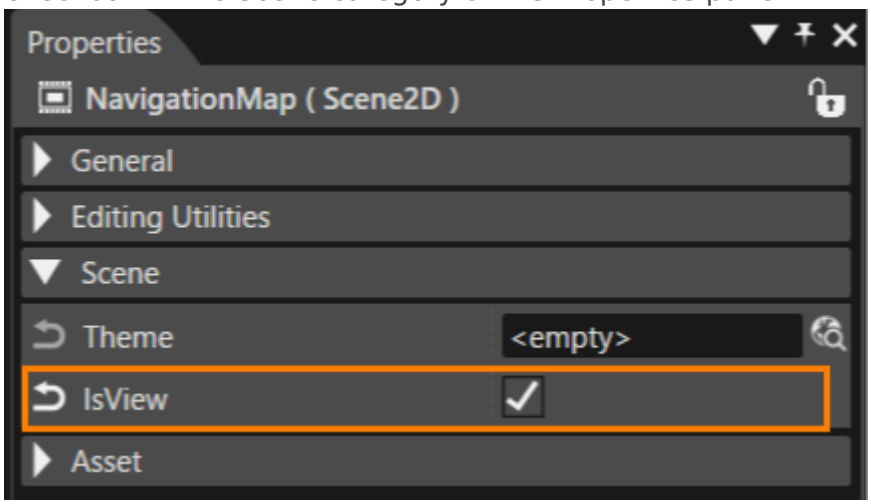
NavigationMain is for the normal display, and RouteDetailsContent and NavigationBack are components for the detail display. These are not shared displays — you treat them as targets to switch between inside NavigationView.

Creating Navigation-Related Views

Here, you use both the case of creating a simple View from an existing Scene and the case of creating a new View that combines multiple components. This distinction is also important when learning the parent-child structure.

Creating Views: Using Existing Scenes

1. Select each of the five scenes created in the previous steps and select the IsView checkbox in the Scene category of the Properties panel.



- NavigationMap
- NavigationViewButtons
- NavigationMain
- RouteDetailsContent
- NavigationBack

Creating the New RouteDetailsView

1. Create a new View and name it RouteDetailsView.

2. Double-click the created RouteDetailsView to open the View Definition Editor.

3. Add the following elements to the left pane of the View Definition Editor:
   - RouteDetailsContent
   - NavigationBack

RouteDetailsView is a screen state that bundles the detail display itself and the return operation. By extracting a local grouping like this as a Composite View, you make it easier to treat it as a child View State later.

Configuring NavigationView

Here, you bundle into the parent View only the elements that you want to remain displayed while in Navigation. It is important to include only the shared display in the parent View, and not the parts that you want to switch.

1. In Solution Explorer, select the Solution folder and select **Add > New Item** from the context menu.

2. In the **New Item** dialog, select View and name it NavigationView.

3. Double-click the created NavigationView to open the View Definition Editor.

4. Add the following elements to the left pane of the View Definition Editor:
   - Background
   - Header
   - NavigationMap
   - NavigationViewButtons

You do not include NavigationMain or RouteDetailsView in the NavigationView definition. You add these later as child View States in the View Tree, and treat them as targets to switch between under the parent. This separation clarifies the roles of the shared display and the local switching.

Configuring MainViewTree

Here, you add NavigationView as a top-level sibling View and attach child View States inside it. This lets you experience a two-layer structure — sibling relationships at the top level and a parent-child relationship inside Navigation.

1. Double-click MainViewTree to open the View Tree panel.

2. Drop NavigationView into the empty area below MainView and DetailView to add it as a top-level View State.

3. Drop NavigationMain onto the body of the parent NavigationView to add it as a child View State.

4. Drop RouteDetailsView onto the body of the parent NavigationView to add it as a child View State.

By dropping onto the body of the parent, the child states are treated as internal states of NavigationView while it is active. This is the essence of a parent-child relationship, as opposed to a sibling relationship.

Configuring Navigation-Related Transitions

Here, you configure both the transitions to top-level sibling View States and the switching of child View States under the parent. This shows that the same Jump To View takes on different meanings depending on the relationship of the View you jump to.

MainContent Scene

1. Double-click the MainContent scene to open the Scene Editor.

2. In the Scene Tree, select OpenNavigationButton.

3. Drag and drop a Jump To View behavior onto Action in the Extra Scene Tree.

4. Select the added Jump To View behavior and configure the following properties in the Properties panel:

Property Category	Item	Value
General	Name	JumpToNavigation
Asset	View	NavigationView

NavigationMain Scene

1. Double-click the NavigationMain scene to open the Scene Editor.

2. In the Scene Tree, select ShowDetailsButton.

3. Drag and drop a Jump To View behavior onto Action in the Extra Scene Tree.

4. Select the added Jump To View behavior and configure the following properties in the Properties panel:

Property Category	Item	Value
General	Name	JumpToRouteDetail
Asset	View	RouteDetailsView

NavigationBack Scene

1. Double-click the NavigationBack scene to open the Scene Editor.
2. In the Scene Tree, select BackToNavigationMainButton.
3. Drag and drop a Jump To View behavior onto Action in the Extra Scene Tree.
4. Select the added Jump To View behavior and configure the following properties in the Properties panel:

Property Category	Item	Value
General	Name	JumpToNavigationMain
Asset	View	NavigationMain

NavigationViewButtons Scene

1. Double-click the NavigationViewButtons scene to open the Scene Editor.
2. In the Scene Tree, select JumpToHomeViewButton.
3. Drag and drop a Jump To View behavior onto Action in the Extra Scene Tree.
4. Select the added Jump To View behavior and configure the following properties in the Properties panel:

Property Category	Item	Value
General	Name	JumpToMainView
Asset	View	MainView

The important point here is that, inside NavigationView, NavigationMain and RouteDetailsView switch between each other as child View States, while the Home button exits Navigation entirely and returns to MainView, the top-level sibling View State.

Verifying in the Player

The purpose of this verification is to see how the meaning of the parent-child relationship is reflected in the screen display. Verify that the shared display portion is maintained while only the child state switches.

1. Save the solution and launch the Player.
   - Verify that MainView is displayed immediately after launch.

2. Click the Open Navigation button.
   - Verify that the screen switches to NavigationView.
   - At the same time, verify that Background and Header are displayed.
   - In addition, verify that Navigation Main, Navigation Map, the Show Detail button, and the Home button are displayed.

3. Click the Show Details button.
   - Verify that the screen switches to Route Details.
   - At the same time, verify that Background and Header are displayed.
   - In addition, verify that Navigation Map and the Home button are displayed.

4. Click the Back button.
   - Verify that the screen returns to Navigation Main.

5. Click Home.
   - Verify that the screen returns to MainView.

In this page, you built a structure that uses NavigationView as the parent View State and switches between child View States inside it. This let you confirm in practice that a sibling relationship is suited for exclusive switching of entire screens, while a parent-child relationship is suited for switching only a local state while retaining the shared display. In the next page, you add conditional automatic transitions to DetailView using the Action Chain Editor and the Condition Editor.

Revision #3
Created 2026-04-21 00:50:48 UTC by Tsuyoshi.Kato
Updated 2026-06-18 06:05:50 UTC by Tsuyoshi.Kato