



BLACK IRON SPINE STUDIOS

RPG Maps & Compass

Complete Buyer Guide

Version 1.1 | Unreal Engine 5.8 | Windows

Contents

1. Start here
2. Map and owner setup
3. Controls and recovery
4. View and surface controls
5. Waypoints and UI events
6. Settings and progress
7. Cancel and cleanup
8. Direct Blueprint calls
9. UI fields and landmarks
10. Native capture settings
11. C++ integration
12. Included examples
13. API and field reference
14. Scope and first-test checklist

Complete Blueprint captures are embedded at full source resolution. Zoom the PDF or use the HTML guide to open each original. The HTML copy also provides one-click copying of function names.

1. Start here

Overview

RPG Maps & Compass adds a world map, minimap, compass, landmarks, user waypoints and optional exploration fog to an existing Unreal Engine project. Version 1.1 includes editable Blueprint integration examples and public native interfaces.

Use an authored overview image or the documented native capture route. Your game retains ownership of quests, rewards, player input and its finished UI.

How buyer integration works

Resolve navigation from the exact owning local player. Connect once, submit requests, branch on returned results, and query current state from notifications. Keep the stable waypoint ID for rename/remove. Settings and exploration progress have separate save receipts.

The optional BIS UI host uses its existing MAP dock adapter. Other BIS products are not required for standalone Maps integration.

Choose your route

Start with the shipped wrapper setup. Use Controls and recovery during play; choose Direct Blueprint route for your own UI, Native capture route for generated imagery, or C++ for your game module. The API reference documents fields without implying untested runtime coverage.

First successful test

1. Copy the single RPGMapsCompass folder into your existing project's Plugins folder. Enable it and rebuild/restart when prompted.
2. Play with a possessed local pawn. Confirm the minimap and launcher. Use your existing M/map input to open the world map.
3. Follow Map and owner setup to duplicate the editable region example, assign your own overview and bounds, and pass your exact local controller into Start Buyer Navigation.
4. Call Request Buyer Waypoint from your existing quest/widget input. Retain its returned GUID only on true, and refresh from the outgoing snapshot.
5. Test invalid input, disconnected requests, cancellation, receipt failure/retry and cleanup before using these navigation results in gameplay.

Your game owns the gameplay

The plugin provides local navigation presentation and data. Your game supplies map imagery, quest logic, input integration, networking authority and cloud/profile behavior. Waypoint creation never grants rewards or spawns a physical quest actor.

Read the graphs

Each workflow names the exact saved asset and function, shows its numbered connections and expected result, and links its complete original. Complete graph captures open at their original resolution. Duplicate the shipped assets into your game before customizing them; screenshots are reference views.

Editable examples

Numbered struct UI/native landmark hookups | Included editable examples

Blueprint hookup coverage

Open a workflow for its numbered steps, exact asset/graph names, incoming connections and expected outgoing results. All 12 required workflow groups have authentic graph or configuration captures.

Hookup	Saved example	Instructions
Owner startup	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Map bounds and image	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Landmark settings	BP_RMC_BuyerRegion	Numbered hookup and screenshots
View and surface controls	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Create/rename/remove waypoint	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Settings and progress receipts	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Events, cancellation and cleanup	BP_RMC_BuyerRegion	Numbered hookup and screenshots
Direct public Blueprint nodes	BP_RMC_DirectCaller	Numbered hookup and screenshots
Waypoint/marker output fields	WBP_RMC_BuyerStructUI	Numbered hookup and screenshots
Native marker Blueprint	BP_RMC_BuyerNativeMarker	Numbered hookup and screenshots
Assigned map source Blueprint	BP_RMC_BuyerCalibration	Numbered hookup and screenshots
Capture Blueprint parameters	BP_RMC_BuyerCapture	Numbered hookup and screenshots

2. Map and owner setup

Connect the exact local player

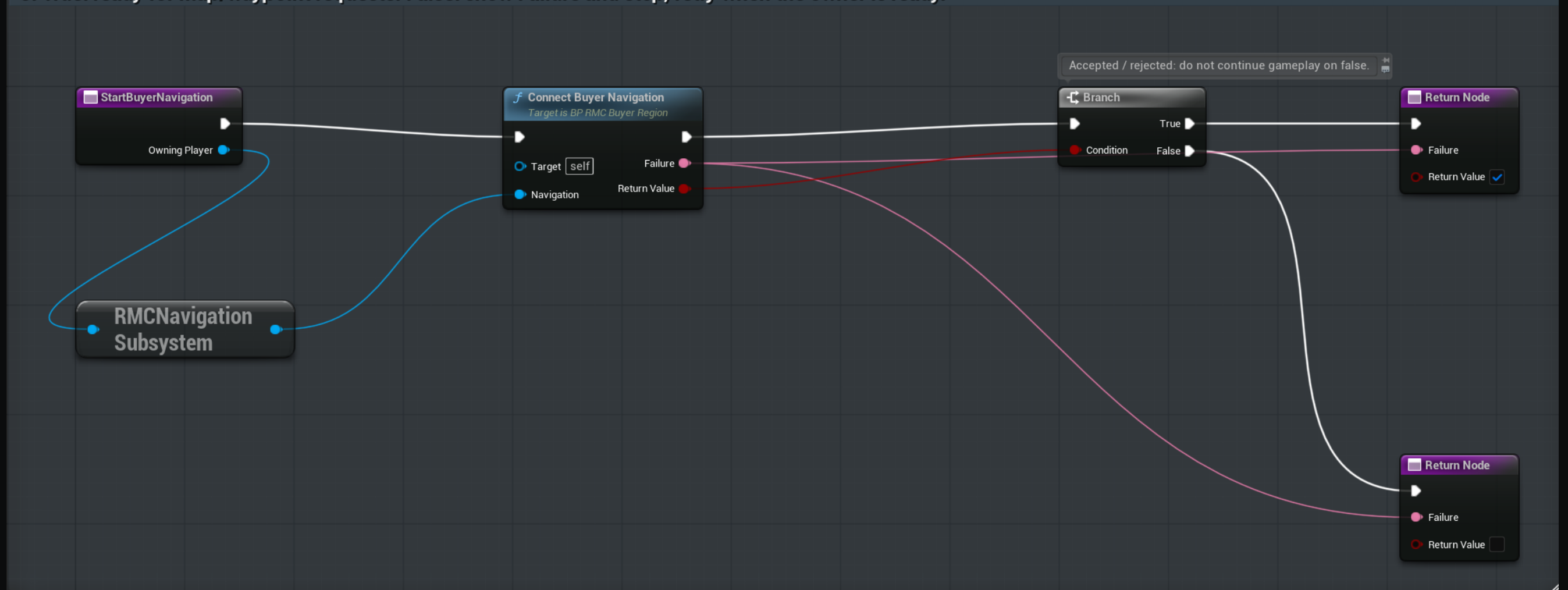
StartBuyerNavigation

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: StartBuyerNavigation

1. Duplicate BP_RMC_BuyerRegion into your game Content folder and place the duplicate in your existing level.
2. From your existing controller or widget, keep a reference to that placed actor. Call Start Buyer Navigation when the owning local PlayerController is ready. Wire that exact controller into Owning Player.
3. Branch on Return Value. On true continue to your map or waypoint input. On false show Failure and retry with the correct controller. Compile and save your duplicate.

Expected result: The owning controller resolves this local player's navigation subsystem; missing ownership returns false with useful failure text.

1. Call Start Buyer Navigation from your existing owning controller BeginPlay or local-player-ready flow.
2. Supply that exact Owning Player; resolve its RMC subsystem and connect this placed actor.
3. True: ready for map/waypoint requests. False: show Failure and stop; retry when the owner is ready.



BP_RMC_BuyerRegion / StartBuyerNavigation - complete Director-supplied graph | Open full-resolution original

Assign your overview and bounds

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: MapSourceSettings

1. Open your duplicated BP_RMC_BuyerRegion. In Components select MapSource, then open Details.
2. Choose your north-up Overview Texture. Set World Center and World Extent as XY half extents in centimeters; clear Use Owner Location as Center when you need a fixed center.
3. Keep Active enabled and choose a priority that makes this the intended source. Compile and save; test center and corner UV projections in Play.

Expected result: The captured shipped defaults have no overview assigned and use owner location as center. This image is configuration evidence; buyer-image rendering remains a separate check.

BP_RMC_BuyerRegion

RPG Maps & Compass

Region Name

Buyer region

Overview Texture

None

None

Minimap Texture

None

None

Scene Mesh Texture

None

None

Use Owner Location as Center

☐

World Center

0.0

0.0

X

0.0

Y

0.0

World Extent

10000.0

10000.0

X

10000.0

Y

10000.0

Priority

100

Active

☒

Relief

Geometry Depth Texture

None

None

Geometry Depth Metrics

R

0.0

G

0.0

B

0.0

A

1.0

Geometry Depth Far

0.0

Relief Texture

None

None

Relief Strength

0.65

Geometry Height Strength

0.55

Geometry Contact Strength

0.22

Contour Strength

0.0

Deprecated

Surface Normal Texture

None

None

Geometry Relief Strength

0.0

Artistic Treatment

Surface Detail Softening

0.15

Relief Contrast

1.35

Relief Shadow Floor

0.55

Cliff Ink Strength

0.25

Tags

Component Tags

0 Array elements

Component Replication

Component Replicates

☐

Activation

Auto Activate

☐

Cooking

Is Editor Only

☐

Variable

Editable when Inherited

☒

Replication

Advanced

Replicate Using Registered Sub Object List

☐

Asset User Data

Advanced

Asset User Data

0 Array elements

Asset User Data Editor Only

0 Array elements

Navigation

Advanced

Can Ever Affect Navigation

☐

BP_RMC_BuyerRegion / MapSourceSettings - retained configuration/reference capture | Open full-resolution original

Name and filter your landmark

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: MarkerSettings

1. In the same duplicated asset select Marker in Components, then open Details.
2. Set Marker Label, category, color and the surfaces where your landmark should appear. Move the placed actor to the intended world location.
3. Compile and save. Exercise your game's visibility/category filter and confirm the landmark disappears after the actor unregisters.

Expected result: The actor supplies landmark data; the navigation subsystem applies category and surface visibility.

BP_RMC_BuyerRegion

RPG Maps & Compass

Marker Label

Buyer landmark

Category

Landmark

Marker Color

R

0.12

G

0.55

B

1.0

A

1.0

Show on Minimap

Show on Compass

Show on World Map

Tags

Component Tags

0 Array elements

Component Replication

Component Replicates

Activation

Auto Activate

Cooking

Is Editor Only

Variable

Editable when Inherited

Replication

Advanced

Replicate Using Registered Sub Object List

Asset User Data

Advanced

Asset User Data

0 Array elements

Asset User Data Editor Only

0 Array elements

Navigation

Advanced

Can Ever Affect Navigation

3. Controls and recovery

Use the maps during play

Click the active game viewport first. Input belongs to its local player; typing a waypoint name does not toggle the map. Use the controls below with the world map open and its settings or waypoint prompt closed unless stated otherwise.

Action	Keyboard / mouse	Controller
Toggle world map	M	Special Left / View
Close the top surface	Escape; dismiss a prompt or settings before the map	Right face button / B / Circle
Pan world map	Drag the map	D-pad or left stick
Zoom	Hover either map and scroll. Minimap: 75-200%; world map: 25-400%.	Left shoulder zooms out; right shoulder zooms in
Move a window	Shift-drag a map window; the standalone launcher can also be Shift-dragged	Use the settings and host UI
Create a waypoint	Ctrl-click inside the world map and complete the naming prompt	Create through your host Blueprint UI
Select a waypoint	Click an existing waypoint	Left face button cycles saved waypoints
Open selected waypoint actions	Use its waypoint menu	Bottom face button
Center on player	Use your host view command or map UI	Top face button
Show world waypoint indicators	Use the minimap eye button or Waypoints in world setting	Use the settings UI

Customize appearance without losing progress

Open the gear and choose World Map or Minimap before changing appearance. Each map retains its own values. Use minimap colors / Use world map colors copies brightness and saturation once; later edits remain independent.

Appearance, marker visibility, minimap zoom and window placement save automatically through the settings path. World-map pan and zoom are session controls. A settings save is separate from a progress save; inspect the matching receipt in your Blueprint UI if you need confirmation.

Setting	Range / behavior
Window scale, width, height	50-150% each
Map image / background / border / settings back glass opacity	0-100% each
Map brightness	25-200%
Map saturation	0-200%
Border thickness	0-12
Text scale	60-180%
Marker scale	50-200%
Map text / direction labels	Independent toggles
Map shape	Square / adjustable, rounded, circle; octagon additionally available for minimap
Fog reveal radius	10-200 m; previously explored cells are retained
Orientation	World map: landscape or north up; changing orientation centers pan
Fit world map	100% world zoom and centered pan; appearance retained
Fit small maps / reset minimap zoom	Minimap settings; reset zoom returns to 100%
Clock	Minimap local system time; 12- or 24-hour format
Category filters	Objectives, Quests, Vendors, Landmarks, Party, Enemies, My waypoints, Resources; applies to maps and compass
Reset appearance	Resets both appearances; preserves window positions, exploration and waypoints
Reset position	Centers the selected map window
Reset standalone launcher position	Restores its responsive factory anchor; does not move a shared suite holder

Hide a category to filter its presentation; this does not delete saved markers. Remove selected waypoint deletes that selected entry. Clear my waypoints removes all user waypoints. Keep cursor visible is a standalone opt-in; a suite coordinator controls its own input.

Practical route: use an authored map image

1. Enable Show Plugin Content, duplicate BP_RMC_BuyerRegion into your game Content, and place your copy in the existing level. Keep the plugin example as a reference.
2. On MapSource, assign a north-up OverviewTexture. For fixed bounds, clear Use Owner Location as Center and enter the XY WorldCenter and positive XY half extents in centimeters. Set the source active and choose its priority deliberately when several regions overlap.

3. Call Start Buyer Navigation with the exact owning local PlayerController. Branch on readiness and operation results; wait for a valid player pose before showing a position or heading.
4. Check two known landmarks on opposite sides of the level. North is world +X, east is world +Y. UV (0.75,0.25) maps to XY (6000,4000) when center is (0,0) and half extents are (12000,8000). The ground trace determines Z when a suitable surface is found.
5. Create and rename a waypoint, retain its accepted GUID, save progress with the progress receipt, travel away and back, then restart the game. Confirm the same ID and name load. Verify settings independently through their settings receipt.

The existing Blueprint guide contains the saved wrapper wiring and result pins. The plugin includes the calibration scene and direct-caller Blueprint under RPGMapsCompass/Examples; the source validation project remains separate.

Practical route: generate a native capture

1. Place a Map Source Actor in your existing level, or create a Blueprint subclass of it. Inspect its MapSource component bounds before generating the overview.
2. Use automatic landscape discovery for loaded landscape terrain. For deliberately selected mesh terrain, tag the intended surfaces RMC.MapGeometry and the intended bounds RMC.MapBounds. Exclude unwanted scene meshes with RMC.MapIgnore. Discovery does not force-load World Partition cells.
3. Start with material base-color capture for opaque or masked surfaces. Call Capture Level Overview after changing topology, bounds or relief settings. Call Refresh Captured Image after surface material colors change without rebuilding terrain relief. Capture Live Minimap refreshes the player-centered image.
4. Check DetectionStatus, SampleCoverage and CapturedSceneMeshCount alongside the actual image. Confirm two landmarks, readable terrain colors and alignment with your markers. A nonempty render target alone is insufficient.
5. Legacy rendered terrain materials applies when material base-color mode is disabled. Tune CaptureExposureBias for this lit route; negative values darken bright capture output. Inspect the image again after the change.

Current native base-color orientation is verified against the northeast/southwest landmarks. Legacy lit exposure and world-map UI image proportions remain unapproved. No final gallery-quality claim is made.

Troubleshoot a concrete symptom

Symptom	Check and next action
M or controller input does nothing	Focus the active game viewport; use the correct owning local controller. Finish any text prompt. Check that your host coordinator has not claimed the surface/input contract.
Ready is false or player appears at origin	Check local controller, spawned pawn and registered root component in the current world. Wait for readiness. The corrected raw example has passed its scoped runtime test. OwnerReady checks reference validity; check player-pose and progress readiness separately.
Image and markers do not align	Check north-up image orientation, XY center and half extents. Distinguish source UV calibration from landscape viewport rotation. Test opposite known landmarks.
Native capture is empty	Check loaded geometry, actor tags, active source and bounds, then Capture Level Overview. Inspect diagnostic fields and the actual image.
Lit capture is washed out	Confirm which capture mode is active. In the legacy lit route reduce CaptureExposureBias and recapture; compare landmark colors. Do not treat this as a packaged DX12 fix.
A waypoint is absent	Check My waypoints category filter, selected source, visibility settings and the loaded progress state. Use its stable GUID rather than its array index.
Settings changed but progress did not save	Read the separate settings and progress receipts. On progress write failure retain pending state and retry after the storage problem is corrected. Do not overwrite protected newer-schema or unreadable progress.
Exploration reveals the whole small level	Reduce the reveal radius; the settings slider spans 10-200 m. Existing explored cells remain revealed.
Map is off-center or oversized	Fit world map restores session zoom/pan. Reset position centers the selected window. Adjust scale, width and height rather than recapturing the map.
Packaged Win64 DX12 errors	Preserve the failing raw log. ONNX GetLastError 206 is a path-handling lead; compute PSO 80070057 is separately unresolved. Editor success does not establish packaged success. Do not repeat an unchanged launch as new proof.

4. View and surface controls

Request zoom and pan

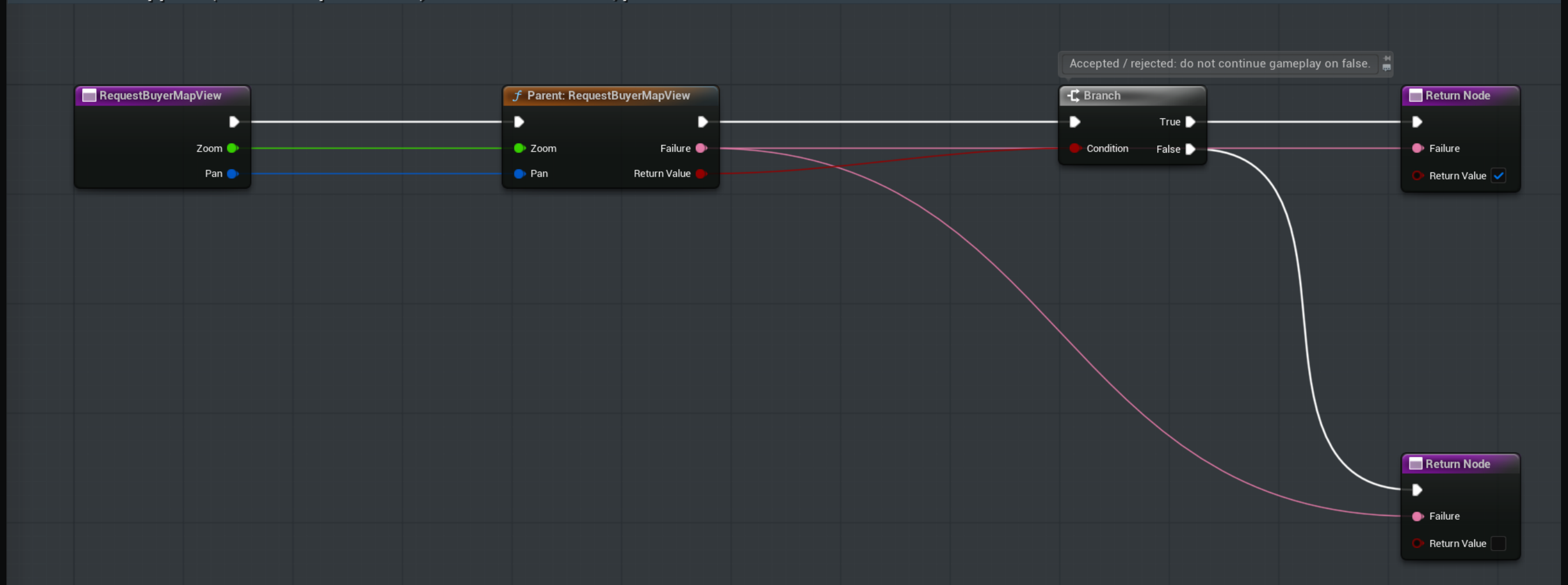
RequestBuyerMapView

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: RequestBuyerMapView

1. From your existing map widget call Request Buyer Map View on the connected region actor.
2. Wire your desired finite Zoom and Pan into the inputs. Pan is a map-view offset; it does not move the pawn.
3. Branch on Return Value. Show Failure on false. Read the effective view after an accepted request because zoom is clamped to 0.25-4.

Expected result: The request affects the owning player's world-map view; missing owner and nonfinite data refuse the request.

1. Call from your own map input with finite Zoom and Pan.
2. Send the view request to the connected owning player's subsystem.
3. True: view applied (Zoom clamped 0.25-4). False: show Failure; previous view is retained.



BP_RMC_BuyerRegion / RequestBuyerMapView - complete Director-supplied graph | [Open full-resolution original](#)

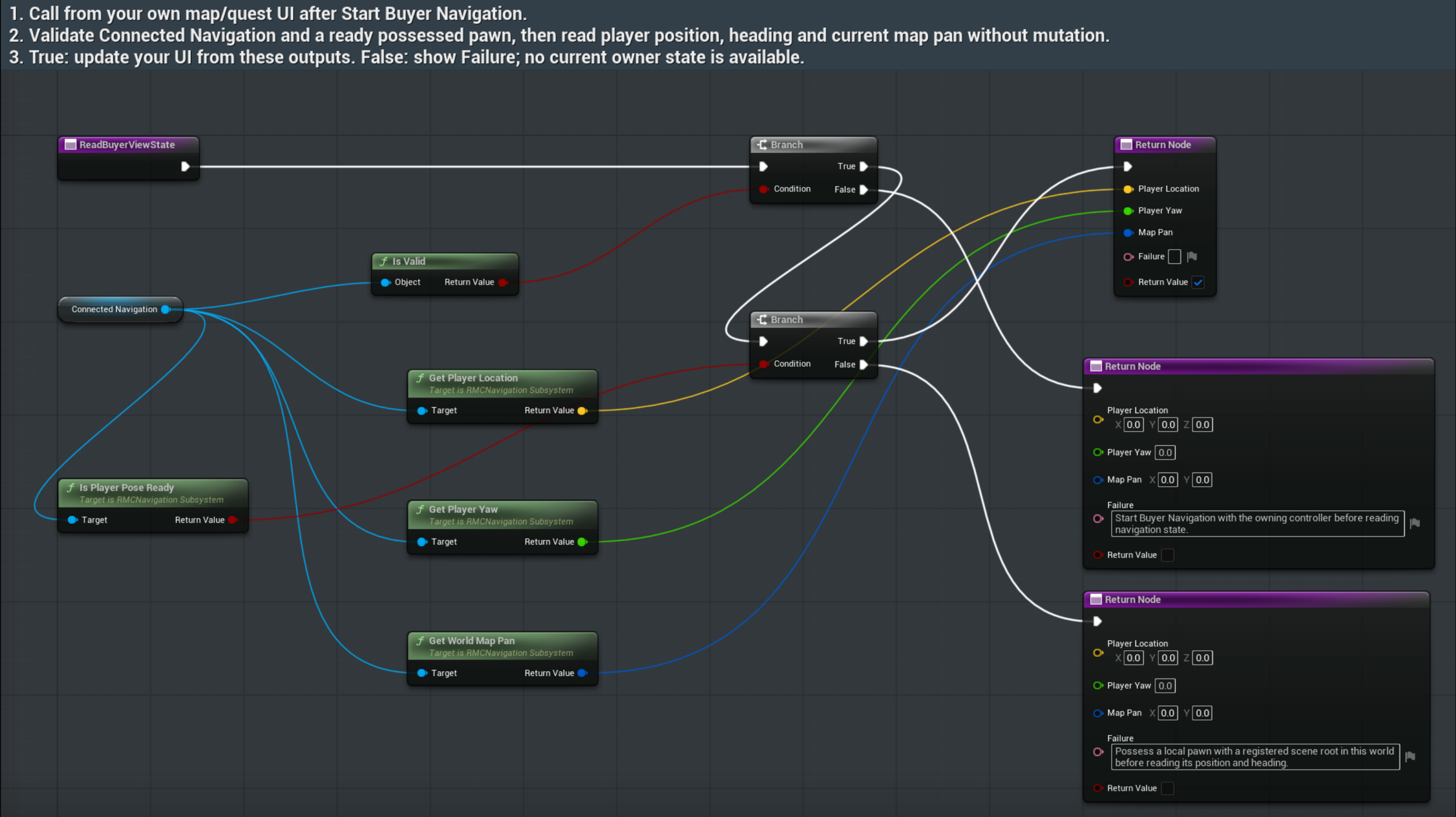
Display real player position and heading

ReadBuyerViewState

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: ReadBuyerViewState

- 1. Call Read Buyer View State from your minimap UI after Start Buyer Navigation.
- 2. Require a possessed local pawn with a registered scene root in the current world. Branch on Return Value before using Player Location, Player Yaw and World Map Pan.
- 3. On false show Failure and wait for a ready pawn or reconnect the owner. Move the pawn, then read again to refresh your UI.

Expected result: The executed saved graph reports the actual moved pawn. It refuses missing pose rather than reporting a fake origin.



BP_RMC_BuyerRegion / ReadBuyerViewState - complete Director-supplied graph | Open full-resolution original

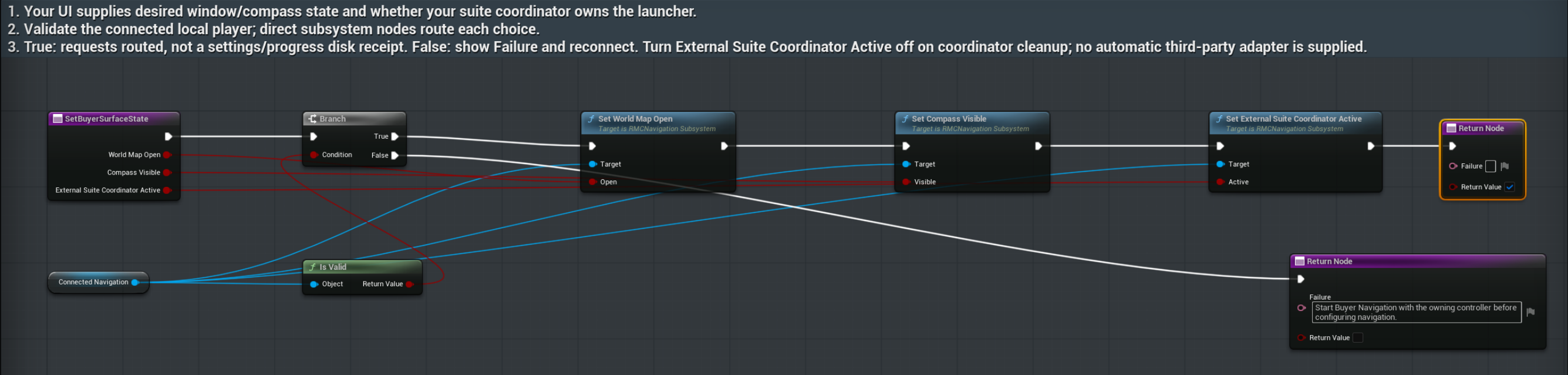
Choose map, compass and launcher ownership

SetBuyerSurfaceState

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: SetBuyerSurfaceState

- 1. Call Set Buyer Surface State from your existing menu or single suite coordinator.
- 2. Wire the desired World Map Open and Compass Visible choices. Set External Coordinator Active only when your actual coordinator owns the standalone launcher.
- 3. Release that ownership flag before coordinator cleanup. Branch on Return Value and show Failure if disconnected.

Expected result: These are runtime surface requests. They do not provide a disk-save receipt or a third-party framework adapter.



BP_RMC_BuyerRegion / SetBuyerSurfaceState - complete Director-supplied graph | Open full-resolution original

5. Waypoints and UI events

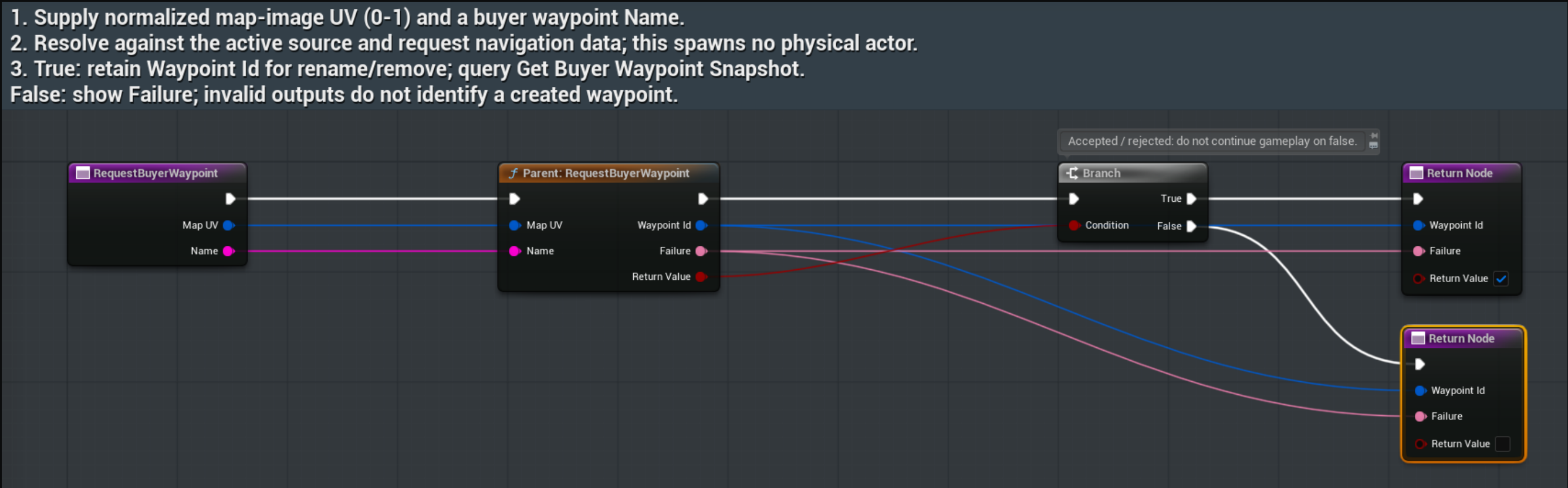
Create navigation data from your input

RequestBuyerWaypoint

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: RequestBuyerWaypoint

- 1. Your quest or map widget supplies normalized Map UV in the range 0-1 and a waypoint Name. Call Request Buyer Waypoint on the connected actor.
- 2. Branch on Return Value. On true retain Waypoint Id in your quest/widget state and read Get Buyer Waypoint Snapshot.
- 3. On false show Failure and discard the invalid output ID. This call creates navigation data; your game owns rewards, quest actors and network authority.

Expected result: Valid input creates a stable GUID and notifies observers. Invalid UV or missing ownership leaves the collection unchanged.



BP_RMC_BuyerRegion / RequestBuyerWaypoint - complete Director-supplied graph | Open full-resolution original

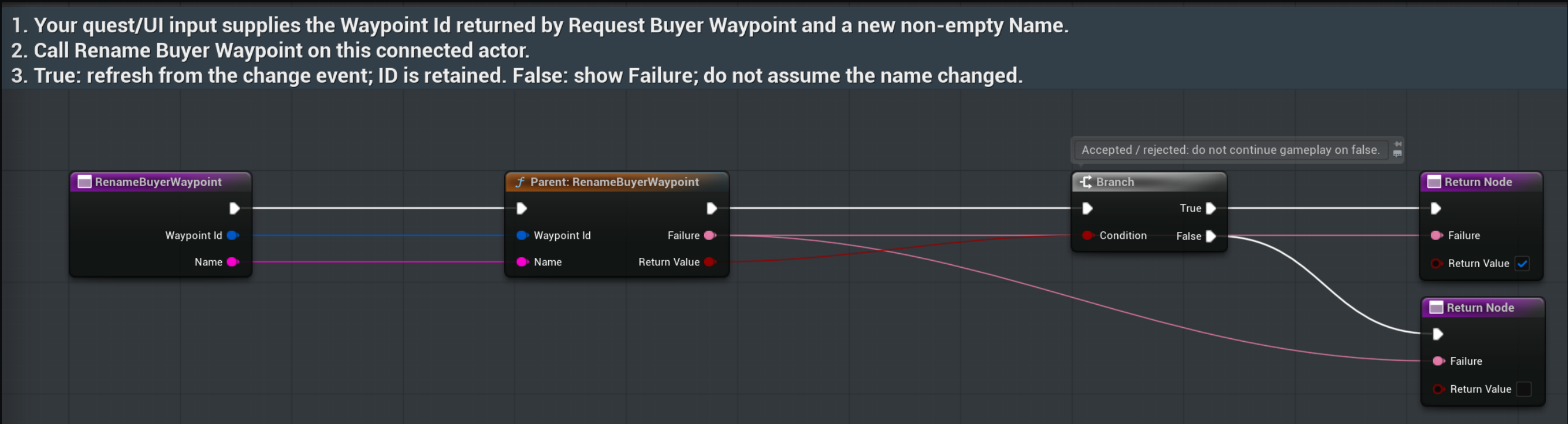
Rename the retained destination

RenameBuyerWaypoint

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: RenameBuyerWaypoint

- 1. Supply the exact Waypoint Id returned by your accepted create request and the desired nonblank Name.
- 2. Call Rename Buyer Waypoint on the connected actor. Branch on the returned result.
- 3. On true refresh your UI from the snapshot. On false show Failure; never treat an unknown ID as a request to rename another waypoint.

Expected result: Rename preserves the stable GUID; blank names, unknown IDs and protected progress refuse mutation.



BP_RMC_BuyerRegion / RenameBuyerWaypoint - complete Director-supplied graph | Open full-resolution original

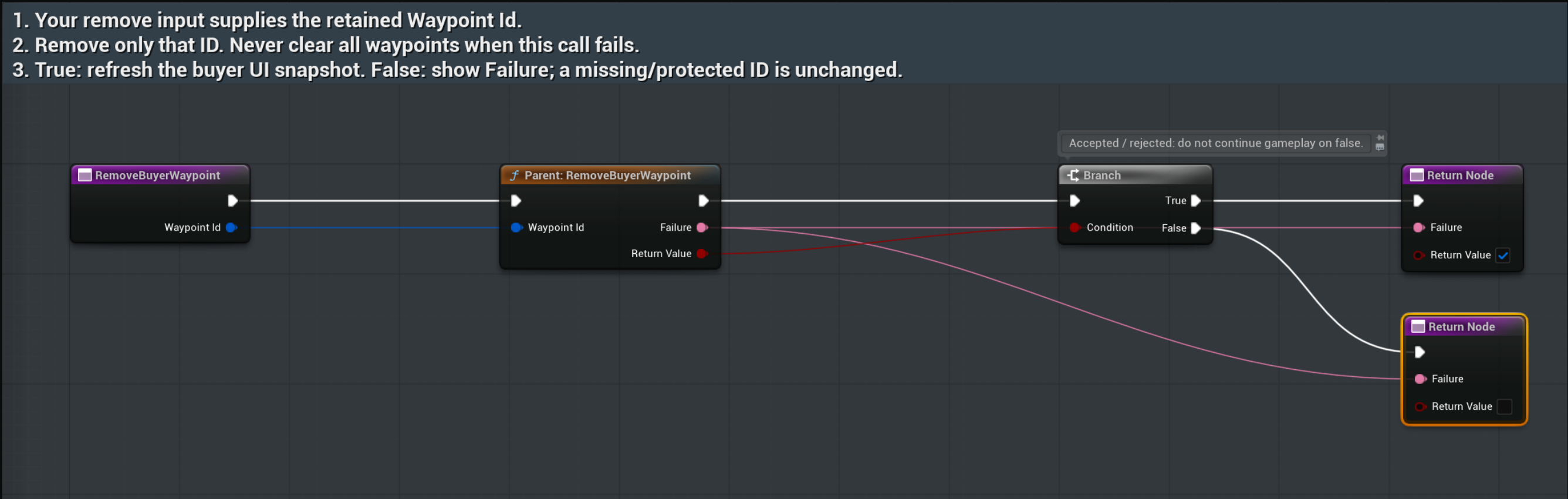
Remove only the chosen ID

RemoveBuyerWaypoint

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: RemoveBuyerWaypoint

- 1. Your remove input supplies the retained Waypoint Id. Call Remove Buyer Waypoint.
- 2. Branch on Return Value. On true refresh from Get Buyer Waypoint Snapshot and clear your retained selection.
- 3. On false show Failure. Never clear the entire waypoint collection as a fallback. Test repeated removal and an unknown GUID.

Expected result: Exactly that waypoint is removed. An unknown or already removed GUID returns false.



BP_RMC_BuyerRegion / RemoveBuyerWaypoint - complete Director-supplied graph | Open full-resolution original

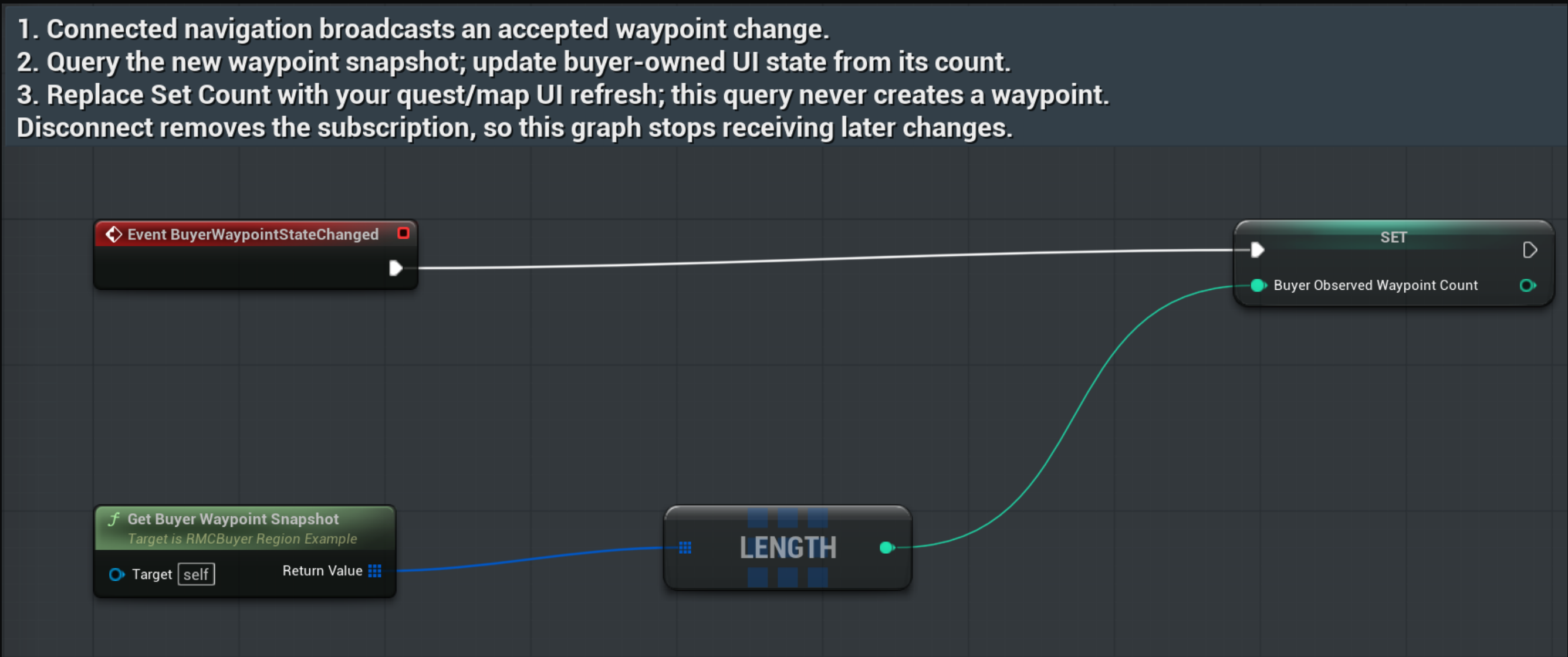
Refresh UI from outgoing state

BuyerWaypointEvents

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: BuyerWaypointEvents

- 1. Open My Blueprint > Graphs > BuyerWaypointEvents in the saved example. Select BuyerWaypointStateChanged.
- 2. After the connected owner broadcasts a waypoint change, Get Buyer Waypoint Snapshot returns the current collection. The example stores its Length in BuyerWaypointCount.
- 3. Extend this event in your duplicate to rebuild your quest markers or widget list. Keep Disconnect Buyer Navigation in cleanup so later changes no longer reach this actor.

Expected result: The saved outgoing event reports both creation and an empty collection after removal.



BP_RMC_BuyerRegion / BuyerWaypointEvents - complete Director-supplied graph | Open full-resolution original

6. Settings and progress

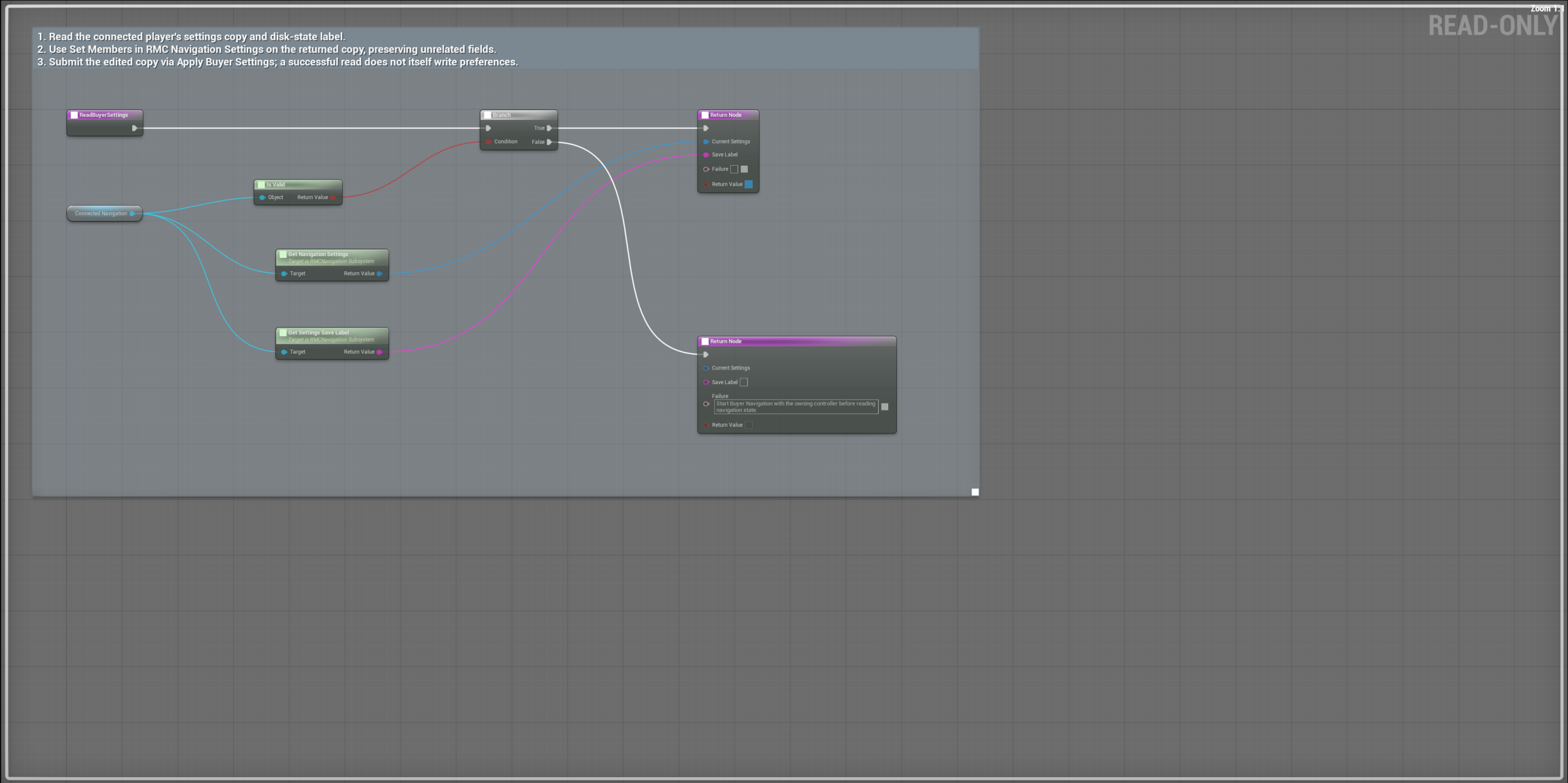
Read an editable settings copy

ReadBuyerSettings

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: ReadBuyerSettings

- 1. Call Read Buyer Settings on the connected actor from your settings UI.
- 2. Branch on Return Value before using Settings and Save Label. Edit a copy, preserving unrelated fields.
- 3. Apply the edited copy through Apply Buyer Settings; read again to display the effective sanitized values and receipt label.

Expected result: The query does not write settings. Missing ownership returns false and useful recovery text.



BP_RMC_BuyerRegion / ReadBuyerSettings - retained configuration/reference capture | Open full-resolution original

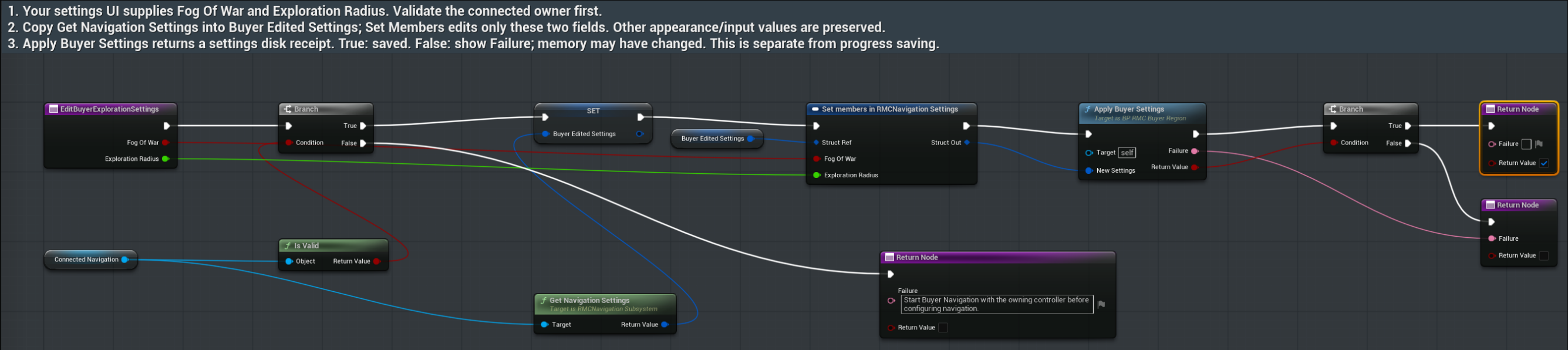
Change fog and reveal radius

EditBuyerExplorationSettings

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: EditBuyerExplorationSettings

- 1. Your settings UI supplies Fog Of War and Exploration Radius. Call Edit Buyer Exploration Settings.
- 2. The saved graph validates the owner, reads current settings, copies them to Buyer Edited Settings and edits only those two fields through Set Members.
- 3. Branch on Return Value after Apply Buyer Settings. On false show Failure: memory may already reflect the choices while the disk receipt is unconfirmed. Retry after the configuration destination is writable.

Expected result: Unrelated settings are preserved. The radius is sanitized to 1000-20000 cm.



BP_RMC_BuyerRegion / EditBuyerExplorationSettings - complete Director-supplied graph | Open full-resolution original

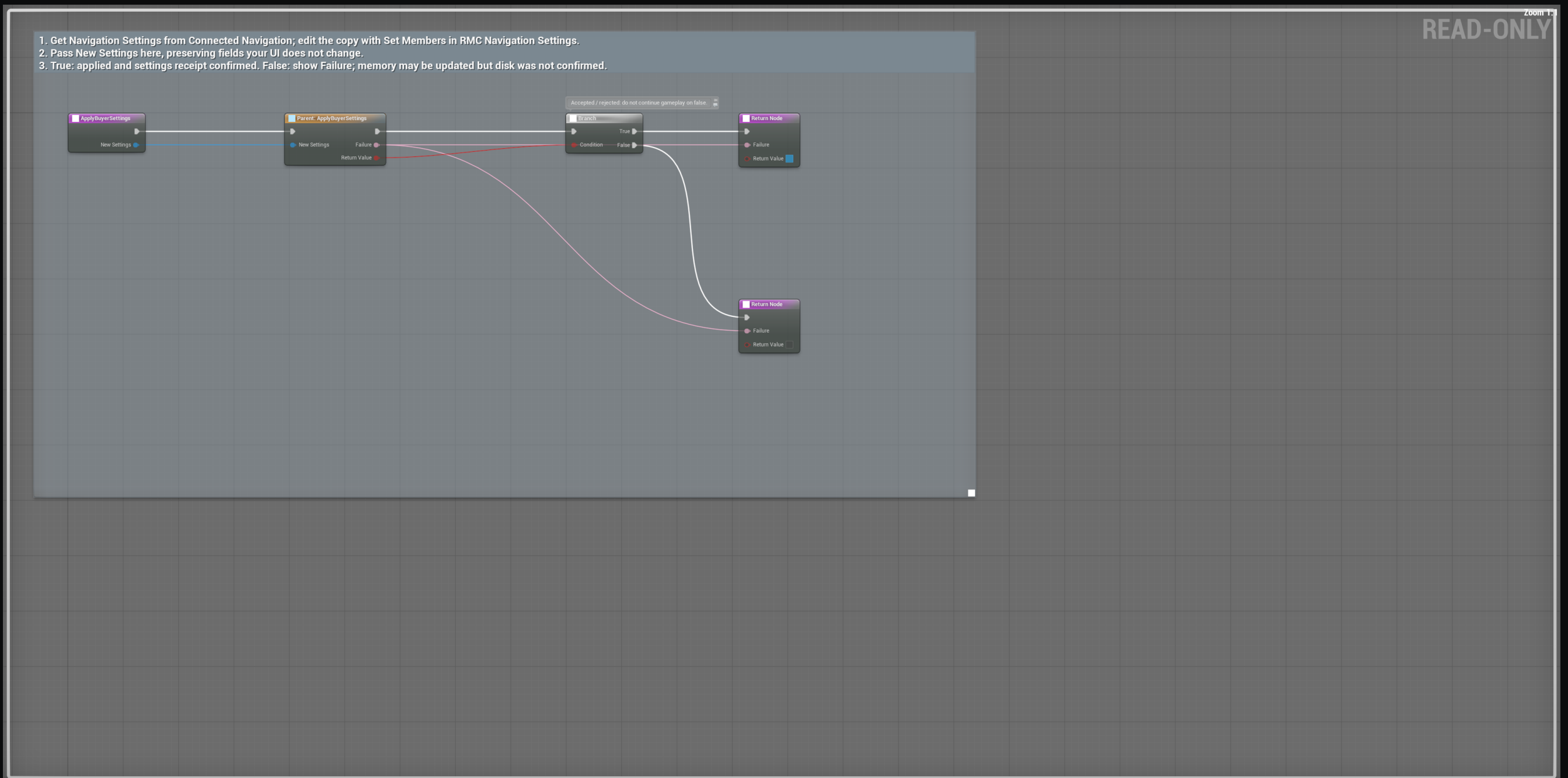
Apply a complete edited copy

ApplyBuyerSettings

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: ApplyBuyerSettings

1. Get the current settings through Read Buyer Settings and edit only your intended members.
2. Pass the complete copy into New Settings on Apply Buyer Settings.
3. Branch on Return Value. True confirms the settings disk receipt; false can leave changes active in memory. Show Failure and read Save Label before presenting a saved confirmation.

Expected result: Actual native and saved Blueprint failure/retry checks distinguish active settings from confirmed disk data.



BP_RMC_BuyerRegion / ApplyBuyerSettings - retained configuration/reference capture | Open full-resolution original

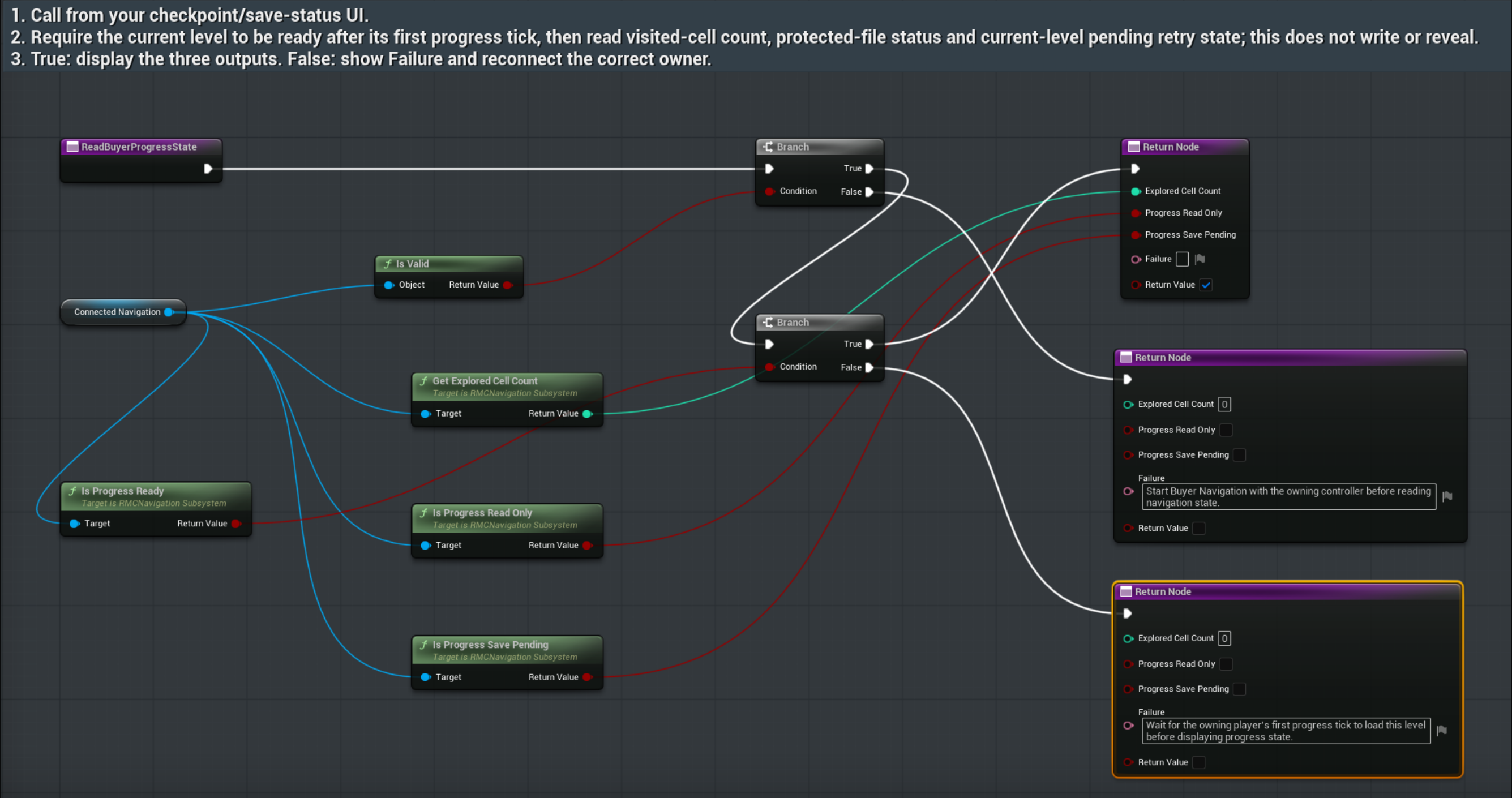
Show ready, protected and pending state

ReadBuyerProgressState

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: ReadBuyerProgressState

- 1. Call Read Buyer Progress State from your checkpoint or save-status UI after connecting the owner.
- 2. Branch on Return Value. Before the owning player's first current-level progress load tick the query refuses with recovery text.
- 3. On true display Explored Cell Count, Progress Read Only and Progress Save Pending separately. The query does not reveal cells or write data.

Expected result: An unsupported newer schema is protected. A physical write failure instead leaves a retry pending; these states are distinct.



BP_RMC_BuyerRegion / ReadBuyerProgressState - complete Director-supplied graph | Open full-resolution original

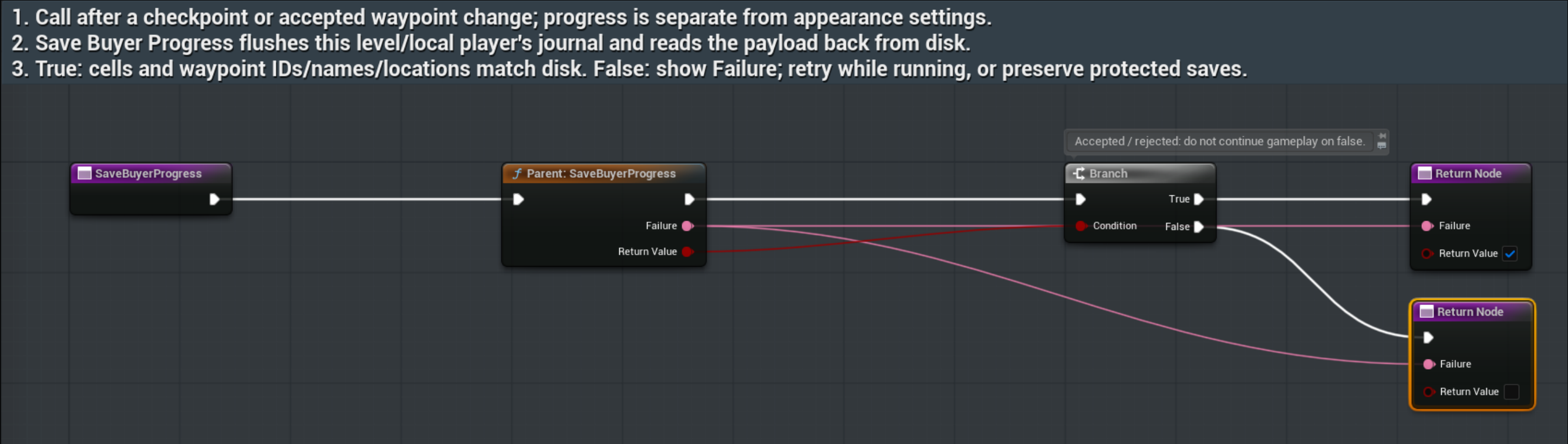
Confirm progress on disk

SaveBuyerProgress

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: SaveBuyerProgress

- 1. Call Save Buyer Progress after an accepted waypoint change or your checkpoint action. This receipt is separate from appearance settings.
- 2. Branch on Return Value. True confirms a fresh readback of supported progress, stable waypoint IDs, names and positions.
- 3. On false show Failure. Retain accepted runtime data and offer a retry when writable; never overwrite a protected newer schema.

Expected result: The physical failure/retry test retains runtime data, fails both native and saved Blueprint receipts, then confirms the same GUID through an independent fresh disk load.



BP_RMC_BuyerRegion / SaveBuyerProgress - complete Director-supplied graph | Open full-resolution original

7. Cancel and cleanup

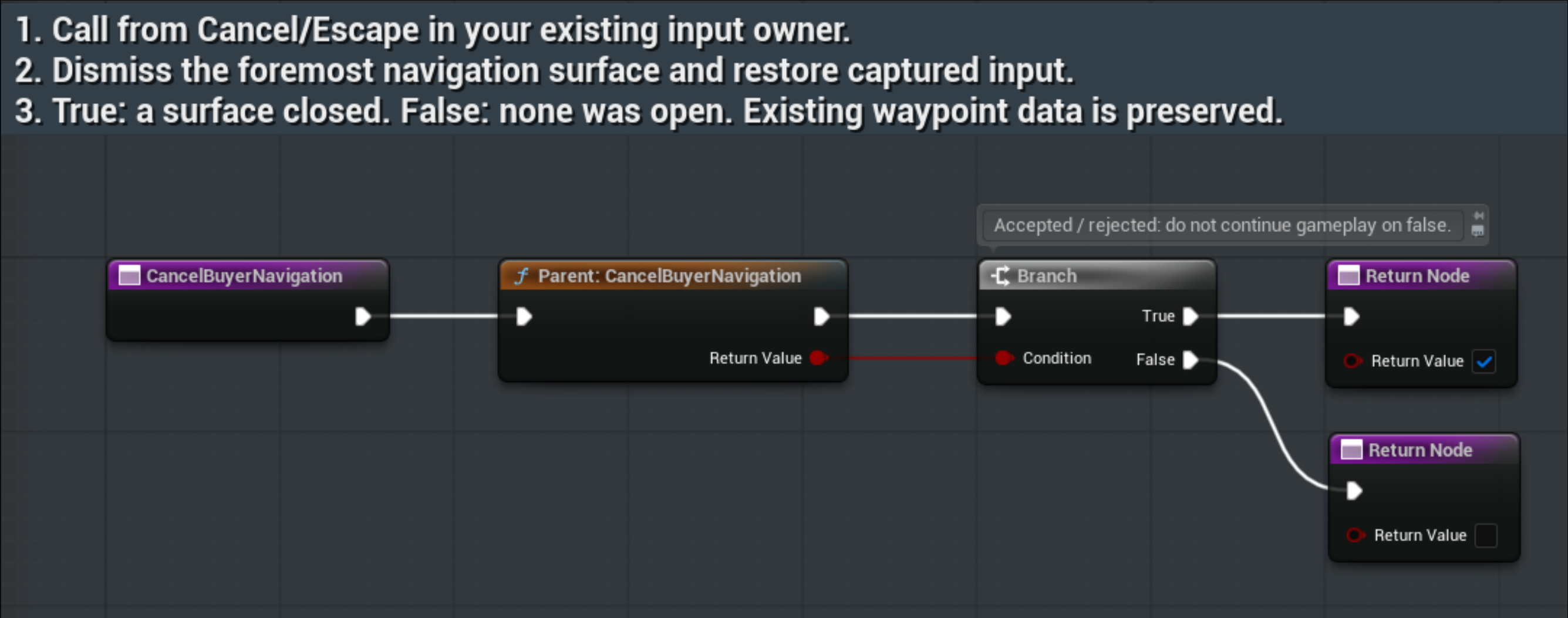
Route your existing cancel input

CancelBuyerNavigation

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: CancelBuyerNavigation

- 1. Call Cancel Buyer Navigation from your existing Escape or back input on the connected actor.
- 2. True means the foremost navigation surface consumed the cancel. False means your game should continue routing its own cancel behavior.
- 3. Do not clear waypoints or gameplay progress when dismissing a surface.

Expected result: Cancellation dismisses navigation UI and preserves gameplay and waypoint data.



BP_RMC_BuyerRegion / CancelBuyerNavigation - complete Director-supplied graph | [Open full-resolution original](#)

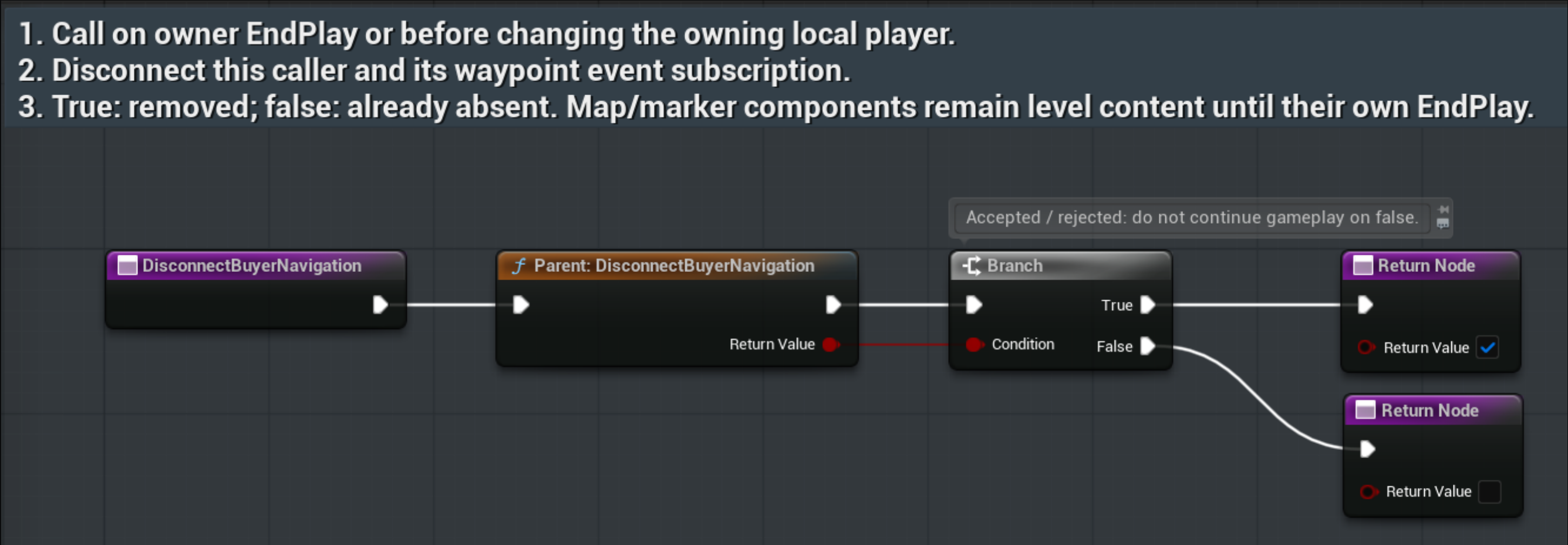
Release your caller

DisconnectBuyerNavigation

Asset: /RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion Tab/function: DisconnectBuyerNavigation

- 1. Call Disconnect Buyer Navigation before changing the owning player or removing your widget/actor reference.
- 2. The example removes only its own observer and clears its connected reference. The actor also calls disconnect during EndPlay.
- 3. Test a request while disconnected, then reconnect the correct owner. Show Failure on refused requests.

Expected result: Cleanup does not clear the plugin-owned waypoint collection. The native fixture verifies actual engine actor destruction and EndPlay reference cleanup.



BP_RMC_BuyerRegion / DisconnectBuyerNavigation - complete Director-supplied graph | [Open full-resolution original](#)

8. Direct Blueprint calls

Use your own actor or widget

- 1. Get the navigation subsystem from the exact owning local player and retain the reference.
- 2. Create the direct public-node call in your existing Blueprint. Supply your own component, location, name or stable GUID inputs.
- 3. Branch on operation results before refreshing UI or using outgoing values.
- 4. Bind your own event to On Waypoints Changed once, and remove that exact event during cleanup.

BP_RMC_DirectCaller is an included editable Actor Blueprint. Duplicate it into your game to adapt its direct public-node wiring, or use the BP_RMC_BuyerRegion wrapper.

OwnerReady in this raw example means the subsystem reference is valid. It does not replace Is Player Pose Ready or Is Progress Ready. Runtime verification covers these saved calls and a disconnected-owner path; it does not prove every possible failure combination.

RegisterMarker

Pass your owned Marker component; registration makes it available to the owning player's navigation.



BP_RMC_DirectCaller / DirectRegisterMarker - complete Director-supplied graph | [Open full-resolution original](#)

UnregisterMarker

Pass the same owned component during cleanup; do not unregister another actor's marker.

Full graph reference: [HTML guide](#), Direct Blueprint calls chapter.

GetVisibleMarkers

Read the outgoing array and break each RMC Map Marker View for label, location, color and per-surface visibility.

Full graph reference: [HTML guide](#), Direct Blueprint calls chapter.

RegisterMapSource

Pass your owned source component after configuring its bounds and texture.

Full graph reference: [HTML guide](#), Direct Blueprint calls chapter.

UnregisterMapSource

Remove your own source during cleanup; query the newly active source afterward.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

GetActiveMapSource

Read the actual source reference; branch on validity before reading its fields.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

ResolveMapLocation

Pass normalized UV and branch on Return Value before using Out Location.

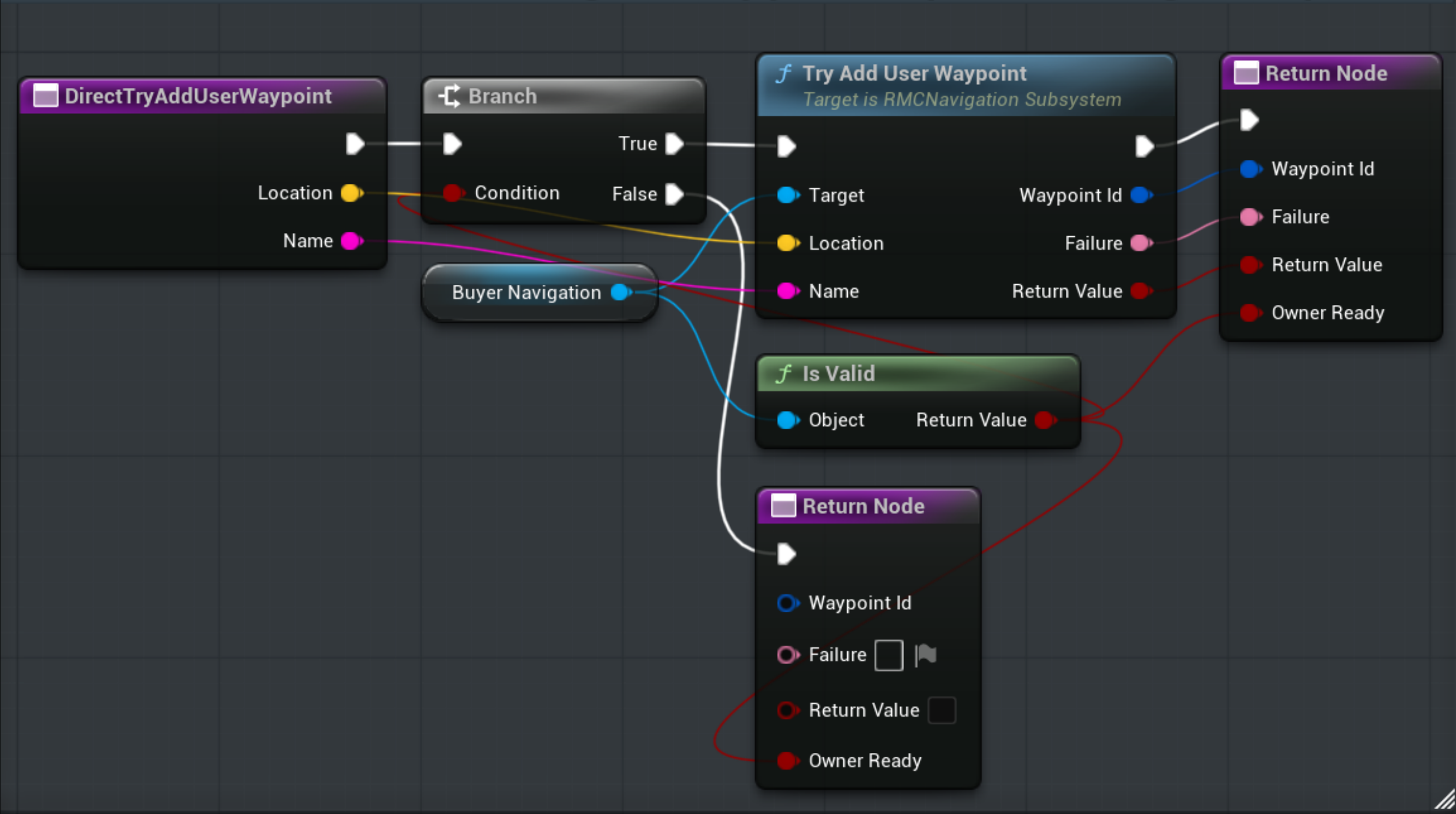
Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

TryAddUserWaypoint

Pass world Location and Name. On true retain Waypoint Id; on false display Failure and retain your retry input.

Direct Try Add User Waypoint

- 1. Assign Buyer Navigation from the exact owning local player's subsystem before this call.
- 2. Your existing Blueprint provides the TryAddUserWaypoint inputs; this saved graph calls the public node directly.
- 3. Retain and branch on returned results, then refresh your UI from current state. Disconnect the observer and unregister only your components during cleanup.



BP_RMC_DirectCaller / DirectTryAddUserWaypoint - complete Director-supplied graph | [Open full-resolution original](#)

AddUserWaypoint

Legacy direct add route returns accepted status; prefer Try Add when your UI needs the returned GUID and Failure.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

RenameUserWaypoint

Pass the retained stable Waypoint Id and new Name, then refresh from the outgoing array.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

RemoveUserWaypoint

Pass the retained stable Waypoint Id; on accepted removal refresh your list.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

ClearUserWaypoints

Clear only the owning local player's user-waypoint collection; refresh the UI afterward.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

GetUserWaypoints

Read the outgoing array; preserve Id, Name and Location together. Array indices are not stable identifiers.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

ProjectWaypoint

Pass a world Location and branch on its projection result before placing your screen indicator.

Full graph reference: [HTML guide](#), [Direct Blueprint calls chapter](#).

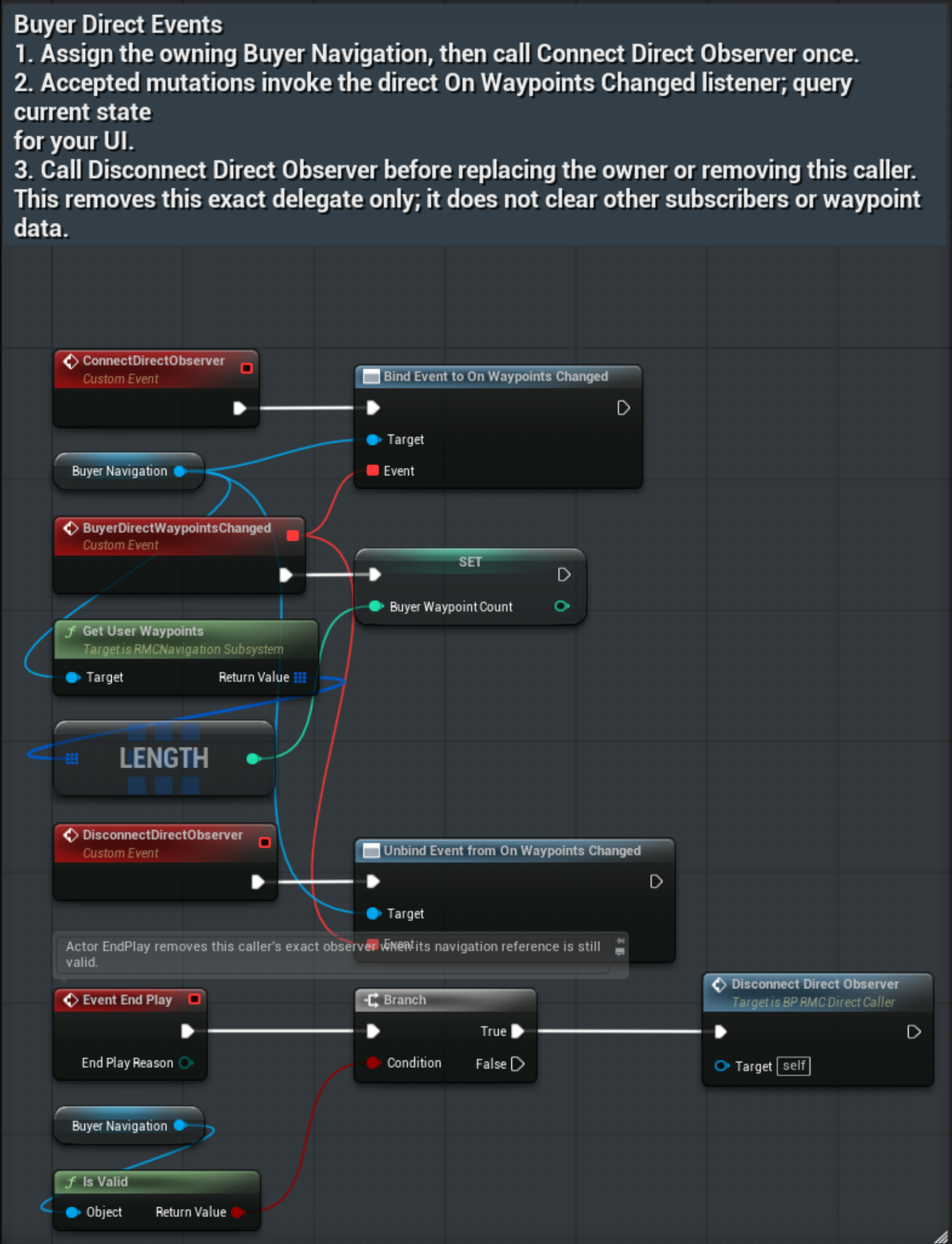
Bind, refresh and clean up

ConnectDirectObserver

BuyerDirectWaypointsChanged

DisconnectDirectObserver

Connect Direct Observer binds the exact Buyer Direct Waypoints Changed event. That event queries the current array and updates Buyer Waypoint Count. Disconnect Direct Observer removes the same event. The saved EndPlay graph disconnects when the navigation reference is valid. A natural actor-destruction test verified that it removes this listener while preserving another actor's listener.



BP_RMC_DirectCaller / BuyerDirectEvents - complete Director-supplied graph | [Open full-resolution original](#)

Use the outgoing structs in your UI

For Get User Waypoints, loop the returned array and use Break RMC User Waypoint: Id is the stable GUID for rename/remove, Name is the displayed name, and Location is the world destination. For Get Visible Markers, loop the returned array and use Break RMC Map Marker View: Id, Label, Category, World Location, Color, Show On Minimap, Show On Compass and Show On World Map. The saved direct graph returns the arrays. The separate WBP_RMC_BuyerStructUI now provides verified editable Break Struct functions; bind its outgoing fields to your own UI.

9. UI fields and landmarks

Get the editable examples

Included editable examples . Enable RPGMapsCompass and Show Plugin Content in your existing UE 5.8.3 project. Find the included plugin examples at the package paths below, then duplicate them into your own game Content to customize them. No additional project or other BIS product is required.

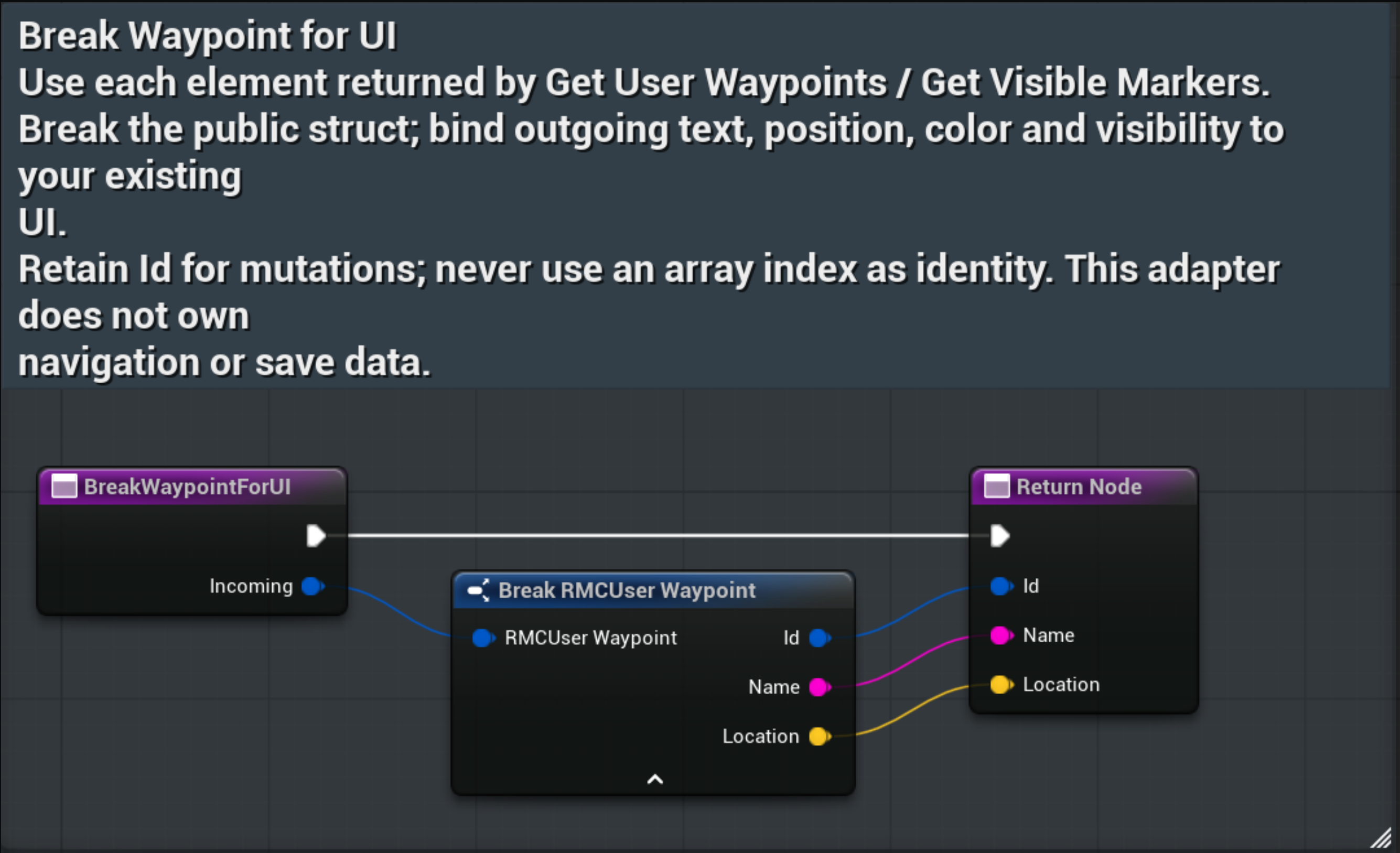
The seven calibration/raw assets are retained. The new widget and native marker are separate examples; they are not already placed in the calibration level. The widget contains binding instructions and editable data functions, not a finished HUD layout.

19. Hook waypoint records to your UI

BreakWaypointForUI

1. Complete the owning-player hookup first. Read the current Get User Waypoints array from that player's navigation subsystem; refresh it after accepted mutations or the scoped waypoint-changed event.
2. In your existing UI, create or retain a WBP_RMC_BuyerStructUI instance with your owning player. For each returned array element, pass the entire record to Break Waypoint For UI.
3. Connect Name to your text widget (convert String to Text where required), Location to your destination/indicator logic, and Id to the row's retained selection key.
4. On rename/remove, supply the retained Id to the relevant navigation call. Branch on that call's accepted/failure result before updating the row; keep editable retry input on refusal.

Expected result: the three fields preserve the same outgoing record. Empty/default records remain empty/default; the adapter creates no destination or save operation. Disable mutation actions until a valid accepted GUID is selected. Array indices are not identity.



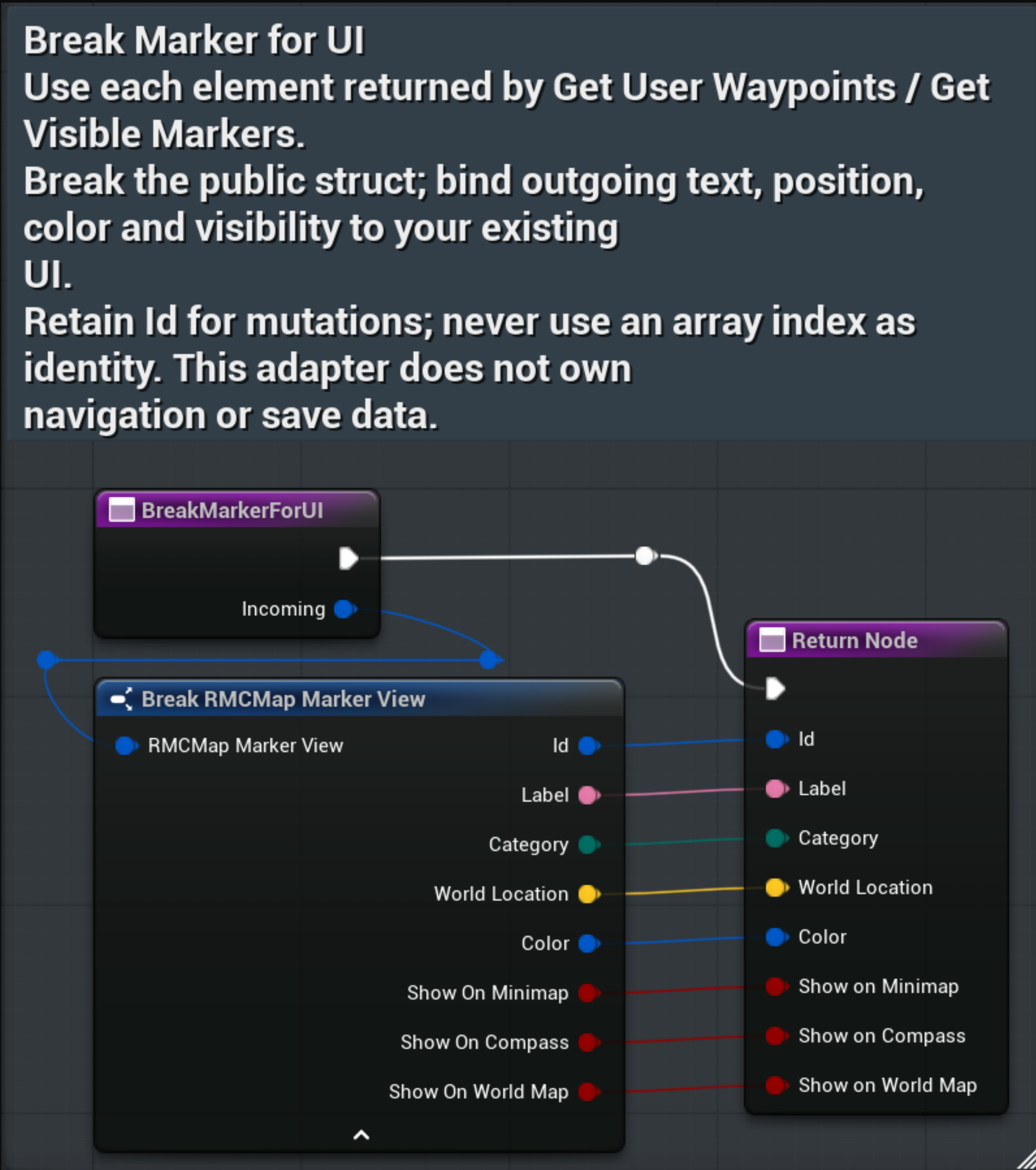
WBP_RMC_BuyerStructUI / BreakWaypointForUI - complete Director-supplied graph | Open full-resolution original

20. Hook marker records to your UI

BreakMarkerForUI

1. Query Get Visible Markers from the exact owning player. Pass each returned record to Break Marker For UI.
2. Bind Label and Color to your existing marker widget, World Location to world/map/projection logic, and Category to your presentation/filter logic. Keep Id as the marker's stable row key.
3. Use Show On Minimap, Show On Compass and Show On World Map for the corresponding display surfaces. Respect the player's category filters; surface flags and category visibility are distinct.
4. Refresh your rows from current state. The adapter reads a copy; it does not alter the registered component, ownership, waypoint collection or disk progress.

Expected result: all eight outgoing fields match the incoming view. For an unavailable/invalid marker reference, retain your UI's unavailable state rather than treating a default view as an accepted gameplay destination.



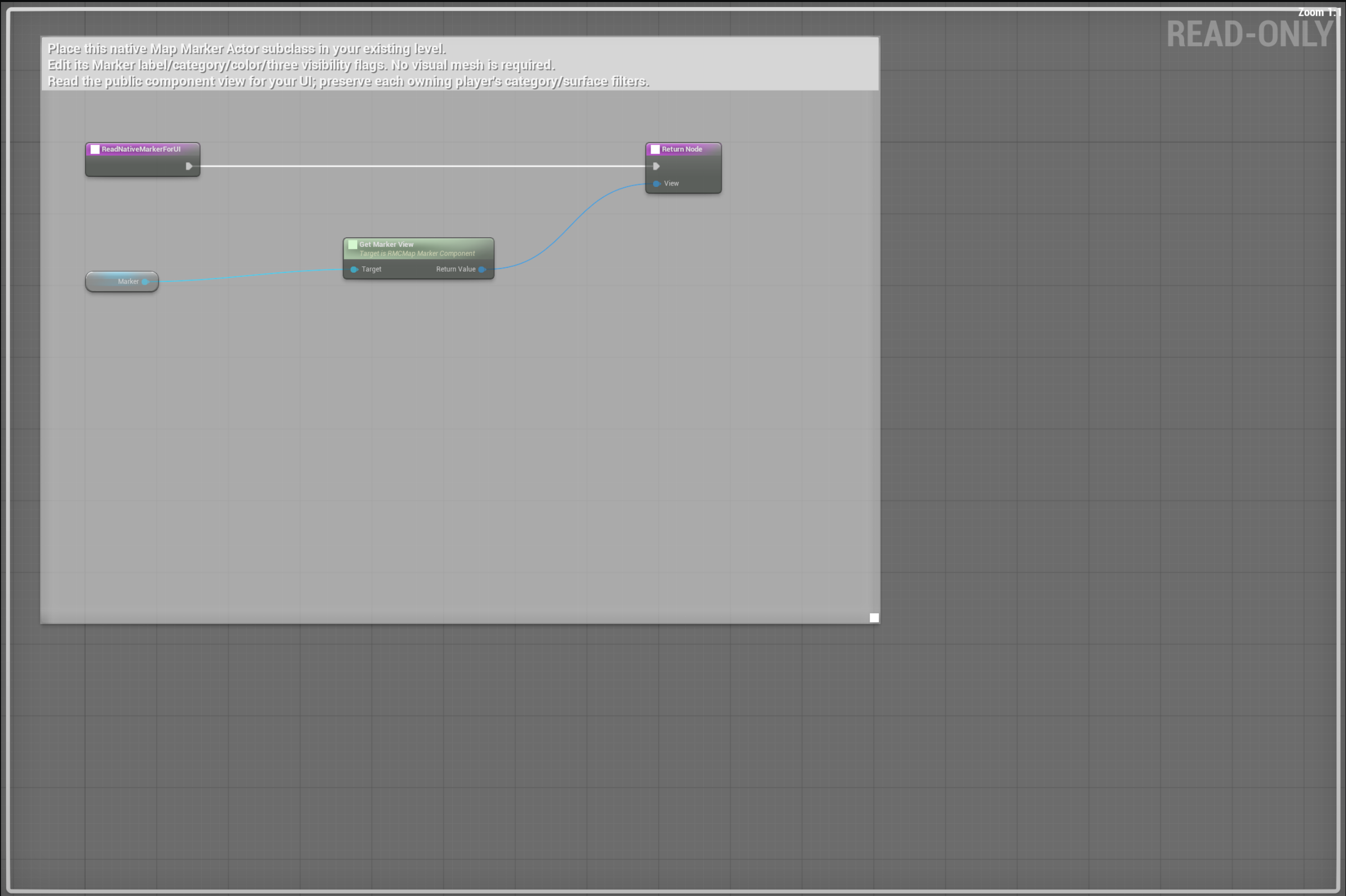
WBP_RMC_BuyerStructUI / BreakMarkerForUI - complete Director-supplied graph | Open full-resolution original

21. Place and customize a native landmark

1. Place BP_RMC_BuyerNativeMarker in your existing gameplay level and choose its transform. It inherits public RMC Map Marker Actor and needs no visual mesh.
2. Select its Marker component. Set Marker Label, Category, Marker Color and the three display flags. The saved example uses label Buyer native landmark, Landmark, color(.8,.2,.1,1), minimap true/compass false/world map true.
3. Retain the valid placed/spawned actor reference. During normal gameplay after world actors initialize, call Read Native Marker For UI; pass its View to Break Marker For UI or your equivalent field bindings.
4. Query registered markers through the owning player's navigation. For explicit registration, register/unregister only your own component during its lifecycle. Preserve other registrations and filters.

Expected result: View contains the component's stable ID, label/category/color/flags and the actor's current world location. Call after normal world actor initialization; this read does not make an unready world ready. Editor/tooling calls before actor initialization can leave default outputs and do not establish successful gameplay execution.

The clean test exercised exact saved read bytecode and explicit public registration/unregistration. Automatic local-player BeginPlay/EndPlay marker registration remains a separate coverage gap; no designed HUD/gallery claim is made.



BP_RMC_BuyerNativeMarker / ReadNativeMarkerForUI - retained configuration/reference capture | Open full-resolution original

RPG Maps & Compass

Marker Label	Buyer native landmark	
Category	Landmark	

Marker Color

R	0.8
G	0.2
B	0.1
A	1.0

Show on Minimap	
Show on Compass	
Show on World Map	

Tags

Component Tags	0 Array elements		
----------------	------------------	--	--

Component Replication

Component Replicates	
----------------------	--

Activation

Auto Activate	
---------------	--

Cooking

Is Editor Only	
----------------	--

Variable

Editable when Inherited	
-------------------------	--

Replication

Advanced

Replicate Using Registered Sub Object List	
--	--

Asset User Data

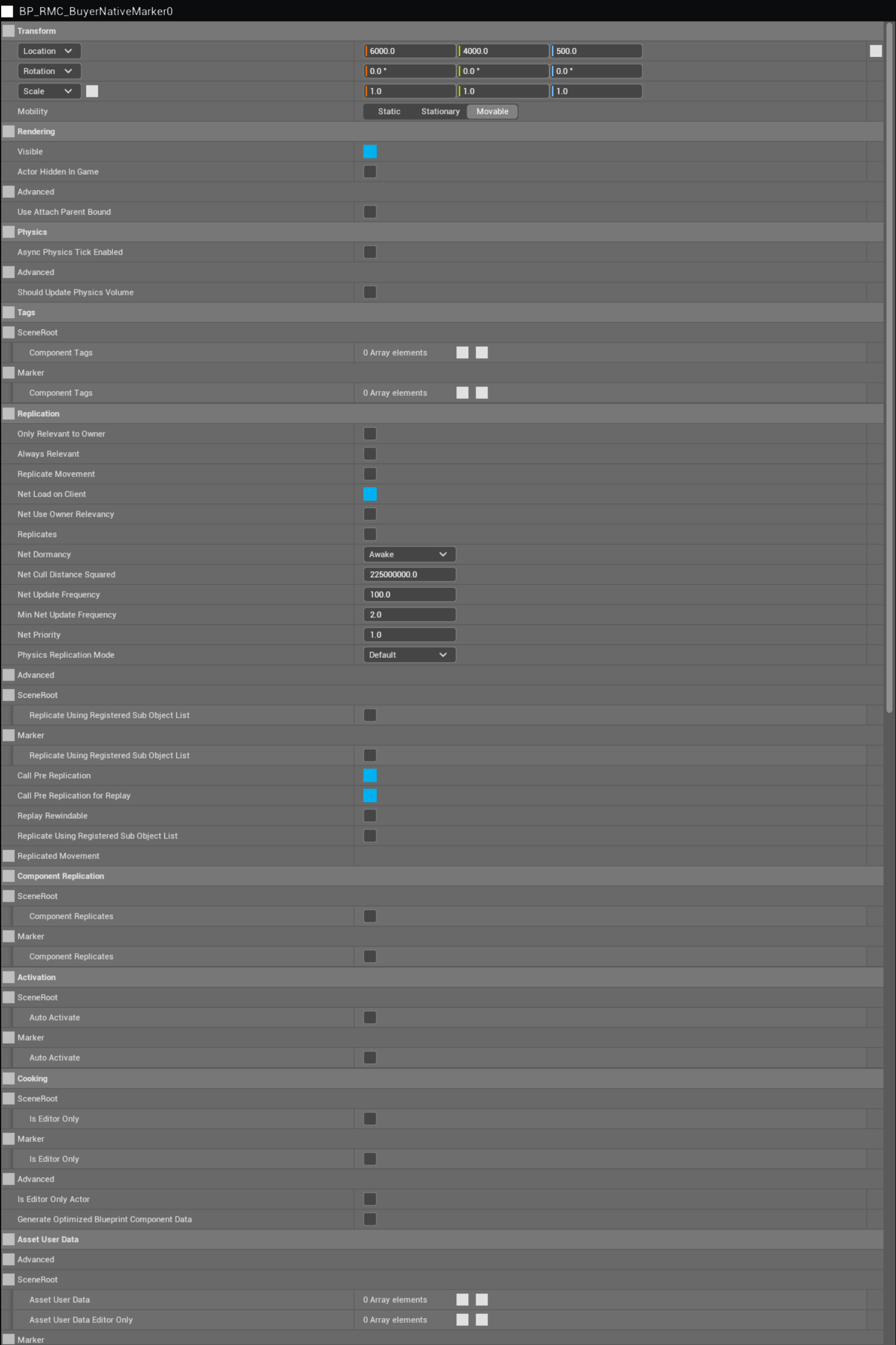
Advanced

Asset User Data	0 Array elements		
Asset User Data Editor Only	0 Array elements		

Navigation

Advanced

Can Ever Affect Navigation	
----------------------------	--



BP_RMC_BuyerNativeMarker / Actor-Details - retained configuration/reference capture | Open full-resolution original

Readiness, failure and cleanup

Input / condition	Buyer action and outgoing result
Subsystem reference unavailable	Branch on reference readiness before navigation calls. Raw OwnerReady means reference validity; query player pose and progress readiness separately.
Default struct or no selected accepted GUID	Adapter maps default fields; keep unavailable/empty UI state and disable mutations. It creates no success/failure receipt.
World actors not initialized or actor reference invalid	Wait for normal gameplay readiness and a valid actor. Treat default outputs as unavailable; do not infer acceptance.
Mutation refused or disk write failed	Branch on the originating operation's result/receipt, retain useful retry input, and keep settings and progress disk receipts separate.
UI listener removed or component owned by your actor ends	Disconnect your exact observer and unregister only your own explicit registrations; do not clear other owners.

10. Native capture settings

Native configuration reference: RMCMapSourceActor.h

Names and defaults below are extracted from the current source declaration. Deprecated fields preserve older serialized assets; use their documented replacements for new setup. This table documents configuration, not successful runtime coverage of every field.

Declaration / default	Category	Access	Editor limits	Purpose
TObjectPtr<UMaterialInterface> MapToneMaterial	RPG Maps & Compass Cartography	Read-only / diagnostic		Hard referenced so the independently installed plugin cooks its image-only tone shader.
TObjectPtr<UMaterialInterface> ShapeSurfaceMaterial	RPG Maps & Compass Cartography	Read-only / diagnostic		UI shader for vertex-tinted polygon backgrounds and texture fallbacks.
bool bEnableSceneMeshCapture = true	RPG Maps & Compass Scene Geometry	Editable		Inspect the field alongside the practical route above.
int32 SceneMeshResolution = 2048	RPG Maps & Compass Scene Geometry	Editable	Min 256, Max 2048	Inspect the field alongside the practical route above.
float SceneMeshRefreshInterval = .5f	RPG Maps & Compass Scene Geometry	Editable	Min 0.1, Max 10	Inspect the field alongside the practical route above.
FName SceneMeshIgnoreTag = TEXT("RMC.MapIgnore")	RPG Maps & Compass Scene Geometry	Editable		Inspect the field alongside the practical route above.
int32 CapturedSceneMeshCount = 0	RPG Maps & Compass Diagnostics	Read-only / diagnostic		Inspect the field alongside the practical route above.
TObjectPtr<USceneComponent> SceneRoot	RPG Maps & Compass	Read-only / diagnostic		Inspect the field alongside the practical route above.
TObjectPtr<URMCMapSourceComponent> MapSource	RPG Maps & Compass	Read-only / diagnostic		Inspect the field alongside the practical route above.
TObjectPtr<USceneCaptureComponent2D> OverviewCapture	RPG Maps & Compass	Read-only / diagnostic		Inspect the field alongside the practical route above.
TObjectPtr<USceneCaptureComponent2D> MinimapCapture	RPG Maps & Compass	Read-only / diagnostic		Inspect the field alongside the practical route above.
TObjectPtr<USceneCaptureComponent2D> GeometryDepthCapture	RPG Maps & Compass	Read-only / diagnostic		Depth-only geometry pass aligned to the overview capture.
TObjectPtr<USceneCaptureComponent2D> SurfaceNormalCapture	RPG Maps & Compass Deprecated	Deprecated compatibility field		Deprecated normal component reference retained for serialized compatibility. Always null.
bool bAutoDiscoverLandscapeBounds = true	RPG Maps & Compass Capture	Editable		Inspect the field alongside the practical route above.
bool bRefreshWhenTerrainChanges = true	RPG Maps & Compass Capture	Editable		Recheck loaded terrain bounds at a bounded cadence; rebuild only after topology changes settle. Does not force-load World Partition.
FString DetectionStatus	RPG Maps & Compass Diagnostics	Read-only / diagnostic		Inspect the field alongside the practical route above.
float SampleCoverage = 0.f	RPG Maps & Compass Diagnostics	Read-only / diagnostic		Inspect the field alongside the practical route above.
bool bCaptureOverviewOnBeginPlay = true	RPG Maps & Compass Capture	Editable		Inspect the field alongside the practical route above.
bool bEnableLiveMinimapCapture = true	RPG Maps & Compass Capture	Editable		Inspect the field alongside the practical route above.
bool bUseMaterialBaseColor = true	RPG Maps & Compass Capture	Editable		Capture actual opaque/masked material base colors, independent of exposure. Disable for legacy artistic relief.
bool bCaptureRenderedTerrainMaterials = false	RPG Maps & Compass Capture	Editable		Legacy lit capture, used only when material-base-color mode is disabled.
bool bGenerateTopographyFromTerrain = true	RPG Maps & Compass Topography	Editable		Build portable north-up relief from real elevation and authored landscape paint layers.
bool bUseLandscapePaintLayers = true	RPG Maps & Compass Topography	Editable		Use each landscape's authored paint weights so sand, soil, grass, farmland, and rock boundaries remain recognizable.
float PastelStrength = 1.f	RPG Maps & Compass Cartography	Editable	Min 0, Max 1	Hand-pigmented relief over measured heights and paint weights. Set to zero for plain relief. Rebuild overview after editing.

Declaration / default	Category	Access	Editor limits	Purpose
float PaperGrainStrength = .35f	RPG Maps & Compass Cartography	Editable	Min 0, Max 1	Inspect the field alongside the practical route above.
FLinearColor CartographyInk = FLinearColor(.065f,.035f,.085f)	RPG Maps & Compass Cartography	Editable		Inspect the field alongside the practical route above.
FLinearColor CartographyPaper = FLinearColor(.74f,.57f,.34f)	RPG Maps & Compass Cartography	Editable		Inspect the field alongside the practical route above.
FLinearColor CartographyOcean = FLinearColor(.035f,.12f,.19f)	RPG Maps & Compass Cartography	Editable		Inspect the field alongside the practical route above.
TMap<FName, FLinearColor> LandscapeLayerColorOverrides	RPG Maps & Compass Topography	Editable		Optional exact layer-name color overrides. Unlisted layers receive a natural color inferred from their name.
int32 TopographyResolution = 2048	RPG Maps & Compass Topography	Editable	Min 128, Max 2048	Inspect the field alongside the practical route above.
FName SeaLevelActorTag = TEXT("RMC.MapSeaLevel")	RPG Maps & Compass Topography	Editable		Inspect the field alongside the practical route above.
FName WaterRouteActorTag = TEXT("RMC.MapWater")	RPG Maps & Compass Topography	Editable		Inspect the field alongside the practical route above.
float WaterRouteWidth = 1800.f	RPG Maps & Compass Topography	Editable	Min 100.0, Max 10000.0	Inspect the field alongside the practical route above.
int32 OverviewResolution = 4096	RPG Maps & Compass Capture	Editable	Min 256, Max 4096	Inspect the field alongside the practical route above.
int32 GeometryDepthResolution = 1024	RPG Maps & Compass Capture	Editable	Min 256, Max 1024	Depth-only geometry capture. A 1024 long axis bounds memory while retaining canopy and rock form.
bool bCaptureGeometryDepth = true	RPG Maps & Compass Capture	Editable		Inspect the field alongside the practical route above.
int32 SurfaceNormalResolution = 2048	RPG Maps & Compass Deprecated	Deprecated compatibility field		Inspect the field alongside the practical route above.
bool bCaptureSurfaceNormals = false	RPG Maps & Compass Deprecated	Deprecated compatibility field		Inspect the field alongside the practical route above.
int32 MinimapResolution = 512	RPG Maps & Compass Capture	Editable	Min 128, Max 2048	Inspect the field alongside the practical route above.
float InitialCaptureDelay = 1.5f	RPG Maps & Compass Capture	Editable	Min 0.0, Max 5.0	Inspect the field alongside the practical route above.
float MinimapRefreshInterval = .1f	RPG Maps & Compass Capture	Editable	Min 0.033, Max 2.0	Inspect the field alongside the practical route above.
float CaptureHeightPadding = 50000.f	RPG Maps & Compass Capture	Editable	Min 1000.0	Inspect the field alongside the practical route above.
float CaptureExposureBias = 0.f	RPG Maps & Compass Capture	Editable	Min -8.0, Max 4.0	Fixed compensation for the map-only manual exposure. Negative values darken very bright landscapes.
FName AdditionalBoundsActorTag = TEXT("RMC.MapBounds")	RPG Maps & Compass Capture	Editable		Inspect the field alongside the practical route above.
FName TerrainGeometryActorTag = TEXT("RMC.MapGeometry")	RPG Maps & Compass Capture	Editable		Add this tag to mesh-based terrain or roads that should be drawn with Landscape actors on the generated map.

Native configuration reference: RMCMapSourceComponent.h

Names and defaults below are extracted from the current source declaration. Deprecated fields preserve older serialized assets; use their documented replacements for new setup. This table documents configuration, not successful runtime coverage of every field.

Declaration / default	Category	Access	Editor limits	Purpose
FText RegionName	RPG Maps & Compass	Editable		Inspect the field alongside the practical route above.
TObjectPtr<UTexture> OverviewTexture	RPG Maps & Compass	Editable		Optional authored or runtime-generated north-up overview.
TObjectPtr<UTexture> MinimapTexture	RPG Maps & Compass	Read-only / diagnostic		Optional live local capture. When absent, the minimap falls back to the overview.
TObjectPtr<UTexture> SceneMeshTexture	RPG Maps & Compass	Read-only / diagnostic		North-up actual mesh color capture; alpha stores inverse coverage. Shares overview UVs.
TObjectPtr<UTexture> GeometryDepthTexture	RPG Maps & Compass Relief	Read-only / diagnostic		North-up linear scene depth aligned to the overview; drives geometry form without moving landmarks.

Declaration / default	Category	Access	Editor limits	Purpose
FLinearColor GeometryDepthMetrics = FLinearColor::Black	RPG Maps & Compass Relief	Read-only / diagnostic		(inverse width, inverse height, world units per U texel, world units per V texel).
float GeometryDepthFar = 0.f	RPG Maps & Compass Relief	Read-only / diagnostic		Clear depth used to reject pixels that contain no captured surface.
TObjectPtr<UTexture> SurfaceNormalTexture	RPG Maps & Compass Deprecated	Deprecated compatibility field		Deprecated normal companion retained only so older placed actors deserialize safely. Always null.
TObjectPtr<UTexture> ReliefTexture	RPG Maps & Compass Relief	Read-only / diagnostic		Cached elevation-derived lighting, never a replacement for material colors.
float SurfaceDetailSoftening = .15f	RPG Maps & Compass Artistic Treatment	Editable	Min 0, Max 1	Suppress small repeating surface detail in the captured map without moving geography or markers.
float ReliefStrength = .65f	RPG Maps & Compass Relief	Editable	Min 0, Max 1	Inspect the field alongside the practical route above.
float GeometryHeightStrength = .55f	RPG Maps & Compass Relief	Editable	Min 0, Max 1	Reconstruct restrained directional form from captured geometry depth.
float GeometryContactStrength = .22f	RPG Maps & Compass Relief	Editable	Min 0, Max 1	Darken lower surfaces immediately beside taller geometry such as rocks and tree canopies.
float GeometryReliefStrength = 0.f	RPG Maps & Compass Deprecated	Deprecated compatibility field		Deprecated normal-lighting strength retained for older serialized sources.
float ReliefContrast = 1.35f	RPG Maps & Compass Artistic Treatment	Editable	Min 0.5, Max 2	Shape the measured relief around neutral lighting so ridges and escarpments read at map scale.
float ReliefShadowFloor = .55f	RPG Maps & Compass Artistic Treatment	Editable	Min 0.25, Max 1	Prevent the artistic relief pass from crushing valleys and back-facing slopes to black.
float CliffInkStrength = .25f	RPG Maps & Compass Artistic Treatment	Editable	Min 0, Max 1	Add restrained ink-like definition only where measured relief departs from flat terrain.
float ContourStrength = 0.f	RPG Maps & Compass Relief	Editable	Min 0, Max 1	Inspect the field alongside the practical route above.
bool bUseOwnerLocationAsCenter = true	RPG Maps & Compass	Editable		Inspect the field alongside the practical route above.
FVector2D WorldCenter = FVector2D::ZeroVector	RPG Maps & Compass	Editable		Inspect the field alongside the practical route above.
FVector2D WorldExtent = FVector2D(200000.f, 200000.f)	RPG Maps & Compass	Editable	Min 100.0	Inspect the field alongside the practical route above.
int32 Priority = 0	RPG Maps & Compass	Editable		Inspect the field alongside the practical route above.
bool bActive = true	RPG Maps & Compass	Editable		Inspect the field alongside the practical route above.

Verified native base-color orientation

This authentic GPU capture uses the existing host's calibration floor and two geometry landmarks. North is +world X; east is +world Y. The native scene's floor material uses a fixed inverse UV rotation to match its mesh mapping. Your own mesh UVs may require a different calibration.

The legacy lit comparison at exposure bias -3 remained pale and washed out in this validation scene. The world-map UI also stretches the calibration image at its current layout. Neither image is approved for final presentation; they are excluded from this guide's illustrations.

Placed native parameter views

These seven views display39 real, noneditable Unreal PropertyEditor handles from the placed native actor. They match its45 public native declarations with six structural component references excluded. Private transient targets are not Details settings. The generic native PropertyEditor framing hides component flag subpanels and groups actual property rows; it changes the view, not the saved values. Actor transform rows remain visible for context. Diagnostics here are saved/editor values, not a claim that capture has already run.

Buyer native capture configuration

Transform

Location

0.00.00.0

Rotation

0.0°0.0°0.0°

Scale

1.01.01.0

Mobility

StaticStationaryMovable

RPG Maps & Compass - native parameters

Map Tone Material

Material

M_RMCMaTone_Artistic

Shape Surface Material

Material

M_RMCSHaSurface

Enable Scene Mesh Capture

Scene Mesh Resolution

2048

Scene Mesh Refresh Interval

0.5

Scene Mesh Ignore Tag

RMC.MapIgnore

Captured Scene Mesh Count

0

Auto Discover Landscape Bounds

Refresh when Terrain Changes

Detection Status

Sample Coverage

0.0

Capture Overview on Begin Play

Enable Live Minimap Capture

Use Material Base Color

Capture Rendered Terrain Materials

Generate Topography from Terrain

Use Landscape Paint Layers

Pastel Strength

1.0

Paper Grain Strength

0.35

Cartography Ink

Cartography Paper

Cartography Ocean

Landscape Layer Color Overrides

0 Map elements

Topography Resolution

512

Sea Level Actor Tag

RMC.MapSeaLevel

Water Route Actor Tag

RMC.MapWater

Water Route Width

1800.0

Overview Resolution

1024

Geometry Depth Resolution

512

Capture Geometry Depth

Surface Normal Resolution

2048

Capture Surface Normals

Minimap Resolution

512

Initial Capture Delay

1.5

Minimap Refresh Interval

0.1

Capture Height Padding

50000.0

Capture Exposure Bias

-3.0

Additional Bounds Actor Tag

RMC.MapBounds

Terrain Geometry Actor Tag

RMC.MapGeometry

BP_RMC_BuyerCapture / NativeParameters-All - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.00.00.0

Rotation

0.0°0.0°0.0°

Scale

1.01.01.0

Mobility

StaticStationaryMovable

RPG Maps & Compass - Capture

Auto Discover Landscape Bounds

Refresh when Terrain Changes

Capture Overview on Begin Play

Enable Live Minimap Capture

Use Material Base Color

Capture Rendered Terrain Materials

Overview Resolution

1024

Geometry Depth Resolution

512

Capture Geometry Depth

Minimap Resolution

512

Initial Capture Delay

1.5

Minimap Refresh Interval

0.1

Capture Height Padding

50000.0

Capture Exposure Bias

-3.0

Additional Bounds Actor Tag

RMC.MapBounds

Terrain Geometry Actor Tag

RMC.MapGeometry

BP_RMC_BuyerCapture / NativeParameters-Capture - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.00.00.0

Rotation

0.0°0.0°0.0°

Scale

1.01.01.0

Mobility

StaticStationaryMovable

RPG Maps & Compass - Scene Geometry

Enable Scene Mesh Capture

Scene Mesh Resolution

2048

Scene Mesh Refresh Interval

0.5

Scene Mesh Ignore Tag

RMC.MapIgnore

BP_RMC_BuyerCapture / NativeParameters-SceneGeometry - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.00.00.0

Rotation

0.0°0.0°0.0°

Scale

1.01.01.0

Mobility

StaticStationaryMovable

RPG Maps & Compass - Topography

Generate Topography from Terrain

Use Landscape Paint Layers

Landscape Layer Color Overrides

0 Map elements

Topography Resolution

512

Sea Level Actor Tag

RMC.MapSeaLevel

Water Route Actor Tag

RMC.MapWater

Water Route Width

1800.0

BP_RMC_BuyerCapture / NativeParameters-Topography - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.0

0.0

0.0

Rotation

0.0°

0.0°

0.0°

Scale

1.0

1.0

1.0

Mobility

Static

Stationary

Movable

RPG Maps & Compass - Cartography

Map Tone Material

Material

M_RMCMaTone_Artistic

Shape Surface Material

Material

M_RMCSHaPeSurface

Pastel Strength

1.0

Paper Grain Strength

0.35

Cartography Ink

Cartography Paper

Cartography Ocean

BP_RMC_BuyerCapture / NativeParameters-Cartography - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.0

0.0

0.0

Rotation

0.0°

0.0°

0.0°

Scale

1.0

1.0

1.0

Mobility

Static

Stationary

Movable

RPG Maps & Compass - Diagnostics

Captured Scene Mesh Count

0

Detection Status

Sample Coverage

0.0

BP_RMC_BuyerCapture / NativeParameters-Diagnostics - retained configuration/reference capture | Open full-resolution original

Buyer native capture configuration

Transform

Location

0.0

0.0

0.0

Rotation

0.0°

0.0°

0.0°

Scale

1.0

1.0

1.0

Mobility

Static

Stationary

Movable

RPG Maps & Compass - Deprecated

Surface Normal Resolution

2048

Capture Surface Normals

BP_RMC_BuyerCapture / NativeParameters-Deprecated - retained configuration/reference capture | Open full-resolution original

Create a native landmark actor

1. Create a Blueprint subclass of RMC Map Marker Actor in your existing game Content and place it at the landmark.
2. On its Marker component, set Marker Label, Category, Marker Color and Show On Minimap / Compass / World Map.
3. Play with the owning local player and query Get Visible Markers. Use the returned Id and World Location in your UI; the actor does not require a visible mesh.

The saved BP_RMC_BuyerNativeMarker now has clean scoped public read/explicit registration proof and authentic native captures. Automatic local-player BeginPlay/EndPlay and designed UI remain separate coverage gaps.

11. C++ integration

Use your existing C++ game module

1. Copy MapsBuyerCheckpoint.h into your game module's Public folder and MapsBuyerCheckpoint.cpp into Private.
2. Add Core, CoreUObject, Engine and RPGMapsCompass to that module's dependencies. Compile the existing module.
3. Create AMapsBuyerCheckpoint in your world and call Connect with the exact owning local controller. Show Failure and stop on false.
4. Your input supplies a world destination and name to SetDestination. Retain its GUID through the sample; use its observer to refresh your UI.
5. ReadView, ReadProgress and ReadSettings provide state or refusal. ApplySettings and SaveProgress provide separate disk receipts. Cancel and Disconnect route dismissal and scoped cleanup.

[Read the complete native instructions](#) | [Header](#) | [Implementation](#)

12. Included examples

Enable Show Plugin Content in the Content Browser. Find the assets below, duplicate the examples into your own game Content to customize them, and follow the linked numbered hookups.

Region setup | Direct caller | Struct UI and native marker | Capture settings

Asset	Plugin package
BP_RMC_BuyerRegion	/RPGMapsCompass/RPGMapsCompass/Examples/BP_RMC_BuyerRegion
BP_RMC_BuyerCalibration	/RPGMapsCompass/RPGMapsCompass/Examples/Blueprints/BP_RMC_BuyerCalibration
BP_RMC_BuyerCapture	/RPGMapsCompass/RPGMapsCompass/Examples/Blueprints/BP_RMC_BuyerCapture
BP_RMC_DirectCaller	/RPGMapsCompass/RPGMapsCompass/Examples/Blueprints/BP_RMC_DirectCaller
M_RMC_BuyerCalibrationSurface	/RPGMapsCompass/RPGMapsCompass/Examples/Materials/M_RMC_BuyerCalibrationSurface
M_RMC_BuyerLandmark	/RPGMapsCompass/RPGMapsCompass/Examples/Materials/M_RMC_BuyerLandmark
T_RMC_BuyerCalibration	/RPGMapsCompass/RPGMapsCompass/Examples/Textures/T_RMC_BuyerCalibration
L_RMC_BuyerCalibration	/RPGMapsCompass/RPGMapsCompass/Examples/Maps/L_RMC_BuyerCalibration
WBP_RMC_BuyerStructUI	/RPGMapsCompass/RPGMapsCompass/Examples/UI/WBP_RMC_BuyerStructUI
BP_RMC_BuyerNativeMarker	/RPGMapsCompass/RPGMapsCompass/Examples/Blueprints/BP_RMC_BuyerNativeMarker

The calibration map demonstrates bounds and authored map imagery. Your existing project supplies the player, controller, input, and finished HUD design. The struct widget exposes data adapters; it is not a completed gameplay interface.

13. API and field reference

Reference scope

This source inventory lists 70 reflected functions and 150 reflected properties, including internal storage declarations. A listed declaration is not an assertion that it is buyer-editable or has full runtime/screenshot coverage. Read Access Metadata and use the documented public route. The tested C10 source matches the inventory source; C10 changed the saved region asset.

Reflected functions

Owner / name	Declaration	Access metadata
ARMCBuyerRegionExample.ConnectBuyerNavigation	bool ConnectBuyerNavigation(URMCNavigationSubsystem* Navigation, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.RequestBuyerMapView	bool RequestBuyerMapView(float Zoom, FVector2D Pan, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.RequestBuyerWaypoint	bool RequestBuyerWaypoint(FVector2D MapUV, const FString& Name, FGuid& WaypointId, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.RenameBuyerWaypoint	bool RenameBuyerWaypoint(FGuid WaypointId, const FString& Name, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.RemoveBuyerWaypoint	bool RemoveBuyerWaypoint(FGuid WaypointId, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.ApplyBuyerSettings	bool ApplyBuyerSettings(const FRMCNavigationSettings& NewSettings, FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.SaveBuyerProgress	bool SaveBuyerProgress(FText& Failure)	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.CancelBuyerNavigation	bool CancelBuyerNavigation()	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.DisconnectBuyerNavigation	bool DisconnectBuyerNavigation()	BlueprintNativeEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.StartBuyerNavigation	bool StartBuyerNavigation(APlayerController* OwningPlayer, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.ReadBuyerViewState	bool ReadBuyerViewState(FVector& PlayerLocation, float& PlayerYaw, FVector2D& MapPan, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example State"
ARMCBuyerRegionExample.ReadBuyerProgressState	bool ReadBuyerProgressState(int32& ExploredCellCount, bool& ProgressReadOnly, bool& ProgressSavePending, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example State"
ARMCBuyerRegionExample.ReadBuyerSettings	bool ReadBuyerSettings(FRMCNavigationSettings& CurrentSettings, FString& SaveLabel, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example State"
ARMCBuyerRegionExample.EditBuyerExplorationSettings	bool EditBuyerExplorationSettings(bool FogOfWar, float ExplorationRadius, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example Configuration"
ARMCBuyerRegionExample.SetBuyerSurfaceState	bool SetBuyerSurfaceState(bool WorldMapOpen, bool CompassVisible, bool ExternalSuiteCoordinatorActive, FText& Failure)	BlueprintImplementableEvent, BlueprintCallable, Category="RPG Maps & Compass Buyer Example Configuration"
ARMCBuyerRegionExample.GetBuyerWaypointSnapshot	TArray<FRMCUserWaypoint> GetBuyerWaypointSnapshot() const	BlueprintPure, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.BuyerWaypointStateChanged	void BuyerWaypointStateChanged()	BlueprintImplementableEvent, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.HandleWaypointsChanged	void HandleWaypointsChanged()	
URMCMapMarkerComponent.GetMarkerView	FRMCMapMarkerView GetMarkerView() const	BlueprintPure, Category="RPG Maps & Compass"
ARMCMapSourceActor.CaptureLevelOverview	void CaptureLevelOverview()	BlueprintCallable, CallInEditor, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.RefreshCapturedImage	void RefreshCapturedImage()	BlueprintCallable, CallInEditor, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.CaptureLiveMinimap	void CaptureLiveMinimap()	BlueprintCallable, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.CaptureSceneMeshes	void CaptureSceneMeshes()	BlueprintCallable, CallInEditor, Category="RPG Maps & Compass Capture"
URMCMapSourceComponent.GetResolvedWorldCenter	FVector2D GetResolvedWorldCenter() const	BlueprintPure, Category="RPG Maps & Compass"
URMCMapSourceComponent.GetSafeWorldExtent	FVector2D GetSafeWorldExtent() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.ToggleWorldMap	void ToggleWorldMap()	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SetWorldMapOpen	void SetWorldMapOpen(bool bOpen)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.ToggleCompass	void ToggleCompass()	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SetCompassVisible	void SetCompassVisible(bool bVisible)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.IsWorldMapOpen	bool IsWorldMapOpen() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.IsMinimapSettingsOpen	bool IsMinimapSettingsOpen() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SetNavigationSettings	void SetNavigationSettings(const FRMCNavigationSettings& NewSettings)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.GetNavigationSettings	FRMCNavigationSettings GetNavigationSettings() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SaveNavigationSettings	void SaveNavigationSettings()	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SaveSetupVerified	bool SaveSetupVerified()	BlueprintCallable, Category="RPG Maps & Compass"

Owner / name	Declaration	Access metadata
URMCNavigationSubsystem.GetSettingsSaveLabel	FString GetSettingsSaveLabel() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.ResetLauncherPosition	void ResetLauncherPosition()	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.MatchMapColors	void MatchMapColors(bool bDestinationWorldMap)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.DismissTopSurface	bool DismissTopSurface()	BlueprintCallable, Category="RPG Maps & Compass Window"
URMCNavigationSubsystem.ResetNavigationSettings	void ResetNavigationSettings()	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.SetExternalSuiteCoordinatorActive	void SetExternalSuiteCoordinatorActive(bool bActive)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.IsStandaloneLauncherSuppressed	bool IsStandaloneLauncherSuppressed() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.RegisterMarker	void RegisterMarker(URMCMapMarkerComponent* Marker)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.UnregisterMarker	void UnregisterMarker(URMCMapMarkerComponent* Marker)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.GetVisibleMarkers	TArray<FRMCMapMarkerView> GetVisibleMarkers() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.RegisterMapSource	void RegisterMapSource(URMCMapSourceComponent* MapSource)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.UnregisterMapSource	void UnregisterMapSource(URMCMapSourceComponent* MapSource)	BlueprintCallable, Category="RPG Maps & Compass"
URMCNavigationSubsystem.GetActiveMapSource	URMCMapSourceComponent* GetActiveMapSource() const	BlueprintPure, Category="RPG Maps & Compass"
URMCNavigationSubsystem.IsPlayerPoseReady	bool IsPlayerPoseReady() const	BlueprintPure, Category="RPG Maps & Compass Player"
URMCNavigationSubsystem.GetPlayerLocation	FVector GetPlayerLocation() const	BlueprintPure, Category="RPG Maps & Compass Player"
URMCNavigationSubsystem.GetPlayerYaw	float GetPlayerYaw() const	BlueprintPure, Category="RPG Maps & Compass Player"
URMCNavigationSubsystem.TrySetWorldMapView	bool TrySetWorldMapView(float Zoom, FVector2D Pan)	BlueprintCallable, Category="RPG Maps & Compass Map View"
URMCNavigationSubsystem.SetMinimapSettingsOpen	void SetMinimapSettingsOpen(bool bOpen)	BlueprintCallable, Category="RPG Maps & Compass Window"
URMCNavigationSubsystem.OpenMapSettings	void OpenMapSettings(bool bWorldMapSection)	BlueprintCallable, Category="RPG Maps & Compass Window"
URMCNavigationSubsystem.GetWorldMapPan	FVector2D GetWorldMapPan() const	BlueprintPure, Category="RPG Maps & Compass Map View"
URMCNavigationSubsystem.SetFogOfWarEnabled	void SetFogOfWarEnabled(bool bEnabled)	BlueprintCallable, Category="RPG Maps & Compass Exploration"
URMCNavigationSubsystem.TryAddUserWaypoint	bool TryAddUserWaypoint(FVector Location, const FString& Name, FGuid& WaypointId, FText& Failure)	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.AddUserWaypoint	bool AddUserWaypoint(FVector Location, const FString& Name)	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.RemoveUserWaypoint	bool RemoveUserWaypoint(FGuid WaypointId)	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.RenameUserWaypoint	bool RenameUserWaypoint(FGuid WaypointId, const FString& Name)	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.IsMarkerCategoryVisible	bool IsMarkerCategoryVisible(ERMCMarkerCategory Category) const	BlueprintPure, Category="RPG Maps & Compass Markers"
URMCNavigationSubsystem.ClearUserWaypoints	void ClearUserWaypoints()	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.GetUserWaypoints	TArray<FRMCUserWaypoint> GetUserWaypoints() const	BlueprintPure, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.ResolveMapLocation	bool ResolveMapLocation(FVector2D UV, FVector& OutLocation) const	BlueprintCallable, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.ProjectWaypoint	bool ProjectWaypoint(FVector Location, FVector2D& Screen) const	BlueprintPure, Category="RPG Maps & Compass Waypoints"
URMCNavigationSubsystem.TrySaveProgressVerified	bool TrySaveProgressVerified(FText& Failure)	BlueprintCallable, Category="RPG Maps & Compass Exploration"
URMCNavigationSubsystem.IsProgressReady	bool IsProgressReady() const	BlueprintPure, Category="RPG Maps & Compass Exploration"
URMCNavigationSubsystem.GetExploredCellCount	int32 GetExploredCellCount() const	BlueprintPure, Category="RPG Maps & Compass Exploration"
URMCNavigationSubsystem.IsProgressReadOnly	bool IsProgressReadOnly() const	BlueprintPure, Category="RPG Maps & Compass Exploration"
URMCNavigationSubsystem.IsProgressSavePending	bool IsProgressSavePending() const	BlueprintPure, Category="RPG Maps & Compass Exploration"

Reflected fields

Owner / name	Declaration	Access metadata
ARMCBuyerRegionExample.SceneRoot	TObjectPtr<USceneComponent> SceneRoot	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.MapSource	TObjectPtr<URMCMapSourceComponent> MapSource	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.Marker	TObjectPtr<URMCMapMarkerComponent> Marker	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.ConnectedNavigation	TObjectPtr<URMCNavigationSubsystem> ConnectedNavigation	BlueprintReadOnly, Transient, Category="RPG Maps & Compass Buyer Example"
ARMCBuyerRegionExample.WaypointStateNotifications	int32 WaypointStateNotifications = 0	BlueprintReadOnly, Transient, Category="RPG Maps & Compass Buyer Example"
URMCExplorationSave.Schema	int32 Schema = 1	
URMCExplorationSave.Cells	TArray<FIntPoint> Cells	
URMCExplorationSave.Waypoints	TArray<FRMCUserWaypoint> Waypoints	
ARMCMapMarkerActor.SceneRoot	TObjectPtr<USceneComponent> SceneRoot	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapMarkerActor.Marker	TObjectPtr<URMCMapMarkerComponent> Marker	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
URMCMapMarkerComponent.MarkerLabel	FText MarkerLabel	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapMarkerComponent.Category	ERMCMarkerCategory Category = ERMCMarkerCategory::Landmark	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapMarkerComponent.MarkerColor	FLinearColor MarkerColor = FLinearColor(.12f, .55f, 1.f, 1.f)	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapMarkerComponent.bShowOnMinimap	bool bShowOnMinimap = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapMarkerComponent.bShowOnCompass	bool bShowOnCompass = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapMarkerComponent.bShowOnWorldMap	bool bShowOnWorldMap = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
ARMCMapSourceActor.MapToneMaterial	TObjectPtr<UMaterialInterface> MapToneMaterial	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Cartography"
ARMCMapSourceActor.ShapeSurfaceMaterial	TObjectPtr<UMaterialInterface> ShapeSurfaceMaterial	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Cartography"
ARMCMapSourceActor.bEnableSceneMeshCapture	bool bEnableSceneMeshCapture = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Scene Geometry"
ARMCMapSourceActor.SceneMeshResolution	int32 SceneMeshResolution = 2048	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Scene Geometry", meta=(ClampMin="256",ClampMax="2048")
ARMCMapSourceActor.SceneMeshRefreshInterval	float SceneMeshRefreshInterval = .5f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Scene Geometry", meta=(ClampMin="0.1",ClampMax="10")
ARMCMapSourceActor.SceneMeshIgnoreTag	FName SceneMeshIgnoreTag = TEXT("RMC.MapIgnore")	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Scene Geometry"
ARMCMapSourceActor.CapturedSceneMeshCount	int32 CapturedSceneMeshCount = 0	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Diagnostics"
ARMCMapSourceActor.SceneRoot	TObjectPtr<USceneComponent> SceneRoot	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapSourceActor.MapSource	TObjectPtr<URMCMapSourceComponent> MapSource	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapSourceActor.OverviewCapture	TObjectPtr<USceneCaptureComponent2D> OverviewCapture	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapSourceActor.MinimapCapture	TObjectPtr<USceneCaptureComponent2D> MinimapCapture	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapSourceActor.GeometryDepthCapture	TObjectPtr<USceneCaptureComponent2D> GeometryDepthCapture	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass"
ARMCMapSourceActor.SurfaceNormalCapture	TObjectPtr<USceneCaptureComponent2D> SurfaceNormalCapture	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Deprecated", meta=(DeprecatedProperty, DeprecationMessage="Use GeometryDepthCapture")
ARMCMapSourceActor.bAutoDiscoverLandscapeBounds	bool bAutoDiscoverLandscapeBounds = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.bRefreshWhenTerrainChanges	bool bRefreshWhenTerrainChanges = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"

Owner / name	Declaration	Access metadata
ARMCMapSourceActor.DetectionStatus	FString DetectionStatus	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Diagnostics"
ARMCMapSourceActor.SampleCoverage	float SampleCoverage = 0.f	VisibleAnywhere, BlueprintReadOnly, Category="RPG Maps & Compass Diagnostics"
ARMCMapSourceActor.bCaptureOverviewOnBeginPlay	bool bCaptureOverviewOnBeginPlay = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.bEnableLiveMinimapCapture	bool bEnableLiveMinimapCapture = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.bUseMaterialBaseColor	bool bUseMaterialBaseColor = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.bCaptureRenderedTerrainMaterials	bool bCaptureRenderedTerrainMaterials = false	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.bGenerateTopographyFromTerrain	bool bGenerateTopographyFromTerrain = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography"
ARMCMapSourceActor.bUseLandscapePaintLayers	bool bUseLandscapePaintLayers = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography"
ARMCMapSourceActor.PastelStrength	float PastelStrength = 1.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Cartography", meta=(ClampMin="0", ClampMax="1")
ARMCMapSourceActor.PaperGrainStrength	float PaperGrainStrength = .35f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Cartography", meta=(ClampMin="0", ClampMax="1")
ARMCMapSourceActor.CartographyInk	FLinearColor CartographyInk = FLinearColor(.065f,.035f,.085f)	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Cartography"
ARMCMapSourceActor.CartographyPaper	FLinearColor CartographyPaper = FLinearColor(.74f,.57f,.34f)	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Cartography"
ARMCMapSourceActor.CartographyOcean	FLinearColor CartographyOcean = FLinearColor(.035f,.12f,.19f)	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Cartography"
ARMCMapSourceActor.LandscapeLayerColorOverrides	TMap<FName, FLinearColor> LandscapeLayerColorOverrides	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography"
ARMCMapSourceActor.TopographyResolution	int32 TopographyResolution = 2048	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography", meta=(ClampMin="128", ClampMax="2048")
ARMCMapSourceActor.SeaLevelActorTag	FName SeaLevelActorTag = TEXT("RMC.MapSeaLevel")	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography"
ARMCMapSourceActor.WaterRouteActorTag	FName WaterRouteActorTag = TEXT("RMC.MapWater")	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography"
ARMCMapSourceActor.WaterRouteWidth	float WaterRouteWidth = 1800.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Topography", meta=(ClampMin="100.0", ClampMax="10000.0")
ARMCMapSourceActor.OverviewResolution	int32 OverviewResolution = 4096	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="256", ClampMax="4096")
ARMCMapSourceActor.GeometryDepthResolution	int32 GeometryDepthResolution = 1024	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="256", ClampMax="1024")
ARMCMapSourceActor.bCaptureGeometryDepth	bool bCaptureGeometryDepth = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.SurfaceNormalResolution	int32 SurfaceNormalResolution = 2048	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Deprecated", meta=(DeprecatedProperty, DeprecationMessage="Use GeometryDepthResolution")
ARMCMapSourceActor.bCaptureSurfaceNormals	bool bCaptureSurfaceNormals = false	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Deprecated", meta=(DeprecatedProperty, DeprecationMessage="Use bCaptureGeometryDepth")
ARMCMapSourceActor.MinimapResolution	int32 MinimapResolution = 512	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="128", ClampMax="2048")
ARMCMapSourceActor.InitialCaptureDelay	float InitialCaptureDelay = 1.5f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="0.0", ClampMax="5.0")
ARMCMapSourceActor.MinimapRefreshInterval	float MinimapRefreshInterval = .1f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="0.033", ClampMax="2.0")
ARMCMapSourceActor.CaptureHeightPadding	float CaptureHeightPadding = 50000.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="1000.0")
ARMCMapSourceActor.CaptureExposureBias	float CaptureExposureBias = 0.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture", meta=(ClampMin="-8.0", ClampMax="4.0")
ARMCMapSourceActor.AdditionalBoundsActorTag	FName AdditionalBoundsActorTag = TEXT("RMC.MapBounds")	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.TerrainGeometryActorTag	FName TerrainGeometryActorTag = TEXT("RMC.MapGeometry")	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Capture"
ARMCMapSourceActor.SceneMeshCapture	TObjectPtr<USceneCaptureComponent2D> SceneMeshCapture	Transient

Owner / name	Declaration	Access metadata
ARMCMapSourceActor.SceneMeshTarget	TObjectPtr<UTextureRenderTarget2D> SceneMeshTarget	Transient
ARMCMapSourceActor.CapturedTerrainActors	TArray<TObjectPtr<AActor>> CapturedTerrainActors	Transient
ARMCMapSourceActor.GeneratedTopographyTexture	TObjectPtr<UTexture2D> GeneratedTopographyTexture	Transient
ARMCMapSourceActor.RuntimeOverviewTarget	TObjectPtr<UTextureRenderTarget2D> RuntimeOverviewTarget	Transient
ARMCMapSourceActor.RuntimeGeometryDepthTarget	TObjectPtr<UTextureRenderTarget2D> RuntimeGeometryDepthTarget	Transient
ARMCMapSourceActor.RuntimeSurfaceNormalTarget	TObjectPtr<UTextureRenderTarget2D> RuntimeSurfaceNormalTarget	Transient, meta=(DeprecatedProperty, DeprecationMessage="Use RuntimeGeometryDepthTarget")
ARMCMapSourceActor.RuntimeMinimapTarget	TObjectPtr<UTextureRenderTarget2D> RuntimeMinimapTarget	Transient
URMCMapSourceComponent.RegionName	FText RegionName	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapSourceComponent.OverviewTexture	TObjectPtr<UTexture> OverviewTexture	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapSourceComponent.MinimapTexture	TObjectPtr<UTexture> MinimapTexture	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass"
URMCMapSourceComponent.SceneMeshTexture	TObjectPtr<UTexture> SceneMeshTexture	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass"
URMCMapSourceComponent.GeometryDepthTexture	TObjectPtr<UTexture> GeometryDepthTexture	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass Relief"
URMCMapSourceComponent.GeometryDepthMetrics	FLinearColor GeometryDepthMetrics = FLinearColor::Black	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass Relief"
URMCMapSourceComponent.GeometryDepthFar	float GeometryDepthFar = 0.f	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass Relief"
URMCMapSourceComponent.SurfaceNormalTexture	TObjectPtr<UTexture> SurfaceNormalTexture	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass Deprecated", meta=(DeprecatedProperty, DeprecationMessage="Use GeometryDepthTexture")
URMCMapSourceComponent.ReliefTexture	TObjectPtr<UTexture> ReliefTexture	VisibleAnywhere, BlueprintReadOnly, Transient, Category="RPG Maps & Compass Relief"
URMCMapSourceComponent.SurfaceDetailSoftening	float SurfaceDetailSoftening = .15f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Artistic Treatment", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.ReliefStrength	float ReliefStrength = .65f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Relief", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.GeometryHeightStrength	float GeometryHeightStrength = .55f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Relief", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.GeometryContactStrength	float GeometryContactStrength = .22f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Relief", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.GeometryReliefStrength	float GeometryReliefStrength = 0.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Deprecated", meta=(DeprecatedProperty, DeprecationMessage="Use GeometryHeightStrength")
URMCMapSourceComponent.ReliefContrast	float ReliefContrast = 1.35f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Artistic Treatment", meta=(ClampMin="0.5", ClampMax="2")
URMCMapSourceComponent.ReliefShadowFloor	float ReliefShadowFloor = .55f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Artistic Treatment", meta=(ClampMin="0.25", ClampMax="1")
URMCMapSourceComponent.CliffInkStrength	float CliffInkStrength = .25f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Artistic Treatment", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.ContourStrength	float ContourStrength = 0.f	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass Relief", meta=(ClampMin="0", ClampMax="1")
URMCMapSourceComponent.bUseOwnerLocationAsCenter	bool bUseOwnerLocationAsCenter = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapSourceComponent.WorldCenter	FVector2D WorldCenter = FVector2D::ZeroVector	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapSourceComponent.WorldExtent	FVector2D WorldExtent = FVector2D(200000.f, 200000.f)	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass", meta=(ClampMin="100.0")
URMCMapSourceComponent.Priority	int32 Priority = 0	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCMapSourceComponent.bActive	bool bActive = true	EditAnywhere, BlueprintReadWrite, Category="RPG Maps & Compass"
URMCNavigationSubsystem.OnWaypointsChanged	FRMCWaypointsChanged OnWaypointsChanged	BlueprintAssignable, Category="RPG Maps & Compass Waypoints"

Owner / name	Declaration	Access metadata
URMCNavigationSubsystem.PendingProgressSaves	TMap<FString, TObjectPtr<URMCExplorationSave>> PendingProgressSaves	Transient
URMCNavigationSubsystem.FogTexture	TObjectPtr<UTexture2D> FogTexture	Transient
URMCNavigationSubsystem.RegisteredMarkers	TArray<TObjectPtr<URMCMapMarkerComponent>> RegisteredMarkers	Transient
URMCNavigationSubsystem.RegisteredMapSources	TArray<TObjectPtr<URMCMapSourceComponent>> RegisteredMapSources	Transient
FRMCMapAppearance.Shape	ERMCMAPShape Shape = ERMCMAPShape::Rectangle	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCMapAppearance.PositionOffset	FVector2D PositionOffset = FVector2D::ZeroVector	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCMapAppearance.Scale	float Scale = 1.f	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCMapAppearance.Width	float Width = 1.f	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCMapAppearance.Height	float Height = 1.f	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCMapAppearance.ImageOpacity	float ImageOpacity = .75f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.Brightness	float Brightness = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style", meta=(ClampMin="0.25",ClampMax="2")
FRMCMapAppearance.Saturation	float Saturation = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style", meta=(ClampMin="0",ClampMax="2")
FRMCMapAppearance.BackgroundOpacity	float BackgroundOpacity = .25f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.BorderOpacity	float BorderOpacity = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.BorderThickness	float BorderThickness = 2.f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.TextScale	float TextScale = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.bShowText	bool bShowText = true	EditAnywhere, BlueprintReadWrite, Category="Labels"
FRMCMapAppearance.bShowDirections	bool bShowDirections = true	EditAnywhere, BlueprintReadWrite, Category="Labels"
FRMCMapAppearance.MarkerScale	float MarkerScale = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCMapAppearance.SettingsOpacity	float SettingsOpacity = .25f	EditAnywhere, BlueprintReadWrite, Category="Style"
FRMCNavigationSettings.MinimapAppearance	FRMCMapAppearance MinimapAppearance	EditAnywhere, BlueprintReadWrite, Category="Minimap"
FRMCNavigationSettings.WorldMapAppearance	FRMCMapAppearance WorldMapAppearance	EditAnywhere, BlueprintReadWrite, Category="World Map"
FRMCNavigationSettings.LauncherOffset	FVector2D LauncherOffset = FVector2D::ZeroVector	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCNavigationSettings.bLauncherPlacementCustomized	bool bLauncherPlacementCustomized = false	EditAnywhere, BlueprintReadWrite, Category="Layout"
FRMCNavigationSettings.bShowWorldWaypoints	bool bShowWorldWaypoints = true	EditAnywhere, BlueprintReadWrite, Category="Waypoints"
FRMCNavigationSettings.bShowWaypointDistance	bool bShowWaypointDistance = true	EditAnywhere, BlueprintReadWrite, Category="Waypoints"
FRMCNavigationSettings.bFitSmallMaps	bool bFitSmallMaps = true	EditAnywhere, BlueprintReadWrite, Category="Minimap"
FRMCNavigationSettings.bShowMinimap	bool bShowMinimap = true	EditAnywhere, BlueprintReadWrite, Category="Visibility"
FRMCNavigationSettings.bShowMinimapClock	bool bShowMinimapClock = true	EditAnywhere, BlueprintReadWrite, Category="Minimap Clock"
FRMCNavigationSettings.bUse24HourClock	bool bUse24HourClock = false	EditAnywhere, BlueprintReadWrite, Category="Minimap Clock"
FRMCNavigationSettings.bPersistentCursor	bool bPersistentCursor = true	EditAnywhere, BlueprintReadWrite, Category="Input"
FRMCNavigationSettings.bAutoCreateMapSource	bool bAutoCreateMapSource = true	EditAnywhere, BlueprintReadWrite, Category="Detection"
FRMCNavigationSettings.bFogOfWar	bool bFogOfWar = false	EditAnywhere, BlueprintReadWrite, Category="Exploration"
FRMCNavigationSettings.ExplorationRadius	float ExplorationRadius = 5000.f	EditAnywhere, BlueprintReadWrite, Category="Exploration", meta=(ClampMin="1000",ClampMax="20000")
FRMCNavigationSettings.bShowCompass	bool bShowCompass = false	EditAnywhere, BlueprintReadWrite, Category="Visibility"
FRMCNavigationSettings.bShowDemoMarkers	bool bShowDemoMarkers = false	EditAnywhere, BlueprintReadWrite, Category="Visibility"
FRMCNavigationSettings.HiddenMarkerCategories	TArray<ERMCMarkerCategory> HiddenMarkerCategories	EditAnywhere, BlueprintReadWrite, Category="Markers"
FRMCNavigationSettings.bRotateMinimap	bool bRotateMinimap = false	EditAnywhere, BlueprintReadWrite, Category="Minimap"
FRMCNavigationSettings.MinimapScale	float MinimapScale = 1.f	EditAnywhere, BlueprintReadWrite, Category="Minimap", meta=(ClampMin="0.5", ClampMax="2.0")
FRMCNavigationSettings.MinimapRange	float MinimapRange = 3000.f	EditAnywhere, BlueprintReadWrite, Category="Minimap", meta=(ClampMin="1000.0", ClampMax="50000.0")
FRMCNavigationSettings.MinimapZoom	float MinimapZoom = 1.f	EditAnywhere, BlueprintReadWrite, Category="Minimap", meta=(ClampMin="0.75", ClampMax="2.0")
FRMCNavigationSettings.WorldMapZoom	float WorldMapZoom = 1.f	EditAnywhere, BlueprintReadWrite, Category="World Map", meta=(ClampMin="0.25", ClampMax="4.0")

Owner / name	Declaration	Access metadata
FRMCNavigationSettings.bLandscapeWorldMap	bool bLandscapeWorldMap = true	EditAnywhere, BlueprintReadWrite, Category="World Map"
FRMCNavigationSettings.GlassOpacity	float GlassOpacity = .25f	EditAnywhere, BlueprintReadWrite, Category="Style", meta=(ClampMin="0.0", ClampMax="1.0")
FRMCNavigationSettings.BorderOpacity	float BorderOpacity = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style", meta=(ClampMin="0.0", ClampMax="1.0")
FRMCNavigationSettings.InterfaceScale	float InterfaceScale = 1.f	EditAnywhere, BlueprintReadWrite, Category="Style", meta=(ClampMin="0.75", ClampMax="2.0")
FRMCMapView.Id	FGuid Id	BlueprintReadOnly, Category="Marker"
FRMCMapView.Label	FText Label	BlueprintReadOnly, Category="Marker"
FRMCMapView.Category	ERMCMarkerCategory Category = ERMCMarkerCategory::Custom	BlueprintReadOnly, Category="Marker"
FRMCMapView.WorldLocation	FVector WorldLocation = FVector::ZeroVector	BlueprintReadOnly, Category="Marker"
FRMCMapView.Color	FLinearColor Color = FLinearColor(.12f, .55f, 1.f, 1.f)	BlueprintReadOnly, Category="Marker"
FRMCMapView.bShowOnMinimap	bool bShowOnMinimap = true	BlueprintReadOnly, Category="Marker"
FRMCMapView.bShowOnCompass	bool bShowOnCompass = true	BlueprintReadOnly, Category="Marker"
FRMCMapView.bShowOnWorldMap	bool bShowOnWorldMap = true	BlueprintReadOnly, Category="Marker"
FRMCUserWaypoint.Id	FGuid Id	EditAnywhere, BlueprintReadWrite, Category="Waypoint"
FRMCUserWaypoint.Name	FString Name	EditAnywhere, BlueprintReadWrite, Category="Waypoint"
FRMCUserWaypoint.Location	FVector Location = FVector::ZeroVector	EditAnywhere, BlueprintReadWrite, Category="Waypoint"

14. Scope and first-test checklist

First successful test in your game

1. Compile your duplicated examples, assign your map image/bounds and landmark settings, and start with a possessed local player.
2. Pass that controller into StartBuyerNavigation. Continue only on true; display Failure on false and reconnect when the owner is ready.
3. Open and close your map, change zoom/pan, and confirm cancellation restores the expected input.
4. Create a waypoint, retain its accepted GUID, rename it, and remove that same waypoint. Refresh UI from the change event and snapshot.
5. Change a settings copy, check its settings receipt, and separately save exploration progress. Restart your own test session and confirm the state you expect.
6. Disconnect before replacing the owner. Ensure your observer stops receiving events without removing another caller's data.

Scope and limitations

The plugin supplies local navigation data and presentation. Your game supplies map content, input ownership, quest logic, multiplayer authority and any cloud/profile synchronization. A waypoint is navigation data; it does not spawn an actor or grant a reward.

The examples demonstrate public integration patterns. The struct widget adapts data for your UI; it is not a complete quest HUD. The supplied calibration map is a setup reference, not a promise that your replacement content is packaged automatically.

Validate your own map bounds, authored imagery, replacement assets, input and packaged game. A successful compile or data read alone does not establish visual/gameplay acceptance.

Support

Email support.blackironspine@proton.me with your engine version, the named Blueprint/function, the failure text and reproducible steps. Keep your existing saves when investigating a refused progress write.