Using Multiple ZoneLayouts

This is a really useful feature. You might have a Tilemap Group comprising a Grid with, say, 8 child Tilemaps and want to use a Chunk Size of 16 and a second Group comprising another Grid with a single Tilemap, but you want a Chunk Size of 64 for this Group because you want to expose sections for a ‘fog of war’ type of situation (which can be done but isn’t further discussed here).

In this hypothetical case you’d have two TSceneSpecs for that TScene along with two TpZoneLayout components. Each TpZoneLayout component has a field that you can use to name it; you copy the names into the two TSceneSpecs.

Then when ChunkedSceneManager loads a Tile Scene (TScene) it can easily make the connections between Layouts and their data sources and load/unload tiles and prefabs from the two different Tilemap Groups automatically using two different ZoneLayouts with different Chunk Sizes.

Chunked Scene Manager reconfigures the TpZoneLayout parameters on a per-scene basis, so you only need as many TpZoneLayout components as the most complex TScene requires.

This is implemented in the demonstration program where a second ZoneLayout is used to load a border. This is an example of two TpZoneLayouts with two separate Grids: MainGrid and BorderGrid.

This uses a second TileFab and Selector in Chunking/DesignTileFabs/Border. This is a single-TileFab selector, which simply means that it returns the same TileFab every time.

Note that the Border-SingleFab selector has the LoadFlags set to None. This selector doesn’t support ‘Chunkifying’ but that’s not needed.

This Border TileFab was created in the same ‘DesignScene’ as the actual TileScenes, and the source tiles were painted on the “Overlay” Tilemap. But the name of the target Tilemap is different: BorderTilemap. Normally this would mean that TileFabLib will look for the Overlay map to place the TileFab’s tiles. But that’s not what we want. There are two ways to handle this:

- Edit the TilemapName field in the BorderTileFab asset.
- Provide a remapping Dictionary to LoadTileFab.

Just to illustrate how to do it, a remapping is performed in ChunkingGameController.LoadingFilter in the section that begins with if `(layout.m_LayoutName == "border")`. The remapping Dictionary is updated in OnAfterTSceneChange, where we know the Current Scene and can determine if there’s a border or not. This is true in Level 0 but not Level 1.

In general, it’s much easier to just change the name in the asset.

Revision #1
Created 11 July 2025 11:46:45 by Vonchor

Updated 11 July 2025 11:48:31 by Vonchor