Destructible Structure Builder Manual

Version: 1.0.0 **Compatibility**: Unity 2022.3 LTS, 2021.3 LTS (Unity 6 not yet supported)

Table of Contents

Overview

Destructible Structure Builder (DSB) is a Unity®-Editor toolkit for assembling gameplay-ready buildings that can splinter, crumble, and collapse in real-time. You author everything inside the main **DSB** window using a toolbar of *Build Modes* for placing connections, members, and walls. At runtime, dedicated components handle stress propagation, pooling, and event dispatch.

> ***Important:*** *All construction happens in the Unity Editor; the toolkit provides* ***no*** *in-game building or editing capabilities.*

Feature Highlights

- **Stress Solver** – graph-based system that finds the shortest path to ground, splits off unsupported chunks, and propagates structural load to detect overstressed members (see Advanced Topics → TechnicalInternals.md).
- **Voxel-Based Damage System** – Structures are built from voxels that split and detach individually on impact.
- **Chunk Merging & Splitting** – Voxels combine into chunks for performance and dynamically split when damaged.
- **Build Modes Toolbar** – Visual, mode-based system for placing nodes, beams, walls, stairs, and more.
- **Wall Designer** – cubes, windows, triangular cut-outs, per-cell health, and live rotation controls.
- **Design Presets** – save any wall as a `WallDesign` ScriptableObject and re-apply instantly.
- **Object Pooling** – pooled debris with adjustable lifetime, pool size, and spawn-rate.
- **Profile-driven effects** – plug-and-play audio/particles using two assets: an EffectsLibrary for per-event defaults and a MaterialEffectsLibrary for per-material overrides. Easily swap libraries per scene without code.
- **Mesh Cache** – persist generated meshes between sessions; safely prefab structures.
- **Full Unity Undo support** – every spawn, delete, or property change is wrapped in a single Undo group.

Package Contents

Folder	Contents
`Scripts/Runtime/`	MonoBehaviours that run in play mode (destruction logic, stress solver, pooling).
`Scripts/Editor/`	Custom UI (Structure Manager, scene tools, inspectors).
`Materials/`, `Textures/`, `Audio/`, `ParticleEffects/`	Included assets.
`Samples/Demo/Scene/`	A sample scene demonstrating a few structures along with a way to destroy them.

Installation

1. Download the `.unitypackage` from the Asset Store.
2. In Unity choose **Assets → Import Package → Custom Package…** and open the file.
3. Click **Import** in the package dialog.

4. *(For URP/HDRP projects)* Included sample materials use the Built-in render pipeline. To convert materials, use the render-pipeline material upgrade tool (e.g., Edit → Render Pipeline → … Upgrade Project Materials).
5. *(Optional)* Install Unity's AI Navigation package (`com.unity.ai.navigation`) via the Package Manager for navigation utilities.

Requirements & Compatibility

Category	Details
Unity Editor	**2022.3 LTS and 2021.3 LTS** (Unity 6 not currently supported)
Render Pipelines	Built-in, URP, and HDRP
Assembly Definitions	`Mayuns.DSB`, `Mayuns.DSB.Editor`, `Mayuns.DSB.Demo`, `Mayuns.DSB.Navigation`
Optional Dependencies	`com.unity.ai.navigation` (Unity AI Navigation) – required only for navigation utilities

---

Limitations

- Editor-only authoring. No in-game construction or live editing.
- Destruction effects and stress simulation run at runtime, but stock tools cannot spawn new beams or walls.
- No built-in support for multiplayer/networking frameworks.
- Structures must be scaled uniformly; non-uniform scaling across axes is unsupported and can cause detached pieces or members to spawn distorted.
- Generating new voxels (placing walls or members) and rebuilding or caching very large structures may display a progress bar or hitch the editor.
- WebGL not supported due to threading restrictions.

Quick Start

1. Open a new scene.

2. Launch **Tools → Destructible Structure Builder**.

3. Click **Create BuildSettings Asset** and save it anywhere under *Assets/*. This stores defaults like member length, materials, and mesh-cache options.

4. In **Global Settings**, click **Create Destruction Effects Manager** to add the manager to your scene. It spawns and manages debris, audio, and particle effects.

   Then **create and assign profiles**:

   - **EffectsLibrary** — per-event default effects (audio + particles).
     Create: **Assets → Create → DSB → Effects Library**
     Assign: **DestructionEffectsManager → Effects Profile**

   - **MaterialEffectsLibrary** — per-material overrides.
     Create: **Assets → Create → DSB → Material Effects Library**
     Assign: **DestructionEffectsManager → Material Overrides Profile**

Note: A ready-to-use EffectsLibrary with preset audio and particle values is included under `Samples/`. Assign this asset to start using the included effects immediately.

Important: If no **Effects Profile** is assigned (and no matching material override exists), no audio/particle effects will play.

5. At the top of the window, Set *Build Mode* to **Create Structure** and click once in the Scene view. A cube (first connection) will appear.
6. Switch to **Grid Member Build**. With gizmos enabled, hover the connection; cyan ghost beams show valid directions. Click a ghost to spawn a member and a new end-connection.
7. Switch to **Grounded Toggle**. Click members that physically touch the ground to mark them **green** (grounded). Use **Auto-Detect Grounded** in the *Structural Group Manager* to automatically mark members touching the layers defined in **Build Settings → Grounded Layers**.
8. Choose **Wall Build**. Hover a member to preview a magenta ghost wall; click and drag to adjust width, click again to start adjusting height, and finally click to create a wall with the **Default Wall Design** (or an empty grid if none). For stairs, switch to **Stair Build**: click once to set the start, move the mouse to set width, click to lock it, then move again to choose the end point and click a final time to place the stairs.
9. *(Optional)* Rotate the preview with the rotate button or the hot-keys below.
10. Select **Apply Design** and create a custom wall design with windows and shaped pieces. Click on any wall in the scene to apply the current design.
11. Press **Play**. Shoot or collide with the structure to see pieces detach, crumble, and spawn pooled debris.

Scene Setup Tips

- Use **Structure Manager → Runtime Debris Settings → Create DestructionEffectsManager** if the scene has none.
- Effects require a manager: Destruction and stress simulation still run without a DestructionEffectsManager, but debris/audio/particles won’t. Add one to enable effects.
- To allow for proper stress simulation in structures, ensure that members which are connected to the ground are marked as grounded.
- If a structure has no grounded members, damaging any member may cause the entire group to detach or collapse as a single mass.
- Use the "Show Stress Gizmos" toggle in a structures inspector to visualize load propagation through the structures members. Adjust the structural strength variable or use the per member capacity override (available through each member's inspector) to ensure that structures don't fail / collapse immediately upon interaction.
- Avoid non-uniform scaling on the structure or any parent transforms. Scaling axes differently causes pieces to spawn or detach distorted.

Performance & Best Practices

Area	Recommendation
Debris Pool	Balance *Max Active Debris* vs *Debris Lifetimes* – long lifetimes need bigger pools.
Wall & Member cell budget	Keep wall grids ≤ 10×10 and reduce member subdivision where possible.
Structure Piece Count Reduction	If you see hitches during large break events, reduce total members/walls/cells per structure.
Physics Layers	Exclude debris-to-debris collisions via the Layer Collision Matrix.

Mesh Cache Management

DSB persists generated meshes to disk so structures can be prefabricated and reopened quickly. When a structure is built, the system looks for meshes on disk before generating new ones. If a cached version exists, it is loaded and used immediately; otherwise the mesh is produced and written to the cache for the next session. This avoids rebuilding complex geometry every time the scene is opened.

How the cache is organized

- Choose a cache root in **Structure Manager → Build Settings → Mesh Cache → Change…**.
- Each **Structural Group Manager** writes its meshes to a subfolder under that root. The subfolder name is derived from the group's name, scene GUID, and instanceID, ensuring a unique folder per group to prevent collisions.
- Mesh files inside the group folder are stored as Unity `Mesh` assets named after the individual piece identifiers. Deleting a mesh file only affects that single piece.

Maintaining the cache

- Use **Validate Cache** in a group's inspector after moving the folder, enabling caching for the first time or changing the structure significantly. This option scans the group's folder and rebuilds any missing or out-of-date meshes.
- If cached meshes appear incorrect or pink, clear the affected group's cache folder to force a full rebuild on the next structure bake.
- Cache folders are not cleaned up automatically. Remove old group folders manually to reclaim disk space or to prevent the project from shipping unused meshes.

Common issues and fixes

- **Missing meshes after relocating the project** – The absolute cache path stored in the settings may no longer be valid. Reassign the root folder and run **Validate Cache**.
- **Renamed or deleted groups still consume disk space** – Manually delete their corresponding subfolders in the cache root.
- **Unexpected geometry after editing a structure** – Stale meshes may still be loaded. Delete the group's cache folder or use **Validate Cache** to regenerate.

Editor Workflow

Build Modes Toolbar

Mode	Description
Create Structure	Click anywhere in the Scene view to place a new root connection. All members and walls branch from this node.
Grid Member Build	Place members snapped to grid directions. Hover a connection or existing member to preview cyan beams, then click to add the member and its end connection.
Free Member Build	Place members at any angle without grid snapping. Click a start connection or member, position the other end with the mouse, and click again to confirm.
Grounded Toggle	Mark or unmark members as grounded. Grounded members are fixed to the world and pass support into the structure during stress simulation.
Wall Build	Attach walls to members. Hover a member to preview a magenta ghost wall, click and drag to set height, then release to place the wall.
Stair Build	Attach stairs to members. Hover a member to preview purple stairs, then click to place. Adjust alignment in the Inspector if needed.
Apply Design	Apply the active `WallDesign` to a wall. Select or create a design, then click an existing wall to update its grid layout.
Apply Material	Apply the currently selected material to structure parts. Click a connection, member, or wall to assign it instantly.
Apply Member Properties	Apply edited member settings. Adjust properties in the manager, then click a member to overwrite its current settings.
Delete	Delete a connection, member, or wall. Deleting a connection also removes any attached members or walls.

Build Overlay

Activating any build mode displays a small overlay in the Scene view. It offers **Rotate Design** ▯ and **Cancel** buttons and shows the state of snapping and axis locks. Use the overlay or hotkeys (*C* to toggle snap; arrow keys to lock X/Y/Z) while placing members, walls, or stairs.

Wall and stair placement expose two dedicated snapping toggles in **Structure Manager → Wall Build** and **Stair Build**:

- **Edge Alignment Snapping** pulls the wall preview toward the nearest edge on the host face. Disable it when you need free placement anywhere across the surface.

- **Endpoint & Midpoint Snapping** keeps the preview aligned with the host member's midpoint or endpoints along its length. Toggle it quickly from the Scene view with the *M* hotkey.

Wall Design Rotation

Control	Effect
+90 degrees	Rotate preview clockwise. Hot-key: **R**

Copy Wall Design From Scene

While in **Apply Design** or **Wall Build**, click **Copy Design From Wall in Scene** in the Structure Manager window. Then click an existing wall to load its layout into the active design or a `WallDesign` asset.

Saving a Wall Design to Disk

1. Enter **Apply Design** and make a new design or copy one from the scene.
2. Click **Select…** to choose a folder under *Assets/*.
3. Type a unique *Design Name* (indicator turns green when valid).
4. Press **Save Design** to create a `.asset` file storing cell types, rows, columns, and materials.

Wall Design Inspector

The **Wall Design Inspector** is a custom editor that appears when a `WallDesign` asset is selected. It shows a color-coded grid of cells and lets you modify layouts in real-time.

Features

Feature	Description
Interactive Grid	Shows each cell in a color-coded preview. Clicking or dragging edits live.
Hover Preview	Moving the cursor over cells reveals a transparent preview of the active brush.
Brush Palette	A legend below the grid lets you pick the current cell type to paint with.
Multi-cell Editing	Click and drag across multiple cells to apply the brush quickly.
Auto Grid Resize	Changing the rows or columns fields automatically adjusts the cell list.

How to Use

1. **Create a Wall Design** – **Assets → Create → DSB → Wall Design** or use **Apply Design** mode to create or open a `WallDesign` asset.
2. **Edit the Design** – Select the asset so the Inspector shows the editable cell grid, or open the asset in **Apply Design**.
3. **Paint Cell Types** – Activate a brush from the legend then click or drag in the grid. Hovering shows a semi-transparent preview.

4. **Use Triangle or Sloped Cells** – Pick the triangle or sloped brush and use the **Rotate** button to cycle orientations for angled pieces.
5. **Erase or Reset** – Select the **Empty** brush to clear a cell.
6. **Apply in Scene** – Use **Apply Design** from the Build Modes Toolbar to apply the saved layout to any wall.

Copy Member Settings From Scene

In the **Structure Manager** under structural-member settings, press **Copy Member From Scene**. Click a member in the scene to mirror its length, thickness, support capacity, material and division settings into the build settings.

Hotkeys

Hotkeys are provided for exiting the build mode easily, toggling snapping, and rotating walls before placement. Toggle them via **Structure Manager → Hotkeys** and configure bindings from **Edit → Shortcuts** (search for `DSB/Structure Build`).

Default Key	Action
`Esc`	Cancel current build mode or preview
`R`	Rotate preview 90 degrees in Wall, Stair, or Apply Design modes
`T`	Rotate the wall preview while placing walls
`C`	Toggle snap while placing members, walls, or stairs
`M`	Toggle midpoint snapping for members, walls, or stairs
`LeftArrow`	Toggle X-axis lock in Free Member or Stair Build
`DownArrow`	Toggle Y-axis lock in Free Member Build
`RightArrow`	Toggle Z-axis lock in Free Member or Stair Build

Prefab a Structure

1. Build your structure and select its **root object** in the Hierarchy (the parent with **Structural Group Manager**).
2. Optionally, in the Inspector's **Mesh Cache** section confirm a cache folder and use **Validate Cache** if the **Structural Group Manager** inspector warns of missing pieces.
3. Drag the root object from the Hierarchy into the Project window to create a prefab asset.

> ***Note:*** *If the prefab asset is deleted later and the instance appears red in the Hierarchy, build modes may not work correctly. Unpack the object or create a new prefab before editing.*

Destruction Effects Manager

The singleton **DestructionEffectsManager** owns all runtime destruction feedback: it prepares reusable debris shells, pools particle systems and audio sources, and reacts to the events raised through `DestructionSignals` (`PieceCrumble`, `MemberStress`, `LargeCollapse`, `WindowShatter`, `DebrisImpact`, and `GroupCrush`).

Create one automatically from
Structure Manager → Runtime Debris Settings → Create DestructionEffectsManager in Scene
or add the component manually. The script enforces a single instance at runtime by destroying duplicate components during `Awake()`.

Assign an **Effects Library** asset for the baseline particle/audio response and optionally a **Material Effects Library** so individual materials can override those defaults. When an event fires, the manager locates the best matching effect, plays pooled audio, spawns pooled particle prefabs at the event location, and schedules both for automatic cleanup. Without an assigned profile no effects will play, so plug in the provided sample library from `Samples/` or create your own.

Inspector breakdown

- **Audio & Particle Effects** – `Effects Profile` (global defaults), `Material Overrides Profile` (per-material overrides), plus pool sizes (`Particle Pool Size`, `Audio Pool Size`) that cap how many inactive systems are retained for reuse.
- **Debris Pooling** – `Max Active Debris` (simultaneous debris cap), `Current Active Debris` (read-only counter), `Max Pool Size` (pooled shell capacity), and `Initial Pool Size` (number of shells pre-generated during `Start()`).
- **Debris Layering & Shadows** – `Debris Layer` assigns a single layer to every debris shell. `Enable Debris Shadows` toggles whether pooled debris and persistent piles cast shadows.
- **Persistent Debris** – `Enable Persistent Debris` keeps resting debris in the scene. Anything that settles on the configured `Ground Layer` (Can be adjusted in settings section of main DSB window) is merged into combined meshes every few seconds so rubble piles persist without thousands of active rigidbodies.
- **Debris Colliders** – `Debris Collider Scale` adjusts the automatically generated box collider (default 0.5 = 50 % shrink) so you can trade precision for stability.
- **Debris Spawn Motion** – `Debris Explosiveness` and `Debris Spin` apply random linear and angular impulses when debris activate.
- **Debris Lifetimes (seconds)** – `Debris Lifetime` controls how long pooled debris remain active before they are returned to the pool. `Detached Group Lifetime` removes large detached structural groups after they have been spawned (these are not pooled).
- **Structural Group Collision Damage** – `Collision Damage Layers`, `Collision Impulse Threshold`, `Collision Impulse Sensitivity`, and `Collision Damage` configure how `ApplyGroupCollisionDamage()` converts collision impulses into damage when detached groups impact the world.
- **Collapse Thresholds** – `Large Collapse Mass Threshold` defines how much mass must detach before the "Large Collapse" effect and audio are allowed to trigger.
- **Editor Settings** – `Hide Pooled Debris In Hierarchy` hides inactive shells, while `Hide All Debris In Hierarchy` hides active debris as well to keep the hierarchy tidy.

Collision-based Damage

Detached structural groups can take damage when they collide with the world. Use the **Group Collision Damage** fields to configure this behaviour:

- **Collision Damage Layers** – layers capable of dealing impact damage.

- **Collision Impulse Threshold** – minimum collision impulse before damage is considered.
- **Collision Impulse Sensitivity** – multiplier applied to impulse over the threshold.
- **Collision Damage** – base damage applied per qualifying impact.

Debris Impact Sound Hook

Every pooled debris carries a **DebrisCollisionSound** component. When debris collides, it raises `DestructionSignals.DebrisImpact`, letting you play custom impact audio or particles by subscribing to the event.

Workflow Quick Tips

- **Balance pools and lifetimes** – Increase *Max Active Debris* and *Max Pool Size* when you want debris to linger, or shorten *Debris Lifetime* for faster cleanup.
- **Persistent piles** – Enable **Persistent Debris** and set **Persistent Ground Layer** to keep rubble piles around without maintaining active rigidbodies.
- **Collider tuning** – Lower **Debris Collider Scale** when debris jitter; raise it toward `1.0` to match the visual mesh more closely.
- **Manual spawning** – Custom scripts can call `GetReusableDebrisShell()` then `RegisterTimedDebris()` (or `RegisterDebris()`) to integrate bespoke debris flows.
- **Event hooks** – Subscribe to `DestructionSignals` (for example `DebrisImpact` or `LargeCollapse`) to layer additional VFX/SFX alongside the pooled defaults.
- **Collision damage** – Detached groups that collide with the layers listed in **Collision Damage Layers** can take damage scaled by **Collision Impulse Sensitivity**.

Structural Group Inspector

Selecting a **Structural Group Manager** exposes foldouts that mirror the custom inspector:

- **Overview** – Toggle `Show Stress Gizmos` to overlay stress colours in the Scene view while in Play Mode. A non-uniform scale warning appears automatically if the hierarchy uses mismatched axes.
- **HP** – `Wall Voxel Health`, `Window Voxel Health` and `Member Voxel Health` each include **Apply to All** buttons that push the entered value across the entire structure (Undo-supported, disabled while playing).
- **Simulation** – `Density` (with **Apply to All**) recalculates voxel masses, while `Structural Strength` and `Validation Interval` control the stress solver. Changes schedule an automatic stress recalculation.
- **Utilities** *(Editor only)* –
  - **Grounded Members** → `Auto-Detect Grounded` marks members touching the grounded layers configured in Global Settings.
  - **Navigation** → `Add NavMesh Rebaker` attaches `StructureNavmeshRebaker` (disabled if already present).
  - **Voxel Operations** → `Rebuild Voxels` regenerates meshes/colliders; `Validate Cache` repairs cached mesh references (available when mesh caching and the cache folder are configured). Warnings highlight missing caches or members needing rebuilds, and the current cache folder path is displayed.

    - Additional info messages appear if mesh caching is disabled or the global cache folder is invalid.
- **Diagnostics** – `Refresh Counts` updates readouts for wall, member, and combined totals to help verify structure contents.

Stress Visualizer

Enable **Show Stress Gizmos** in the inspector of any *Structural Group Manager* to color over-stressed beams while the game is running. (Editor only)

Piece-level Debug Inspectors

Selecting a **Member Piece** or **Wall Piece** shows:

- *Accumulated Damage* (editable at runtime).
- *Max/Current Health* readouts.

Navigation

DSB ships with optional helper that ties into Unity's ai.navigation package. Install `com.unity.ai.navigation` to keep pathfinding data aligned with collapsing structures.

StructureNavmeshRebaker

Attach this component to a **Structural Group Manager** to automatically refresh a `NavMeshSurface` whenever pieces crumble or large collapses occur.

- Auto-adds/configures a `NavMeshSurface` to collect child physics colliders.
- Throttles rebakes via **debounceSeconds** and **minDelaySeconds**.
- Validates colliders on startup, warning about meshes that cannot be baked.

Texture Scaling

Every wall and structural member generated by DSB stores two numbers — **Texture Scale X** and **Texture Scale Y**. Each value represents the real-world meters covered by one repeat of the material along the object's local X or Y axis. During the build process DSB divides the wall or member's world size by these scales to derive a tiling factor, then multiplies all mesh UVs by that factor so textures repeat consistently regardless of object dimensions. A scale of `1` means one meter per texture tile; `0.5` doubles the tiling (two repeats per meter); `2` halves it. Setting a scale to `0` falls back to a one-meter repeat. Global defaults live in **Structure Build Settings** (*Member Texture Scale X/Y* and *Wall Texture Scale X/Y*), while individual walls and members can override their own values — rebuild after changing them to see the effect.

Damage & Destructible Workflow

Objects that splinter inherit from **`Destructible`** and implement **`IDamageable`** via concrete voxel scripts.

Script / Type	Responsibilities
`IDamageable`	Single-method contract: `void TakeDamage(float damage)` – opt-in to DSB’s

	damage pipeline.
Destructible (abstract)	1. Stores an array of **DebrisData** (pre-cut meshes + materials) via `CreateAndStoreDebrisData()`. 2. Finds scene **DestructionEffectsManager** in `Awake()`. 3. On `Crumble()` spawns pooled debris (throttled), then destroys itself.

> ***Heads-up:*** *If no **DestructionEffectsManager** exists, `Crumble()` still destroys the object, but no debris spawns.*

How to Apply Damage

Pieces like `Chunk` already implement `IDamageable`. Call `TakeDamage(amount)` from your game scripts whenever a raycast, trigger or explosion hits a structure. Damage accumulates per voxel and eventually leads to splitting or a call to `Crumble()`.

```
if (hit.collider.TryGetComponent<IDamageable>(out var dmg))
    dmg.TakeDamage(25f);
```

Extensibility Guide

DestructionEffectsManager exposes lightweight hooks so you can add gameplay logic and VFX/SFX without editing core tools. Most integrations fall into two buckets:

- **Event-driven logic** — Subscribe to `DestructionSignals` events to react to destruction moments (voxel crumbles, stress spikes, collapses, impacts). All events share the same signature: **`(StructuralGroupManager group, Material material, Vector3 position)`**, enabling structure-aware, material-aware and location-based effects.
- **Content & tuning** — Drive effects via the **Material Effects** array in the inspector (per-material particle/audio overrides), and tune behaviour at runtime by adjusting `maxActiveDebris`, `maxPoolSize`, the debris debris lifetime, and the detached group lifetime. Changes apply instantly in Play Mode.

Events — Quick Reference

Event name	Parameters	Trigger
DestructionSignals.PieceCrumble	(StructuralGroupManager group, Material material, Vector3 position)	A structural piece (member, wall, or connection) crumbles.
DestructionSignals.MemberStress	(StructuralGroupManager group, Material material, Vector3 position)	The stress solver flags a beam as overstressed.
DestructionSignals.LargeCollapse	(StructuralGroupManager group, Material material, Vector3 position)	A detached group of more than four members collapses.
DestructionSignals.WindowShatter	(StructuralGroupManager group, Material material, Vector3 position)	A window panel is destroyed and shatters.

DestructionSignals.DebrisImpact	(StructuralGroupManager group, Material material, Vector3 position)	A flying debris collides with the world.
DestructionSignals.GroupCrush	(StructuralGroupManager group, Material material, Vector3 position)	A detached group slams into collision layers above the force threshold.

Runtime Scripting API

```csharp
using Mayuns.DSB;
using UnityEngine;

/// <summary>
/// Example of wiring custom logic into every stage of destruction.
/// </summary>
public class DsbEventBridge : MonoBehaviour
{
    void OnEnable()
    {
        // voxel-level crumble
        DestructionSignals.PieceCrumble += OnCrumble;

        // individual beam stress
        DestructionSignals.MemberStress += OnMemberStress;

        // wholesale collapse
        DestructionSignals.LargeCollapse += OnCollapse;

        // glass shatters
        DestructionSignals.WindowShatter += PlayGlassTinkle;

        // debris hits anything with a collider
        DestructionSignals.DebrisImpact += SpawnDustPuff;

        // detached groups crunch on the ground
        DestructionSignals.GroupCrush += OnGroupCrush;
    }

    void OnDisable()
    {
        DestructionSignals.PieceCrumble -= OnCrumble;
        DestructionSignals.MemberStress -= OnMemberStress;
        DestructionSignals.LargeCollapse -= OnCollapse;
        DestructionSignals.WindowShatter -= PlayGlassTinkle;
        DestructionSignals.DebrisImpact -= SpawnDustPuff;
        DestructionSignals.GroupCrush -= OnGroupCrush;
    }

    /* ----------------- event handlers ----------------- */
    void OnCrumble(StructuralGroupManager group, Material m, Vector3 pos)
    {
        Debug.Log($"Chunk of {m.name} crumbled at {pos}");
    }

    void OnMemberStress(StructuralGroupManager group, Material m, Vector3 pos)
    {
        // e.g. flash a warning UI
    }

    void OnCollapse(StructuralGroupManager group, Material m, Vector3 pos)
    {
        // big boom – spawn camera shake, etc.
    }

    void PlayGlassTinkle(StructuralGroupManager group, Material m, Vector3 pos)
    {
        // trigger bonus shards
    }

    void SpawnDustPuff(StructuralGroupManager group, Material m, Vector3 pos)
    {
        // lighter particle on every debris impact
    }

    void OnGroupCrush(StructuralGroupManager group, Material m, Vector3 pos)
    {
        // group hits ground – maybe spawn dust
    }
}
```

Advanced Topics

See Technical Internals for an in-depth look at the Wall Manager, Structural Member, and Stress Solver systems.

Troubleshooting

This section covers common issues you may encounter when working with **Destructible Structure Builder (DSB)**. For more details, see the linked sections of the manual.

Editor & Build Tools

Q: Why are the build tools not working?
A: Ensure gizmo visibility is enabled.
(See Editor Workflow)

Q: Diagonal or orthogonal ghost beams never appear.
*A: In **Build Settings → Hide Axis** set **None** or **Orthogonal**.*
(See Editor Workflow → Grid Member Build)

Prefabs & Mesh Cache

Q: Why is my structure invisible or missing pieces when I place it in a prefab?
*A: Select the root (parent) of the structure, revealing the **Structural Group Manager** in the inspector, and verify cache folder path. Then click **Validate Cache**.*
(See Mesh Cache Management)

Q: Detached pieces look stretched or members spawn distorted.
A: Non-uniform scaling is likely applied somewhere in the hierarchy. Use uniform scale (same value on all axes) on the structure and its parents to prevent distortion.
(See Limitations)

Materials & Visuals

Q: Why are my walls pink?
*A: Assign a material in **Build Settings** → **Wall Material** or use **Apply Material** mode.*
(See Editor Workflow → Apply Material)

Q: Stair steps are rotated improperly.
A: Step orientation is determined relative to the structure's orientation, not global orientation. Ensure the parent structure has a uniform and non-rotated transform.
(See Editor Workflow → Stair Build)

Runtime Effects & Navigation

Q: How do I use the default effects without creating my own Effects Library asset?
*A: The included `Samples/` folder contains a pre-made Effects Library with audio and particle presets. Assign it in the **DestructionEffectsManager** → **Effects Profile** inspector.*
(See Quick Start and Destruction Effects Manager)

Q: Pieces, debris, and detached structural groups disappear too quickly.
*A: Increase *Max Active Debris* or adjust debris lifetime in **DestructionEffectsManager**.* (See Destruction Effects Manager → Workflow Quick Tips)

Q: NavMesh doesn't rebake on collapse.
*A: Install `com.unity.ai.navigation` and add **StructureNavmeshRebaker** to your structure's root.*
(See Navigation)

Support

Email: **support@mayuns.com** *Discord:* https://discord.gg/73GaMeP6JF *Website:* https://mayuns.com

---

License