Screen Transitions with View Tree

In this page, you use the basic Views prepared in the previous page to compose two screen-level Views, MainView and DetailView, and arrange them as siblings in the View Tree.

The central themes of this page are how to assemble a single screen in the View Definition Editor and why a sibling relationship in the View Tree is suited for mutually exclusive screen switching. You also configure Jump To View on a button and verify in the Player that MainView and DetailView switch without overlapping. This page is the core part for learning the basics of the View Editor — defining screen-level Views and using sibling View transitions.

Creating MainView

Here, you bundle the background, the common header, and the main content into a View that represents the entire main screen. Proceed while understanding that the View Definition Editor is "the place to assemble a screen-level View."

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 MainView.

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

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

MainView is a View that bundles the background, the common header, and the contents of the main screen into a single screen state. It is also important that you reuse Header as an existing View — this teaches the idea of extracting shared components as separate Views and reusing them.

Creating DetailView

Next, you define a View that represents the entire detail screen. As with MainView, you use the background and header as shared components, and only the content portion differs. This lets multiple screens share the same visual framework while only switching their content.

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 DetailView.

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

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

DetailView is also an independent View that represents the entire screen. Because it shares the same background and header as MainView and only replaces the content, you can easily perform screen-level switching in the View Tree later.

Configuring MainViewTree

Here, you place MainView and DetailView in the View Tree. The most important point is to make the two a sibling relationship. A sibling relationship is suited for switching between independent screen states.

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

2. In the **New Item** dialog, select ViewTree and name it MainViewTree.

3. Double-click the created MainViewTree to open the View Tree panel.

4. Drag and drop MainView onto the View Tree panel to add it as a top-level View State.

5. In the same way, drag and drop DetailView into the empty area below MainView to add it as a top-level View State.

Verifying the Sibling Relationship in the View Tree

- MainView and DetailView are at the same hierarchical level (DetailView is not a child of MainView).
- The two are targets to switch between exclusively.

> If you drop DetailView onto the body of MainView here, you create a parent-child relationship, and the resulting structure may unintentionally behave close to simultaneous display. Because you want to switch between the entire screens this time, always drop into the empty area below to place them as siblings.

Configuring Transitions Between Views

From here, you make it possible to move between the sibling Views defined in the View Tree using Jump To View. The key point is that you specify the completed screen-level View as the transition target, not a content View.

1. Double-click the MainContent scene to open the Scene Editor.
2. In the Scene Tree, select OpenDetailButton.
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	JumpToDetailView
Asset	View	DetailView

You specify DetailView (a Composite View that includes the background and header) — not DetailContent (a Scene with IsView=true on its own) — as the transition target because you want to switch at the screen level, including the background and header. The flow of treating a Composite View created in the View Definition Editor as a state in the View Tree connects here.

In the same way, configure the return transition from DetailView to MainView. This completes the round-trip switching between sibling Views.

1. Double-click the DetailContent scene to open the Scene Editor.
2. In the Scene Tree, select BackButton.
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

Verifying in the Player

Here, you verify not just the switching between MainView and DetailView, but also whether the Background's clear configuration is effective. Even when the transition itself succeeds, if the previous screen remains visible, the cause is more often on the Background side than in the View Tree.

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

2. Click the Open Detail button.
   - Verify that the screen switches to DetailView.

3. Click the Back button.
   - Verify that the screen returns to MainView.

In this page, you composed MainView and DetailView as screen-level Views in the View Definition Editor and placed them as siblings in the View Tree. You also implemented basic screen switching between sibling Views using Jump To View. In the next page, you learn a different use case — a parent-child relationship that switches only a local state while preserving the shared display.

Revision #2
Created 2026-04-14 07:12:37 UTC by Tsuyoshi.Kato
Updated 2026-06-18 06:05:08 UTC by Tsuyoshi.Kato