

REDICION STUDIO · CORE ENVIRONMENT FAMILY



GrassCore

Stylized Grass & Vegetation System

USER DOCUMENTATION

Version 1.17.0 · Unity 6 · Universal Render Pipeline
Redicion Studio · Contact@RedicionStudio.com



Contents

Chapters are grouped into six parts. Every chapter follows the same structure: *What it does* · *How to set it up* · *Important Settings* · *Example Workflow* · *Performance Notes* · *Common Problems*.

PART 1 · GETTING STARTED	4
01 Introduction to GrassCore	5
02 Requirements & Installation	8
03 Quick Start – Your First Meadow	11
04 The GrassCore World	13
05 Surfaces – Terrains, Meshes and Blockers	17
PART 2 · GRASS & FLOWERS	19
06 Grass Species Profiles	20
07 Custom Grass Meshes & LOD0 / LOD1 / LOD2	23
08 Flowers & Flower Species	26
09 Grass Vegetation Areas & Local Overrides	28
10 Spline Areas & Mask Volumes	31
11 Paths, Grass Removal & Path Falloff	33
12 Terrain Layer Painting & Path Decals	35
PART 3 · WIND, INTERACTION & EFFECTS	37
13 Wind & Wind Interaction	38
14 Player & Object Interaction	40
15 Explosion Shockwaves	43
16 Helicopter & Propeller Downwash	45
17 Mowing, Cut Height & Grass Clippings	47
18 Fire, Burning & Burned Grass	50
19 Seasons & Snow Accumulation	52
20 Dynamic Color Zones	54
21 Bioluminescent & Magic Grass	56
PART 4 · WATER	58
22 Water Volumes, Lakes & Rivers	59
23 Underwater Vegetation & River Current	62
24 Interactive Water – Ripples, Wakes & V-Waves	65
PART 5 · TREES & BUSHES	67
25 Trees & Bushes – Overview	68
26 Procedural Trees & Bushes	70
27 Custom Tree & Bush Meshes	73
28 Tree & Bush LODs and Billboards	75
29 Manual Placement & Vegetation Areas	77
30 Tree & Bush Wind, Flowers & Fruits	80
31 Tree & Bush Interaction and Optional Colliders	82
32 Destroyable Trees, Tree Damage & Falling Trees	84

PART 6 · PERFORMANCE & REFERENCE

87

33

Performance & Optimization

88

34

Debugging & Common Problems

91

35

Frequently Asked Questions

93

36

Runtime API – Short Examples

95

37

Best Practices

99

38

Reference & Support

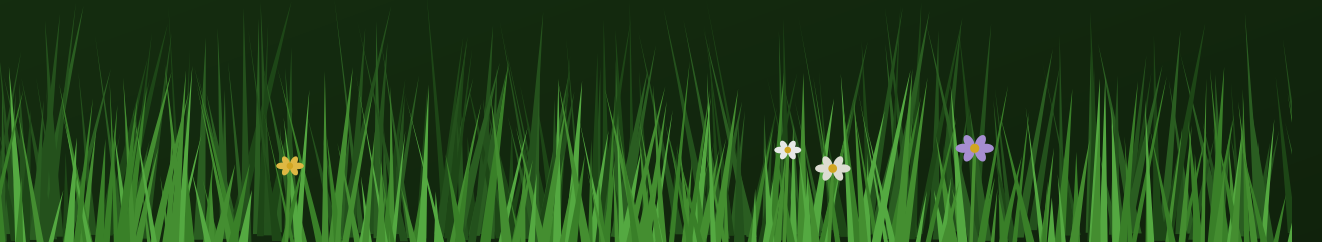
101

PART 1

Getting Started

Install GrassCore, create your first meadow in minutes and learn the basic building blocks: the GrassCore World, surfaces and quality settings.

01	Introduction to GrassCore	5
02	Requirements & Installation	8
03	Quick Start – Your First Meadow	11
04	The GrassCore World	13
05	Surfaces – Terrains, Meshes and Blockers	17



01

OVERVIEW

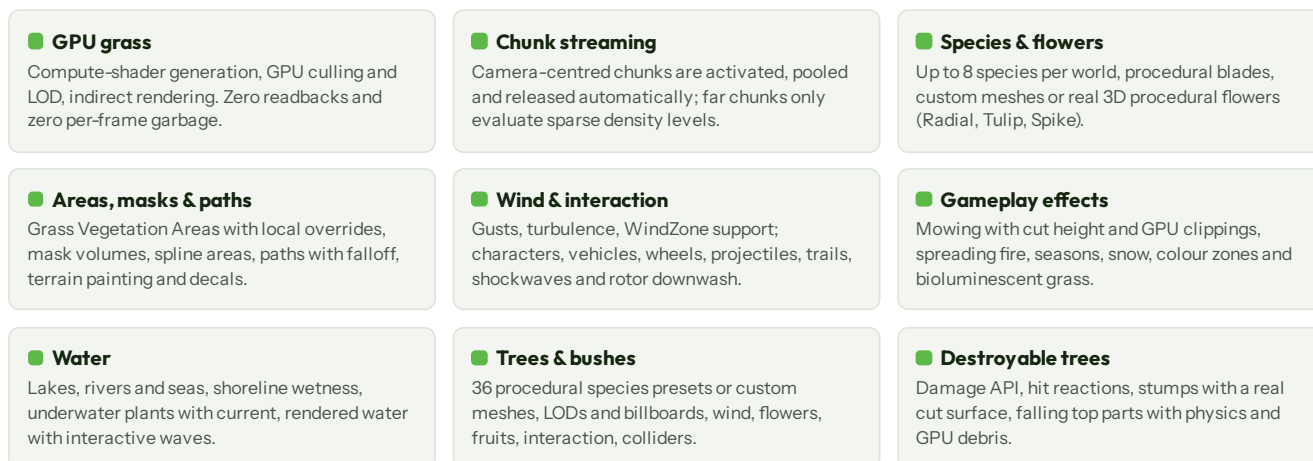
Introduction to GrassCore

GrassCore renders dense, animated and interactive grass, flowers, trees and bushes over very large worlds – generated on the GPU, with no baking and no painted instance data. Add a world, pick a surface, choose species, and grass appears.

What it does

GrassCore is a complete vegetation system for **Unity 6** and the **Universal Render Pipeline (URP)**. All grass is generated procedurally on the GPU around the camera: compute shaders decide where blades grow, cull them and sort them into level-of-detail buckets, and indirect draw calls render the result. Nothing is stored per blade and there are no GameObjects per blade, so meadows of millions of blades cost the same amount of memory as an empty field.

On top of the grass renderer GrassCore offers a set of optional systems. Each of them is switched on only when you use it – a scene without water, trees or mowers does not pay for them.



How GrassCore works

Understanding the basic pipeline helps you to choose good settings. Every frame the CPU only performs cheap bookkeeping: it moves the chunk grid with the camera and builds the cached *tiles* (height, normal and species density of a chunk) for chunks that have just become active or whose surface changed. Everything that scales with the number of blades runs on the GPU, once per camera.

ONCE PER FRAME (CPU, budgeted)



PER CAMERA (GPU, no readbacks)



Chunks outside the view never dispatch work. Far chunks only enumerate the sparse levels of the lattice, so the candidate count follows what is visible – not the view distance.

Figure 1.1 – The GrassCore frame: cheap CPU bookkeeping, then GPU generation, culling, LOD bucketing and one indirect draw per species and LOD.

The key idea is the **nested density lattice**. Every possible blade position belongs to a density level; each level halves the density of the previous one. Close to the camera all levels are used, further away only the sparse levels remain, and a chunk far away only enumerates the few positions that can survive. Positions never move when the level changes, so there is no popping – blades simply fade in and out.

Package contents

Folder	Contents
Runtime	The GrassCore runtime: world, chunk streaming, rendering, species, surfaces, masks, wind, interaction, fire, mowing, seasons, water and vegetation systems, shaders and compute shaders.
Editor	Inspectors, Setup Wizard, Scene view tools and gizmos, path painter, profiler window, menu items. Editor-only code – nothing of it ends up in a build.
Samples/Species	Ready-made grass and flower species with materials (meadow, tall grass, dry steppe, shore reeds, wildflowers, glowgrass and six procedural flower species).
Samples/Vegetation	38 tree and bush profiles, including <i>Oak (Destroyable)</i> and <i>Spruce (Destroyable)</i> .
Samples/WaterPlants	Seagrass, Water Weed, Kelp, Reed and Lily Pad.
Samples/Demo	Demo scene builder, demo player / vehicle / axe and the Canvas demo UI.
Documentation	This manual, a Quick Start, the API reference and the architecture notes.

How to set it up

The fastest way into GrassCore is the **Setup Wizard**. It opens once per Unity editor session and is always available under **Tools > GrassCore > Setup Wizard**. The *Getting Started* page shows a checklist with one-click actions; nothing in your scene changes unless you press a button.

- 1 Make sure your project uses URP (see [chapter 2](#)).
- 2 Open the Setup Wizard and follow the *Getting Started* checklist, or use the *Quick Setup* page to create a fully configured world in one step.
- 3 Press **Create Demo Scene** to see every feature in action.

The Quick Start in [chapter 3](#) walks you through the same steps in detail.

Important Settings

GrassCore follows a few principles that you will meet throughout this manual:

Principle	What it means for you
One world per scene	A single GrassCore World component renders all grass, trees and effects of a scene. Only one world renders at a time.
Profiles are assets	Grass species, tree / bush profiles and water plants are ScriptableObject assets that you can share between scenes.
Zero cost when unused	Optional systems only allocate GPU memory and run work while you use them (see the table below).
Quality presets	Low, Medium, High, Ultra or Custom control density, distance, LOD, shadows and budgets in one place.
Undo everywhere	Every editor action and every menu command is undoable.

● Grass Vegetation Areas	none in the scene	→ no shader variant, no extra tile data
● Mowing	no mower / no mown grass	→ no cut maps, no clippings buffer
● Underwater vegetation	no volume with plants	→ no buffers, dispatches or draws
● Water surfaces	no surface enabled	→ no meshes, simulation or draws
● Trees & bushes	no instances	→ no buffers, culling or draws
● Destroyable trees	until the first hit	→ no per-tree scripts or updates

Figure 1.2 – Optional systems and when they cost nothing.

Example Workflow

A typical project uses GrassCore like this:

- 1 Create the world and let it detect your terrains automatically.
- 2 Pick two or three species from `Samples/Species`, or create your own.
- 3 Shape the result with filters (slope, elevation, terrain layers), Grass Vegetation Areas, mask volumes and paths.
- 4 Add interactors to the player and vehicles; add effects such as fire, mowing or seasons where your game needs them.
- 5 Choose a quality preset per platform and check the result in the Profiler window.

Performance Notes

GrassCore is designed so that cost follows what you see, not the size of your world. At the same view distance, a huge terrain renders at roughly the same GPU cost as a small one. The most effective controls are the quality preset, **Halving Distance**, the render distance and shadows – see [chapter 33](#).

Common Problems

Problem	Solution
Nothing renders after adding the world	Check the warnings at the top of the GrassCore World inspector: URP active, compute shaders supported, a species assigned, a surface found.
Unsure where to start	Run Tools > GrassCore > Create Demo Scene and press Play. The demo shows every system with its default settings.

02

GETTING STARTED

Requirements & Installation

What your project needs to run GrassCore, how to install the package and how to verify that everything is ready – in five minutes.

What it does

This chapter lists the supported Unity versions, render pipelines and graphics APIs, and explains how to import GrassCore and check the project with the built-in tools.

Supported versions

Requirement	Supported
Unity	Unity 6 (6000.0 or newer)
Render pipeline	Universal Render Pipeline 17+ (the URP version that ships with Unity 6). The Built-in Render Pipeline and HDRP are not supported.
Rendering paths	Forward, Forward+ and Deferred. In Deferred the grass is drawn in a forward-only pass. SSAO and fog are supported.
Graphics APIs	Any API with compute shader support: DirectX 11, DirectX 12, Vulkan and Metal.
Cameras	Multiple cameras, Scene view cameras and (optionally) reflection cameras.
Packages for the samples	The demo works with the legacy Input Manager or the Input System package. The Canvas demo UI needs the Unity UI package (<code>com.unity.ugui</code>).

COMPUTE SHADERS ARE REQUIRED

GrassCore generates grass with compute shaders. On devices without compute shader support the GrassCore World reports *Unsupported* in its inspector and does not render. The Runtime API offers `GrassCoreRuntimeAPI.IsAvailable` to check this at runtime.

How to set it up

- 1 Import GrassCore from the Unity Package Manager (*My Assets*). Keep the default folder `Assets/GrassCore`.
- 2 Make sure a **URP asset** is assigned in **Edit > Project Settings > Graphics** (and per quality level in *Quality*).
- 3 Open **Tools > GrassCore > Setup Wizard**. On the *Getting Started* page step **1Project Check** must show three green dots: URP active, compute shaders supported, GrassCore resources found.
- 4 If URP is not active, the wizard offers **Open Graphics Settings**.

The Setup Wizard

The wizard opens automatically **once per Unity editor session** – not after script or domain reloads – and can be turned off with **Show at startup** at the bottom of the window. It has three pages:

- **Getting Started** – a checklist (Project Check, GrassCore World, Grass Surfaces, Species, Interaction) with status dots and buttons that call the normal GrassCore menu commands. Below the checklist, *More Tools* creates areas, paths, masks, lakes, trees and colour zones, and *Explore & Learn* opens **this documentation** (*Open Documentation*, *Open Documentation Folder*), the demo scene, the profiler and the Quick Start guide.
- **Quick Setup** – choose terrains, mesh surfaces, species, a quality preset, an interactor and the statistics overlay, then press **Create GrassCore** (or **Update GrassCore** when the scene already has a world). The whole setup is a single undo step.
- **Support** – copyable support e-mail, Discord link, *Open Documentation* / *Open Documentation Folder*, links to the Quick Start, API and architecture notes and a *Copy System Info* button for bug reports.

The wizard never changes your scene or project settings by itself – only when you press one of its buttons.

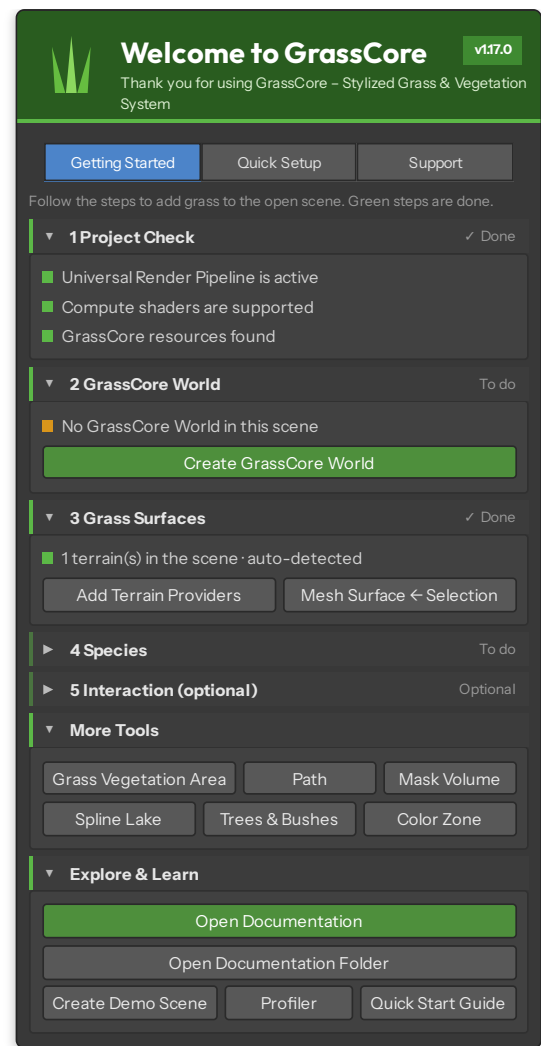


Figure 2.1 – Setup Wizard, Getting Started page.

Important Settings

Setting	Where	Notes
URP asset	Project Settings › Graphics / Quality	Required. GrassCore warns in the World inspector when URP is not active.
Depth Texture & Opaque Texture	URP asset	Only needed for water surfaces (refraction, depth colour, soft shores). The Water Volume inspector has a button that enables both.
Show at startup	Setup Wizard footer	Turns the once-per-session wizard on or off.
Input handling	Project Settings › Player	Only relevant for the demo scripts: both the legacy Input Manager and the Input System are supported.

Example Workflow

- 1 Import the package into a URP project.
- 2 Let the wizard open (or open it from the Tools menu) and check the Project Check step.
- 3 Press **Create Demo Scene** on the *Explore & Learn* section, save your current scene when asked, and press Play.
- 4 Return to your own scene and continue with the Quick Start in [chapter 3](#).

Performance Notes

Installation itself has no runtime cost. The `Editor` folder (inspectors, wizard, tools) is compiled into an editor-only assembly and never ends up in a player build. Runtime resources (shaders and compute shaders) are loaded from

`Runtime/Resources/GrassCore`.

Common Problems

Problem	Solution
The active render pipeline is not URP	Assign a URP asset in Graphics and Quality settings.
World badge shows UNSUPPORTED	The current graphics API or device has no compute shader support, or GrassCore resources are missing. Reimport the package.
Water surface without refraction or depth colour	Enable <i>Depth Texture</i> and <i>Opaque Texture</i> on the URP asset (button Enable Depth & Opaque Texture on the URP Asset in the Water Volume inspector); the Depth Texture Mode must be <i>After Opaques</i> .
Demo controls do nothing	Check <i>Active Input Handling</i> in the Player settings; the demo supports both input systems.
Open Documentation reports that the PDF was not found	<code>Documentation/GrassCore_Documentation.pdf</code> was deleted or not imported – re-import GrassCore from the Package Manager (<i>My Assets</i>).

03

GETTING STARTED

Quick Start – Your First Meadow

From an empty URP scene to a waving, interactive meadow in a few minutes – using only menu commands and default settings.

What it does

This chapter creates a complete, working setup: a GrassCore World on a terrain, a few species, a character that bends the grass and a quality preset. Every step uses a menu command that you can undo.

How to set it up

- 1 Open a scene that uses URP and contains a **Unity Terrain** (any ground mesh works too – see [chapter 5](#)).
- 2 Choose **GameObject > GrassCore > GrassCore World**. The world is created with the *Default Meadow Grass* species.
- 3 Look at the Scene view: terrains are detected automatically (*Surfaces* tab → **Auto-detect Terrains**), and grass appears around the Scene view camera.
- 4 Select the world and open the **Quality** tab. Pick a preset and adjust **Blades / m²** in *Density & Streaming*.
- 5 Open the **Species** tab, click **+** > *Empty Slot* and drag more species into the new slots – for example *Tall Grass* and *Wildflowers* from *Samples/Species*.
- 6 Select your player and choose **Tools > GrassCore > Add Interactor to Selection**. Press Play and walk through the grass.

PREVIEW WIND IN EDIT MODE

Wind is animated in the Scene view as well. Enable **Always Refresh** in the Scene view's effects menu (the sun icon drop-down in the Scene view toolbar) to see it move without pressing Play.

Alternative: Quick Setup

Open **Tools > GrassCore > Setup Wizard**, switch to **Quick Setup** and configure everything on one page:

Section	What you choose
1 Project	Status of URP, compute shaders and resources.
2 Surfaces	Which terrains get grass (all ticked = auto-detection), and optional mesh surfaces (<i>Add Mesh Slot</i> , <i>Add Selected Objects</i>).
3 Species	The species list of the world (<i>Add Slot</i> , <i>Create New Species...</i>). Starts with the existing world's species or the default meadow grass.
4 Quality	The quality preset with a short summary of render distance, full density distance, shadows and blade budget.
5 Extras	<i>Interactor on</i> (adds a Character interactor, suggested: the main camera) and <i>Statistics overlay</i> .

Press **Create GrassCore** (or **Update GrassCore**). The wizard records everything as one undo step and selects the world.

Important Settings

The defaults are chosen to look good immediately. These are the first values worth changing:

Setting	Location	Effect
Quality	World > Quality	Overall budget: Low, Medium, High (default), Ultra or Custom.
Blades / m²	World > Quality > Density & Streaming	Base density (default 48) before species density and quality scaling.
Species list	World > Species	Up to 8 species; each blade picks one of them.
Density (species)	Species asset > Density & LOD	Share of the density lattice used by the species (0–1).
Height / Width	Species asset > Size & Shape	Blade height and width ranges in metres.

Example Workflow

Try the demo scene

Tools › GrassCore › Create Demo Scene builds a complete demo in a new scene (it asks to save your current scene first): a generated terrain with layers, a lake, a dirt path, a mesh plateau, rocks as blockers, several species, Grass Vegetation Areas, a walkable player, a vehicle that leaves tracks, masks and the Canvas demo UI. Press Play:

Key	Action	Key	Action
W A S D	Move	E (hold)	Cut grass
Shift	Sprint	R (hold)	Restore grass
RMB	Look around	T	Next season
LMB	Chop a destroyable tree	Z	Colour zone (cycles presets)
Space	Shockwave	X	Clear colour zones
B	Fire	G	Debug view
V (hold)	Rotor downwash	1 – 4	Quality Low ... Ultra
F	Wind gust	F9	Statistics panel
N	Glow aura	H	Hide / show the UI

The **Canvas demo UI** shows the controls (bottom left), *Live Settings* with clickable quality, season, debug view and statistics buttons (top right), a *Statistics* panel (top left) and a chop prompt with the tree's health. Add it to any scene with

Tools › GrassCore › Create Demo UI.

Performance Notes

The default **High** preset targets current desktop GPUs. For laptops and consoles start with **Medium**, for low-end hardware with **Low**. Always check performance in a build or in the Game view with the *Statistics* panel or **Tools › GrassCore › Profiler** – the Scene view also renders grass and adds to the cost while it is visible.

Common Problems

Problem	Solution
Grass only appears around the Scene view camera	Correct: in edit mode streaming follows the Scene view camera, in Play mode the main camera (or the <i>Streaming Anchor</i>).
Grass does not move in the Scene view	Enable <i>Always Refresh</i> in the Scene view.
No grass on my ground mesh	Meshes are not detected automatically. Select them and use Tools › GrassCore › Add Mesh Surface to Selection .
Character does not bend the grass	Add a <i>GrassCore Interactor</i> ; interaction runs in Play mode (or with <i>Simulate In Edit Mode</i> on the world).

04

BASIC SETUP

The GrassCore World

One component renders everything. The GrassCore World owns the species list, the quality settings, streaming, rendering, wind, interaction and all optional systems of a scene.

What it does

The **GrassCore World** is the heart of every GrassCore scene. It streams a grid of chunks around the camera, builds the cached surface tiles, runs the GPU generation and draws the grass for every camera. All global settings live on this component; its inspector is organised in eight tabs.

Tab	Contents
Species	The world's species list (up to 8, the index is the mask bit), the Grass Vegetation Areas of the scene and an inline editor for the selected species.
Surfaces	<i>Auto-detect Terrains</i> , the registered surfaces, quick actions for surfaces and volumes, shoreline wetness and runtime edits, underwater vegetation and water surfaces.
Quality	Quality preset, density, streaming, rendering options and the resolved live values.
Wind	Global wind, WindZone and external wind sources, current wind state.
Season	Season, season looks, snow, snow interaction and colour zones.
Interaction	Bending & trails, live counters, bioluminescent glow, fire and mowing.
Trees	Tree and bush modules: rendering, shared systems, features, destroyable trees, placement and status.
Debug	Debug views, scene gizmos and live statistics.

The header shows the state of the world: **RUNNING** (green) when it renders, **INACTIVE** when disabled or not the active world, **NOURP** or **UNSUPPORTED** (red) when the project or device cannot run GrassCore. Validation messages appear directly below the header – for example when no species is enabled, no surface was found, the main camera's far clip plane is shorter than the grass render distance, or blades are dropped because the instance budget is full.

How to set it up

- 1 Create the world with **GameObject > GrassCore > GrassCore World** (or **Tools > GrassCore > Create GrassCore World**). If the scene already has a world, the command selects it instead of creating a second one.
- 2 In **Species** add or replace species (see [chapter 6](#)).
- 3 In **Surfaces** keep *Auto-detect Terrains* on, or add surface providers (see [chapter 5](#)).
- 4 In **Quality** choose a preset. Choose **Custom** – or press *Customise This Preset* – to edit every value.
- 5 Use the **Debug** tab and the Scene view overlay to verify chunks, LOD rings and the blade count.

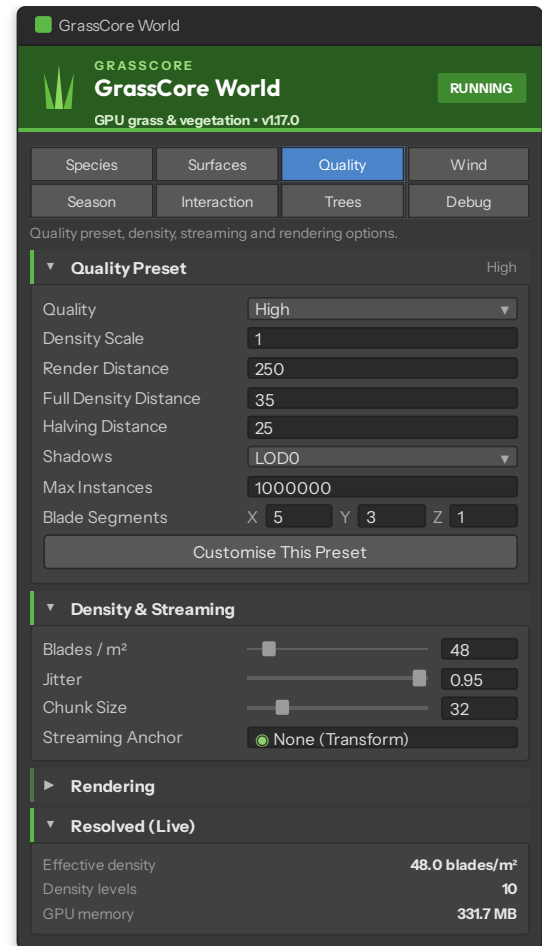


Figure 4.1 – GrassCore World inspector, Quality tab.

Important Settings

Quality presets

Value	Low	Medium	High	Ultra
Density Scale	0.5	0.75	1	15
Render Distance (m)	120	180	250	350
Full Density Distance (m)	15	25	35	50
Halving Distance (m)	14	20	25	35
Fade Width (m)	6	8	10	12
LOD Bias	0.6	0.8	1	13
Shadows	Off	LOD0	LOD0	LOD0 + LOD1
Shadow Distance (m)	0	25	40	60
Interaction Range (m)	48	64	96	128
Interaction Resolution	256	256	512	1024
Tile Resolution	32	64	64	64
Max Instances	300 000	600 000	1 000 000	2 000 000
Blade Segments (LOD0/1/2)	3 / 2 / 1	4 / 2 / 1	5 / 3 / 1	6 / 3 / 2
Tile Builds Per Frame	4	6	8	12

The values of a **Custom** profile have the same meaning:

Setting	Description
Density Scale	Multiplier for the world density (blades per m ²).
Render Distance	Maximum distance (m) at which grass is rendered.
Full Density Distance	Distance (m) up to which grass is rendered at full density.
Halving Distance	Every additional <i>Halving Distance</i> the density is halved. Larger values give a denser far field.
Fade Width	Width (m) of the distance fade of each density level – prevents popping.
LOD Bias	Multiplier for all species LOD distances.
Shadows / Shadow Distance	Which LODs cast shadows; instances closer than <i>Shadow Distance</i> are kept with a margin so off-screen blades still cast shadows.
Interaction Range / Resolution	World size (m) and texture resolution of the camera-centred interaction area.
Tile Resolution	Resolution of the cached surface tiles per chunk (height, normal, species density).
Max Instances	Maximum number of blades per camera. Excess blades are dropped and reported in the statistics.
Blade Segments	Geometry complexity of procedural blades for LOD0 / LOD1 / LOD2.
Tile Builds Per Frame	Streaming budget: chunk tiles built per frame.

Density, streaming and rendering

Setting	Default	Description
Blades / m²	48	Blades per square metre at full density, before species density and quality scaling.
Jitter	0.95	Random offset inside each lattice cell (0 = regular grid, 1 = fully jittered).
Chunk Size	32 m	Size of a streaming chunk. Larger chunks reduce CPU work, smaller chunks cull more precisely.
Streaming Anchor	–	Optional transform used as streaming centre. Default: main camera (Play mode) or Scene view camera (edit mode).
Render In Game Cameras / Scene View / Reflections	on / on / off	Which cameras render grass.
Layer	0	Unity layer of the grass draws – camera culling masks respect it.
Rendering Layers	1	Rendering layer mask (light layers, decals).
Receive Shadows	on	Grass receives shadows from other objects.

Example Workflow

Setting up quality per platform

- 1 Start with **High** and look at the *Resolved (Live)* section of the Quality tab: effective density, lattice per chunk, number of density levels, distance where density reaches zero, tile slots and GPU memory.
- 2 Press **Customise This Preset**. The current preset values are copied into an editable *Custom* profile.
- 3 Lower *Halving Distance* first if the far field is too expensive, then *Render Distance*, then *Density Scale*.
- 4 At runtime switch presets with `GrassCoreRuntimeAPI.SetQuality(GrassCoreQualityLevel.Medium)` – for example from your game's graphics menu.

Performance Notes

- The quality preset is the single most important performance control. Each preset step roughly doubles the blade budget.
- *Shadows* are expensive for dense grass. *LOD0* only is usually the best compromise; *Off* for low-end hardware.
- *Tile Builds Per Frame* limits CPU and GPU spikes when many chunks stream in at once, eg. after a teleport.
- Changing the preset or the tile resolution at runtime rebuilds all tiles once.

Common Problems

Problem	Solution
<i>Another GrassCoreWorld is active</i>	Only one world renders at a time. Remove or disable the second world.
<i>Blades were dropped because the instance budget is full</i>	Increase <i>Max Instances</i> (Custom profile) or reduce density / render distance.
<i>All chunk tile slots are in use</i>	Increase the chunk size or reduce the render distance.
<i>Grass ends before the horizon</i>	Check the camera's far clip plane, the species <i>Max Distance</i> and the quality <i>Render Distance</i> .
<i>Grass invisible in one camera</i>	Check the camera's culling mask against the world's <i>Layer</i> , and <i>Render In Game Cameras</i> .

05

BASIC SETUP

Surfaces – Terrains, Meshes and Blockers

Surfaces tell GrassCore where grass may grow. Unity Terrains are detected automatically; any mesh can become a grass surface or a blocker that removes grass beneath it.

What it does

GrassCore reads surfaces once per chunk, when the chunk streams in or when the surface changes – never per frame. Three kinds of surface exist:

Kind	Component	Use
Terrain	Terrain Surface Provider (or automatic)	Unity Terrains: heights, normals, holes and terrain layers.
Mesh surface	Mesh Surface Provider , Mode <i>Surface</i>	Any ground mesh: hills, plateaus, roofs, floating islands. The top-most surface of the meshes is used.
Blocker	Mesh Surface Provider , Mode <i>Blocker</i>	Rocks, houses, roads and props: grass underneath is removed.

No read/write meshes are required – mesh providers render their renderers once from above into a small target.

How to set it up

Terrains

- 1 Keep **Auto-detect Terrains** on (World > Surfaces). Every active terrain without a provider is used.
- 2 For per-terrain control, add a **Terrain Surface Provider** to the terrain – or run **Tools > GrassCore > Add Terrain Providers to All Terrains**.

Meshes

- 1 Select the ground objects and run **Tools > GrassCore > Add Mesh Surface to Selection**.
- 2 For rocks and buildings use **Tools > GrassCore > Add Mesh Blocker to Selection**.
- 3 When you change renderers by script, call `RefreshRenderers()` or press **Refresh Renderers**.

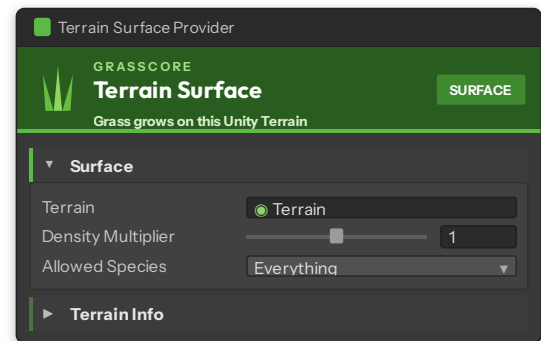


Figure 5.1 – Terrain Surface Provider.

Important Settings

Terrain Surface Provider

Setting	Default	Description
Terrain	this object	Terrain used as surface.
Density Multiplier	1	Density multiplier for all species on this terrain (0–2).
Allowed Species	Everything	Species allowed on this terrain (one bit per species index).

The closed *Terrain Info* section shows statistics of the terrain.

Mesh Surface Provider

Setting	Default	Description
Mode	Surface	<i>Surface</i> : grass grows on the meshes. <i>Blocker</i> : grass under the meshes is removed.
Include Children	on	Use all MeshRenderers on this object and its children.
Renderers	empty	Explicit renderers, used in addition to the automatic ones.
Use Vertex Color Mask	off	Red vertex colour channel as density mask (black = no grass).
Density Multiplier	1	Density multiplier on these meshes (0–2).
Allowed Species	Everything	Species allowed on these meshes.
Blocker Min Height	0.05 m	Blocker only: geometry must be at least this high above the ground to remove grass.
Track Transforms	on	<i>Advanced</i> : detect moved, rotated or scaled renderers automatically (small CPU cost per frame).

Species have their own **Mesh Surface Density** multiplier (see [chapter 6](#)).

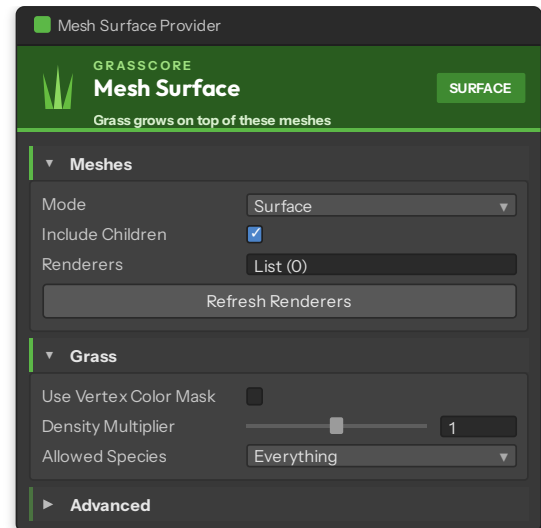


Figure 5.2 – Mesh Surface Provider.

Example Workflow

A village on a terrain

- 1 The terrain is detected automatically – grass appears everywhere.
- 2 Select all houses, fences and rocks and run *Add Mesh Blocker to Selection*. Grass disappears beneath them.
- 3 A raised stone plaza should stay bare: it is already a blocker. A grassy roof: add a *Mesh Surface* provider to the roof.
- 4 Use the red vertex colour of the roof mesh with *Use Vertex Color Mask* to paint where grass grows on it.

Performance Notes

- Surfaces are captured once per chunk tile. Moving a provider rebuilds only the chunks it overlaps.
- *Track Transforms* checks transforms every frame; switch it off for static meshes in very large scenes.
- Terrain holes are respected automatically.
- After editing a terrain from a script, call `GrassCoreRuntimeAPI.RefreshRegion(bounds)` or `RebuildAll()` to rebuild the affected tiles.

Common Problems

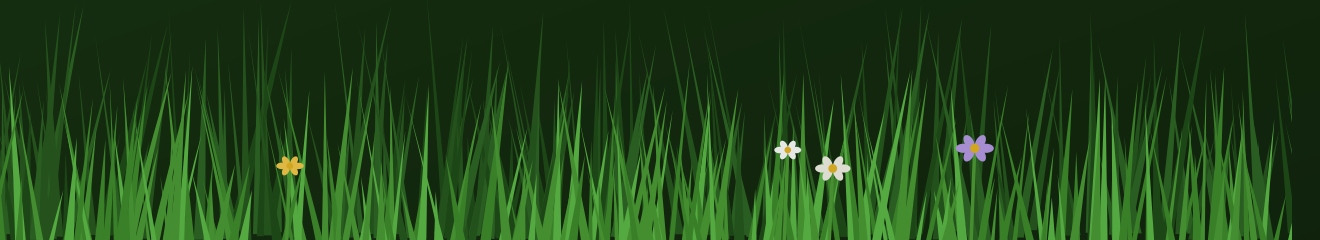
Problem	Solution
No grass surface found	Add a Terrain (auto-detected) or a Mesh Surface Provider.
Grass grows through buildings	Add a <i>Blocker</i> provider to the building or an <i>Exclude</i> mask volume.
Small objects (stones, kerbs) do not remove grass	Lower <i>Blocker Min Height</i> .
Grass floats after moving a mesh at runtime	Keep <i>Track Transforms</i> on, or call <code>MarkChanged()</code> / <code>RefreshRenderers()</code> on the provider.
A terrain should not get grass	Turn off auto-detection and add providers only to the terrains you want, or set <i>Density Multiplier</i> to 0.

PART 2

Grass & Flowers

Define what grows where: species profiles, custom meshes and LODs, procedural flowers, Grass Vegetation Areas, masks, paths and terrain painting.

06	Grass Species Profiles	20
07	Custom Grass Meshes & LOD0 / LOD1 / LOD2	23
08	Flowers & Flower Species	26
09	Grass Vegetation Areas & Local Overrides	28
10	Spline Areas & Mask Volumes	31
11	Paths, Grass Removal & Path Falloff	33
12	Terrain Layer Painting & Path Decals	35



06

GRASS & FLOWERS

Grass Species Profiles

A species is a reusable asset that describes one kind of plant – its geometry, size, density share, LOD, wind and interaction response and where it may grow. Its material defines the colours.

What it does

Every blade rendered by GrassCore belongs to a **GrassSpecies**. A world holds up to **8 species**; they share one density lattice, and each lattice point picks one species, weighted by the species' *Density* and by the local filters (slope, elevation, terrain layers, water, masks). This lets you mix grass, tall grass and flowers naturally in one field.

A species is a ScriptableObject asset (`GrassCoreSpecies`). Its **material** uses the shader **GrassCore/Grass** and defines colours, texture, lighting and glow; the species asset defines everything else.

How to set it up

- 1 Create a species with **Assets > Create > GrassCore > Grass Species** – or duplicate one from `Samples/Species` .
- 2 Assign a material that uses **GrassCore/Grass**. Leave it empty to use the default grass material.
- 3 Add the species to the world: World > **Species** > + > *Empty Slot*, then drag the asset into the slot. The list also offers *Default Meadow Grass* and *New Species Asset...*
- 4 Use the toggle and slider in each list row to switch a species on or off and set its density without opening the asset.
- 5 Select a row to edit the species inline below the list.

KEEP INDICES STABLE

The index of a species in the world list is its *mask bit*: mask volumes, paths, terrain and mesh providers refer to species by index. Disable a species with its toggle instead of removing it, so the other indices stay the same.



Figure 6.1 – Species inspector (Meadow Grass sample).

Important Settings

The species inspector groups its settings into collapsible sections.

Section / Setting	Default	Description
Enabled	on	Hide the species without removing it from the world.

Section / Setting	Default	Description
Rendering › Material	–	Material with the GrassCore/Grass shader. Empty = default grass material.
Blade Source	Procedural	<i>Procedural</i> blades, <i>Custom Meshes</i> (chapter 7) or <i>Procedural Flower</i> (chapter 8).
Blade Taper	0.85	How much narrower a procedural blade gets towards the tip (0 = rectangle, 1 = sharp tip).
Blade Roundness	0.4	Rounds the normals across the blade for a softer, cylindrical look.
Shadows	Quality	<i>Quality</i> follows the world's quality setting, <i>Off</i> never casts shadows.
Size & Shape › Height / Width	0.35–0.75 / 0.035–0.06 m	Random height and width range per blade.
Height From Density	0.6	How strongly local density (masks, layer weights, patch edges) shortens blades.
Curvature	0.35	Natural forward bend.
Align To Surface	0.25	How much blades follow the surface normal (0 = always vertical).
Face Camera	0.3	Rotates thin blades towards the camera to avoid paper-thin silhouettes.
Far Width Expansion	15	Extra width at distance that compensates for the reduced far density.
Density & LOD › Density	1	Share of the world lattice used by this species (0–1).
Max Distance	0	Maximum render distance of this species (0 = world render distance).
LOD1/ LOD2 Distance	18 / 55 m	Where LOD1 and LOD2 start (scaled by the quality LOD Bias). A bar shows the ranges.
Wind › Strength, Flutter	1, 0.35	Overall wind response and high-frequency flutter.
Stiffness	2	Higher values keep the lower part of the blade straighter (1–4).
Phase Variation	1	Random phase offset between blades.
Interaction › Bend / Flatten Response	1 / 1	Response to bending and flattening (characters, vehicles, impulses, trails).
Mow Response / Cut Style	1 / Auto	How much mowers shorten the species and how cut plants look (chapter 17).
Season & Snow	1 / 1 / 1	<i>Season Response</i> (0 = evergreen), <i>Snow Response</i> , <i>Color Zone Response</i> .
Placement Filters › Slope	0–40° (falloff 6°)	Allowed slope with a soft transition.
Elevation	±10 000 m (falloff 2 m)	Allowed world height with a soft transition.
Terrain Layer Mode / Terrain Layers	Include / empty	<i>Include</i> : grow only on the listed layers (weighted by their splat weight). <i>Exclude</i> : never on them. Empty = all layers.
Mesh Surface Density	1	Density multiplier on Mesh Surface Providers.
Water & Shoreline › Water Clearance	0.05 m	Minimum height above the water surface.
Shoreline Width / Density	3 m / 0	Density change in a band along the shore (–1 = none at the shore, +2 = triple).
Aquatic	off	The species only grows below the water surface.

The GrassCore/Grass material

Group	Properties
Colour	Root Colour, Tip Colour, Gradient Power, Dry Colour, Dry Variation, Field Noise Scale (m), Field Noise Strength, Root Occlusion
Texture	Albedo (RGB) Alpha (A) with tiling, Alpha Clipping and Alpha Cutoff (for textured cards, flowers, ferns)
Lighting	Wrap Lighting, Normal Softening, Translucency, Translucency Tint, Smoothness, Specular Strength, Wetness Darkening
Bioluminescence	Glow toggle, Glow Colour, Glow Intensity, Glowing Tip Length, Pulse Speed / Amount / Scale, Per-Blade Variation, Reactive Glow Colour, Reactive Glow (see chapter 21)
Advanced	Render queue

Sample species

Sample	Type	Notes
Meadow Grass	Procedural blades	0.35–0.75 m, density 0.85 – a good base layer.

Sample	Type	Notes
Tall Grass	Procedural blades	0.7-13 m, density 0.15 – mix in for variation.
Dry Steppe Grass	Procedural blades	Short, dry colours.
Wildflowers	Procedural blades	Low density colour accents.
Shore Reeds	Procedural blades	1-18 m, <i>Shoreline Density</i> = 2 – grows along water.
Glowgrass	Procedural blades	Bioluminescent material (chapter 21).
Demo White / Blue Wildflowers, Demo Poppies	Procedural Flower (Radial)	Daisy-like and poppy flowers.
Demo Red / Yellow Tulips	Procedural Flower (Tulip)	Closed cups with broad leaves.
Demo Lavender	Procedural Flower (Spike)	Lavender clumps.

Example Workflow

A natural meadow with three species

- 1 World › Species: slot 0 *Meadow Grass* (density 0.85), slot 1 *Tall Grass* (0.15), slot 2 *Demo White Wildflowers* (0.04).
- 2 Tall Grass › Placement Filters: *Slope* 0–25° so it stays on flat ground.
- 3 Wildflowers › Placement Filters: *Terrain Layer Mode* = *Include* with your grass layer, so flowers never grow on dirt or rock.
- 4 Meadow Grass material: lower *Dry Variation* for a fresh spring look, raise *Field Noise Strength* for large-scale colour patches.
- 5 Look at the result with *Debug View* = *Species* (World › Debug) to see which species grows where.

Performance Notes

- All species share the same lattice – adding species does not add blades. Species densities are shares, not additions.
- Each species and LOD in use is one indirect draw call.
- Filters (slope, elevation, layers, water) are evaluated once per tile, not per frame; changing them rebuilds tiles over a few frames.
- *Max Distance* per species is an effective way to stop small flowers rendering far away.
- *Shadows* = *Off* on short or sparse species (flowers, ground cover) saves shadow-pass cost.

Common Problems

Problem	Solution
Species does not appear	Is it enabled, in the world list and is its <i>Density</i> > 0? Check the filters – an empty layer list means <i>all</i> layers in <i>Include</i> mode.
A species dominates the field	Densities are relative shares. Lower its <i>Density</i> or raise the others.
Grass looks paper-thin from the side	Raise <i>Face Camera</i> or <i>Blade Roundness</i> .
Far grass looks sparse	Raise <i>Far Width Expansion</i> or the quality <i>Halving Distance</i> .
The material should use the 'GrassCore/Grass' shader	Assign a material with the GrassCore/Grass shader (or a shader built on its includes).

07

GRASS & FLOWERS

Custom Grass Meshes & LOD0 / LOD1 / LOD2

Use your own meshes – clumps, ferns, cards, pebbles – as a grass species, with three levels of detail and the same culling, wind, interaction and mowing as procedural blades.

What it does

Every species renders in three geometric levels of detail. With *Blade Source = Procedural* GrassCore builds tapered blades whose segment count comes from the quality preset. With **Custom Meshes** you provide the geometry for LOD0, LOD1 and LOD2 yourself.

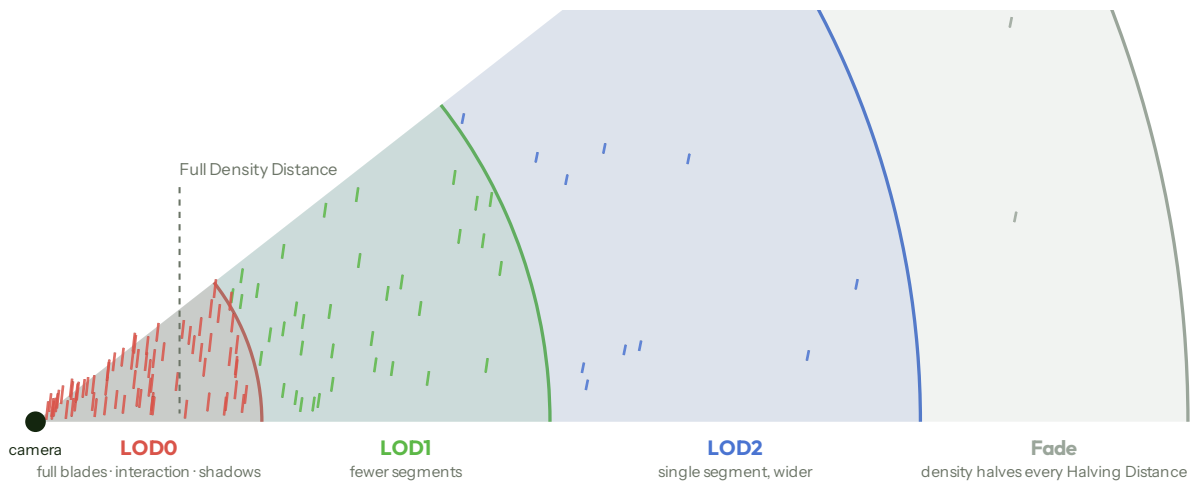


Figure 7.1 – LOD ranges around the camera. Beyond the full density distance the density halves every Halving Distance and fades out smoothly.

Level	Used for	Typical content
LOD0	Close range (up to <i>LOD1Distance</i>)	Full blade or mesh, interaction, shadows.
LOD1	Middle range (up to <i>LOD2Distance</i>)	Reduced geometry, optional shadows (Ultra).
LOD2	Far range (up to the render distance)	Cheapest representation; procedural blades use a single segment.
Fade	Beyond <i>Full Density Distance</i>	Density halves every <i>Halving Distance</i> and fades out per level – no popping rings.

Procedural blade geometry per quality preset (a blade with n segments has $2n - 1$ triangles and $2n + 1$ vertices):

Preset	LOD0	LOD1	LOD2
Low	3 seg · 5 tris	2 seg · 3 tris	1 seg · 1 tri
Medium	4 seg · 7 tris	2 seg · 3 tris	1 seg · 1 tri
High	5 seg · 9 tris	3 seg · 5 tris	1 seg · 1 tri
Ultra	6 seg · 11 tris	3 seg · 5 tris	2 seg · 3 tris

How to set it up

- 1 Model the plant with its **pivot at the root, +Y up; X** is the width axis (Z is depth).
- 2 Export up to three LOD meshes. Missing LODs fall back to the previous one.
- 3 In the species set **Blade Source = Custom Meshes** and assign the meshes to **LOD Meshes (0, 1, 2)**. Until LOD0 is assigned, procedural blades are used and the inspector shows a warning.
- 4 For cards with transparency (leaves, ferns, flowers) enable **Alpha Clipping** in the GrassCore/Grass material and use a texture with alpha.
- 5 Set **Height** (final plant height in metres) and **Width (mesh scale)** in **Size & Shape**.

HOW CUSTOM MESHES ARE SCALED

The mesh is scaled vertically so that its top (bounds max Y) matches a random value of **Height**. X and Z are multiplied by **Width (mesh scale)** – a value of 1 keeps the mesh's original width. Wind, interaction and curvature bend the mesh along its height, exactly like a procedural blade.

Important Settings

Setting	Default	Description
Blade Source	Procedural	<i>Custom Meshes</i> enables the LOD mesh slots.
LOD Meshes (0, 1, 2)	empty	Custom meshes per LOD. Missing LODs reuse the previous one.
Height	0.35–0.75 m	Final height range; the mesh is normalised to it.
Width (mesh scale)	0.035–0.06	Scale of X and Z. For meshes, values around 1 are typical.
LOD1/ LOD2 Distance	18 / 55 m	Switch distances, multiplied by the quality <i>LOD Bias</i> .
Max Distance	0	Optional per-species render distance (0 = world).
Shadows	Quality	<i>Off</i> is recommended for small or dense meshes.
Cut Style	Auto	For custom meshes <i>Auto</i> = Shrink : the whole plant is scaled down to the cut height (chapter 17).
Blade Segments (quality)	5 / 3 / 1	Procedural blades only – segments per LOD of the active quality preset.

Recommended geometry budgets

GrassCore draws hundreds of thousands of instances, so per-instance geometry matters much more than for normal props. The built-in procedural flowers (36 / 18 / 6 triangles) are a good reference. Recommended budgets for custom grass meshes:

Content	LOD0	LOD1	LOD2
Grass clump / tuft	12–40 tris	6–16 tris	2–6 tris
Flower or small plant	24–60 tris	12–30 tris	4–8 tris
Fern / leafy card cluster	30–80 tris	12–30 tris	4–8 tris
Hero plant (low density)	up to 150 tris	40–60 tris	8–12 tris

Keep vertex counts low (shared vertices, no unnecessary UV seams) and use alpha-clipped cards for complex outlines instead of modelling every leaf.

Example Workflow**Fern clumps under trees**

- 1 Model a fern with three LODs: 64, 24 and 6 triangles, alpha-clipped leaf cards, pivot at the root.
- 2 Create a species *Fern*, *Blade Source* = *Custom Meshes*, assign the three meshes.
- 3 Material: GrassCore/Grass with the fern texture, *Alpha Clipping* on, *Translucency* around 0.8.
- 4 Size: *Height* 0.4–0.8 m, *Width (mesh scale)* 0.9–1.2. *Density* 0.05, *Max Distance* 60 m.
- 5 Placement: *Slope* 0–35°, or restrict it to a Grass Vegetation Area under the trees ([chapter 9](#)).
- 6 Set *Shadows* = *Off* if the ferns are dense.

Performance Notes

- Triangles per instance multiply with the instance count. A 60-triangle LOD0 at 50 000 visible instances is 3 million triangles – keep dense species simple.
- LOD2 is drawn most often at distance; keep it at a few triangles.
- Use *Max Distance* for low-density detail species so they do not reach the full render distance.
- Alpha-clipped materials are slightly more expensive than opaque ones; use them only where needed.

Common Problems

Problem	Solution
Mesh lies on its side	The mesh must be +Y up. Re-export or rotate it in your DCC tool (custom grass meshes have no rotation setting).
Mesh floats or is buried	The pivot must be at the root (y = 0 at the base).
Mesh far too wide	<i>Width</i> is a scale for custom meshes – use values around 1 instead of blade widths in metres.
Leaves render as solid quads	Enable <i>Alpha Clipping</i> in the material and assign a texture with alpha.
Visible LOD switch	Make LODs more similar in silhouette, or move <i>LOD1 / LOD2 Distance</i> further away.

08

GRASS & FLOWERS

Flowers & Flower Species

Procedural Flower species generate real small 3D flowers – stem, leaves, alpha-clipped petal head and raised centre – with three LODs. They grow, move and are culled exactly like grass.

What it does

Set a species' *Blade Source* to **Procedural Flower** and GrassCore builds the flower geometry itself. Three styles are available:

Style	Looks like	Triangles LOD0 / LOD1 / LOD2	Samples
Radial	Daisy, poppy, cornflower: stem, leaves, cupped petal head, raised centre	36 / 18 / 6	Demo White Wildflowers, Demo Blue Wildflowers, Demo Poppies
Tulip	Closed cup of six petals on a strong stem with broad basal leaves	44 / 28 / 7	Demo Red Tulips, Demo Yellow Tulips
Spike	Lavender clump: up to three stalks with flower spikes, basal leaves	28 / 18 / 6	Demo Lavender

Flowers share the grass lattice, wind, interaction, mowing, fire, seasons and colour zones. For flowers **Width** is the head (or spike) diameter and **Height** the plant height.

How to set it up

- 1 Duplicate a **Demo ...** species from **Samples/Species** – the easiest start, because it comes with a matching material and atlas.
- 2 Or create a new species and set **Blade Source = Procedural Flower**. Without a material, the default flower material of the chosen style is used (Radial: white wildflowers, Tulip: red tulips, Spike: lavender).
- 3 Choose the **Flower Style** and shape the flower in the *Rendering* section.
- 4 Set **Height** and **Head Diameter** in *Size & Shape*.
- 5 Keep **Alpha Clipping** enabled in the material – otherwise the petal cups render as solid discs (the inspector warns about this).
- 6 Add the species to the world with a low **Density** (0.03–0.06), or to a Grass Vegetation Area ([chapter 9](#)).

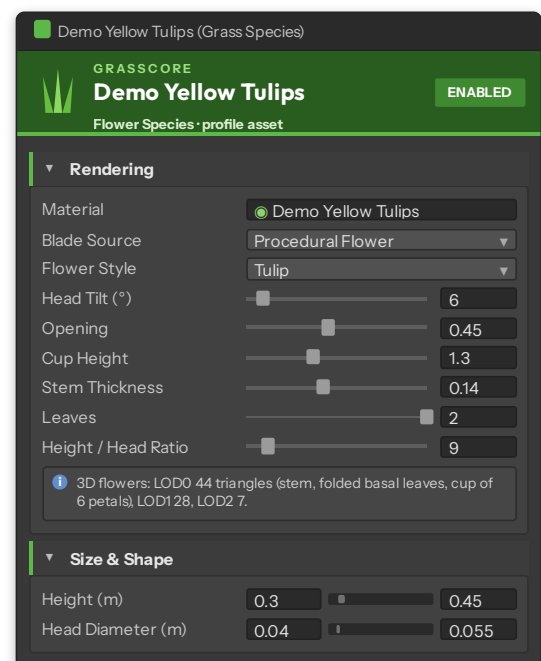


Figure 8.1 – A Tulip flower species (Demo Yellow Tulips).

Important Settings

Setting	Default	Description
Flower Style	Radial	Radial, Tulip or Spike.
Head Tilt (°)	28	Tilt of the flower head (Radial, Tulip).
Cupping / Opening	0.3	Radial: how much the petals cup. Tulip: how far the cup opens.
Centre Size / Centre Height	0.36 / 0.14	Radial: size and height of the raised centre (fraction of the head diameter).
Cup Height	135	Tulip: height of the cup in head diameters.
Stalks / Spike Length	3 / 0.3	Spike: stalks per plant (1–3) and spike length (fraction of the plant height).

Setting	Default	Description
Stem Thickness	0.08	Fraction of the head diameter.
Leaves	2	Leaves on LOD0 (0–2); tulip leaves stay visible on LOD1 / LOD2 as simpler shapes.
Leaf Length / Width / Position / Upright	0.9 / 1 / 0.2 / 0	Leaf length in head diameters, width (1= slender daisy leaf), position along the stem (0 = basal leaves), 0 = arching, 1= upright.
Height / Head Ratio	6	Proportion of plant height to head size.
Height / Head Diameter	per species	Plant height and head diameter in metres.

Materials and atlases

Petals, centre, stem and leaf colours come from the material's texture **atlas**: all styles share one layout (left half: petal head, next quarter: centre, last quarter: stem and leaves). Example atlases are in [Samples/Species/Textures](#). To create a new colour variant, duplicate a sample material and paint a new atlas with the same layout.

Example Workflow

A tulip field with a path

- 1 Duplicate *Demo Red Tulips* and *Demo Yellow Tulips*.
- 2 Create a Grass Vegetation Area in *Replace* mode and add *Meadow Grass* (density 1) to the grass list and both tulip species (density 1) to the flower list.
- 3 Raise the tulips' *Density* in the area via **Local Overrides** – the profiles stay unchanged ([chapter 9](#)).
- 4 Draw a path through the field ([chapter 11](#)); *Mow Response* of the tulips stays 1, so mowers can cut them.

Performance Notes

- Flowers cost 6–44 triangles per instance – keep their *Density* low (the samples use 0.04–0.06).
- Shadows = Off* is a good choice for small flowers.
- Use *Max Distance* so flowers do not render up to the full grass distance.
- Flower meshes are generated once per shape and shared by all species with the same shape.

Common Problems

Problem	Solution
Flowers look like solid discs	Enable <i>Alpha Clipping</i> on the material.
Wrong colours	The material atlas defines the colours; use a sample material or an atlas with the same layout.
Flowers too large or small	<i>Width</i> is the head diameter (eg. 0.04–0.06 m) and <i>Height</i> the whole plant.
Flowers everywhere	Lower <i>Density</i> , or restrict them with filters or a Grass Vegetation Area.

Grass Vegetation Areas & Local Overrides

Draw an outline on the ground and decide exactly which grass and flower species grow inside it – with their own amounts, sizes and flower shapes, without changing the species profiles.

What it does

A **Grass Vegetation Area** is a spline outline with its own species mix. Use it for flower beds, tulip fields, a dry clearing, a lush patch near water or a different grass type in a garden.

- **Replace** mode: only the area's species grow inside the outline; the surrounding grass fades out over the *Falloff*.
- **Add** mode: the area's species grow on top of the existing grass.
- Species that are **not** in the world's species list are placed into free species slots automatically and grow **only inside areas** (badge *area*). Species from the world list keep growing outside (badge *world*). A world holds 8 species in total, including area species.
- **Local Overrides** change *Density*, *Size* & *Shape* and the *Flower Shape* of an entry **only inside this area**. The species profile is never modified – the entry is drawn as a runtime-only variant (badge *local*) that uses no extra species slot.

Every area is independent: the same profile can be used in many areas with different amounts and overrides at the same time. Nested areas keep their own configuration – the inner (smaller) area wins inside its outline and blends smoothly over its edge; overlapping areas combine.

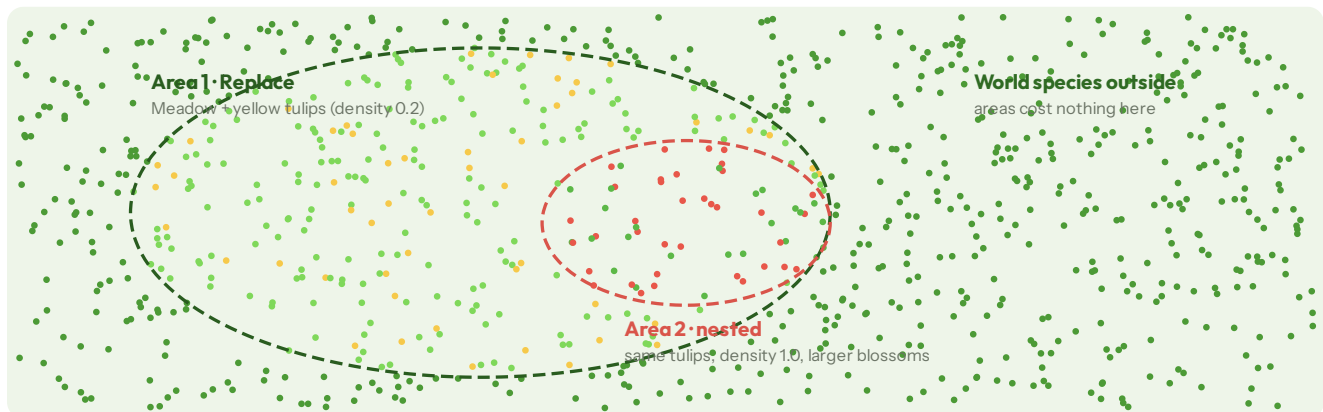


Figure 91 – Two nested areas using the same tulip profile with different densities and blossom sizes.

How to set it up

- 1 Create an area with **GameObject** › **GrassCore** › **Grass Vegetation Area** – or with the button in World › Species › *Grass Vegetation Areas* or in the Setup Wizard (*More Tools*).
- 2 Draw the outline: press **Edit Points in Scene View**, drag points (they stick to the terrain), click the line to insert a point, Shift + click to add one, Ctrl/Cmd + click or Delete to remove one, Esc to stop.
- 3 Choose the **Mode** (*Replace* or *Add*).
- 4 Add species with the + button of the *Grass Species* or *Flower Species* list. The menu lists all species assets of that kind, an *Empty Entry* and *New Species Asset...*. The two lists are only for overview – both kinds work in either list.
- 5 Set each entry's amount with its slider (1 = the species' full density) and switch entries on or off with their toggle.
- 6 Select an entry to edit its **Local Overrides**: tick a value to override it; unticked values show the profile value greyed out.
- 7 Shape the edge with **Falloff (m)** and **Edge Noise (m)**.

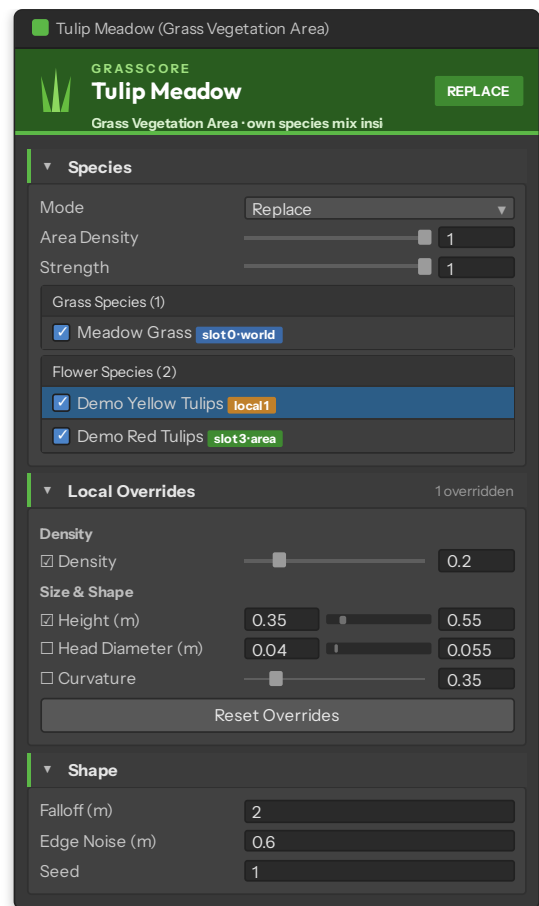


Figure 9.2 – Grass Vegetation Area inspector with a selected entry and its local overrides.

Important Settings

Setting	Default	Description
Mode	Replace	<i>Replace</i> : only the area's species grow inside. <i>Add</i> : they grow on top of the existing grass.
Area Density	1	Overall density multiplier of the area's species.
Strength	1	0 = no effect, 1 = full effect – blend an area in or out.
Grass / Flower Species	–	Entries with toggle, species and amount (fraction of the species' density).
Falloff (m)	2	Soft edge inside the outline: area species fade in, surrounding grass fades out.
Edge Noise (m)	0.6	Irregular outline for a natural look (0 = exact outline).
Seed	1	Seed of the edge noise.

Local Overrides

Override	Applies to
Density	Share of the world lattice inside this area (replaces the species' <i>Density</i>).
Height / Width (Head Diameter for flowers, mesh scale for custom meshes)	Size ranges inside the area.
Height From Density, Curvature, Align To Surface, Face Camera, Far Width Expansion	Shape values, as in the species profile.
Flower Shape	Procedural flowers: style, tilt, cup / spike, stem and leaves inside the area.

Reset Overrides clears all overrides of the selected entry. Entries with identical overrides share one variant; up to **24 different override sets** can exist per scene.

Entry badges

Badge	Meaning
slot n · world	The species is in the world list: it grows everywhere, the area sets its amount inside.
slot n · area	Placed automatically: grows only inside Grass Vegetation Areas.
local n	The entry has local overrides and is drawn as local variant n .
no slot (red)	More than 8 species (world list + area species): this species does not render.
slot n · Δ	More than 24 different override sets in the scene: this entry uses the profile values.

Example Workflow

One tulip profile, two different fields

- 1 Area 1 (*Replace*): add *Meadow Grass* and *Demo Yellow Tulips*. Select the tulips and override **Density = 0.2**.
- 2 Area 2, a smaller area inside Area 1: add the same *Demo Yellow Tulips*, override **Density = 1** and a larger **Head Diameter**.
- 3 Both fields show their own configuration – Area 2 wins inside its outline and blends over its edge into Area 1.
- 4 The *Demo Yellow Tulips* asset itself is unchanged and still renders with its original values anywhere else.

Performance Notes

- **Zero cost without areas:** the area shader variants and the per-texel area data exist only while at least one area exists.
- Areas are applied when chunk tiles are built – there is no per-frame cost.
- Changing an area's outline, species or amounts rebuilds only the chunks under the area. Editing override values never rebuilds tiles.
- Area-only species use species slots like normal species (8 in total). Overrides use separate draw slots and do not reduce the 8 species slots.

Common Problems

Problem	Solution
No species assigned: in Replace mode the area removes all grass inside it	Add at least one entry, or switch to <i>Add</i> mode.
A species is listed more than once; the highest density is used	Remove duplicate entries without overrides.
A world holds at most 8 species (red)	Remove unused species from the world list or from areas.
More than 24 different local override sets in the scene	Reuse identical overrides (they are shared) or remove some overrides.
Hard visible edge	Increase <i>Falloff (m)</i> and <i>Edge Noise (m)</i> .

Spline Areas & Mask Volumes

Mask volumes and spline areas change the grass density in a region – remove it, thin it out, override it or restrict species – with soft edges and optional textures you can paint in the Scene view.

What it does

A **Mask Volume** modifies the grass density inside a shape. It is applied when chunk tiles are built, so it has no per-frame cost.

Shape	Description
Box, Sphere, Cylinder	Unit-size shape scaled by the transform. Optional mask texture projected from above.
Spline	Free-form area drawn with points in the Scene view; follows the terrain. Created with GameObject > GrassCore > Spline Mask Area .

Mode	Effect
Exclude	Removes grass inside the volume (default).
Multiply	Multiplies the density by <i>Density</i> .
Override	Replaces the density with <i>Density</i> .
Include Only	The affected species grow only inside Include volumes.

The same spline system is used by paths ([chapter 11](#)), colour zones ([chapter 20](#)), burn sources ([chapter 18](#)), water volumes ([chapter 22](#)) and Grass Vegetation Areas ([chapter 9](#)).

How to set it up

- 1 Create a volume with **GameObject > GrassCore > Mask Volume** and scale its transform – or a free-form area with **GameObject > GrassCore > Spline Mask Area**.
- 2 Choose **Shape** and **Mode**, set **Strength** and the soft edge (**Falloff** for box / sphere / cylinder, **Falloff (m)** and **Edge Noise (m)** for splines).
- 3 Restrict the effect to some species with **Affected Species** (a mask of the world's species slots).
- 4 Optional (box, sphere, cylinder): assign a **Mask Texture** (red channel) – or paint one directly in the Scene view (see below).

Editing spline shapes

Action	How
Start / stop editing	Edit Points in Scene View button · Esc stops editing
Move a point	Drag it – points stick to the terrain
Insert a point	Click on the line
Add a point	Shift + click
Remove a point	Ctrl / Cmd + click, or select it and press Delete
Path width at a point	Drag the square handles of the selected point (paths only)
Whole shape	Move, rotate or scale the GameObject

The *Spline Shape* settings are **Type** (Area = closed outline, Path = line with a width), **Width (m)** for paths, **Smoothness** (0 = straight lines, 1 = smooth curves) and **Resolution (m)** (size of one GPU sample of the shape). Buttons: **Snap to Ground**, **Reverse**, **Reset Shape**; a selected point has a **Width Multiplier** (paths) and **Delete Point**.

Painting a density mask

Open the closed **Density Painting** section of a box / sphere / cylinder volume. **Create Paint Texture...** creates a readable mask texture. Press **Paint in Scene View**: left mouse raises the mask value, Shift + left mouse lowers it, **[** and **]** resize the brush. *Brush Size (m)*, *Strength* and *Hardness* control the brush; *Fill White* and *Fill Black* reset the texture. While painting, the header badge shows **PAINTING**.

Important Settings

Setting	Default	Description
Shape	Box	Box, Sphere, Cylinder or Spline.
Mode	Exclude	Multiply, Exclude, Include Only or Override.
Strength	1	Strength of the effect.
Falloff	0.15	Box / sphere / cylinder: soft edge as a fraction of the volume size.
Falloff (m) / Edge Noise (m)	15 / 0	Spline: soft edge inside the outline and irregularity of the outline.
Density	0.25	Target density for <i>Multiply</i> and <i>Override</i> (0–2).
Mask Texture / Invert Texture	– / off	Optional texture (red channel) projected from above.
Affected Species	Everything	Species affected by this volume.

Example Workflow

A clearing with a soft, natural edge

- 1 **GameObject** › **GrassCore** › **Spline Mask Area**; draw the outline of the clearing.
- 2 Mode *Exclude*, *Falloff (m)* 3, *Edge Noise (m)* 15.
- 3 To keep some flowers in the clearing, untick the flower species in *Affected Species*.
- 4 For a thin, patchy lawn instead: Mode *Multiply*, *Density* 0.35.

Performance Notes

- Masks are evaluated once per tile build. Up to 32 mask volumes can affect one chunk tile; up to 16 mask textures are used at a time.
- Changing a mask rebuilds only the chunks it overlaps.
- Spline shapes are baked into small GPU cells; a smaller *Resolution* gives crisper details at slightly higher memory.

Common Problems

Problem	Solution
Volume has no effect	Is the volume overlapping the ground? Scale the transform (box / sphere / cylinder) and check <i>Affected Species</i> .
An area needs at least 3 points	Add points (Shift + click) or press <i>Reset Shape</i> .
Cannot paint	The mask texture must be readable: enable <i>Read/Write</i> in its import settings, or create one with <i>Create Paint Texture...</i>
Spline masks ignore the vertical scale	Spline shapes project onto everything below them; scale only horizontally.
Include Only removes everything	Include Only affects the selected species everywhere outside Include volumes – restrict <i>Affected Species</i> .

11

GRASS & FLOWERS

Paths, Grass Removal & Path Falloff

Draw footpaths, dirt roads and trails as splines. GrassCore removes or thins the grass along the line, with a soft edge where the grass grows back in.

What it does

A **GrassCorePath** is a spline line with a width. Along the line it removes grass (or applies another mask effect), and beyond the edge the grass grows back in over the **EdgeFalloff**. Each point can have its own width multiplier, so a path can widen and narrow naturally.

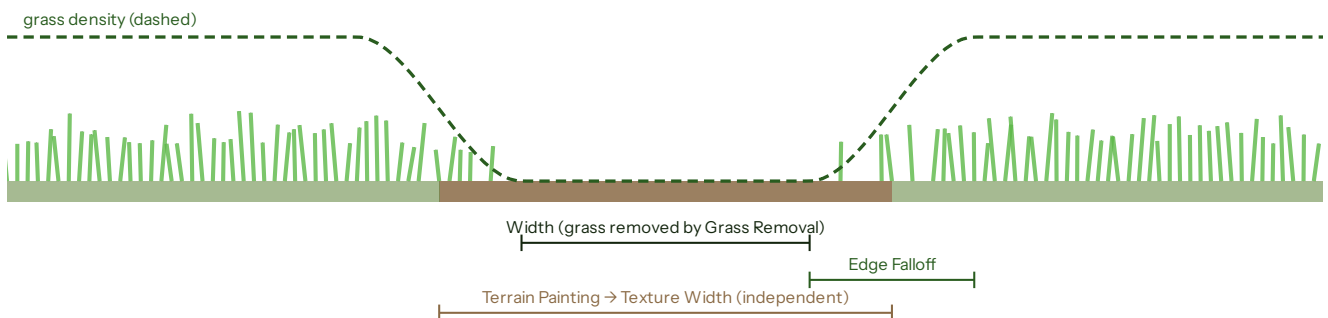


Figure 11.1 – Cross-section of a path: grass is removed inside the Width, grows back in over the Edge Falloff. Terrain painting (chapter 12) has its own, independent width.

How to set it up

- 1 **GameObject** > **GrassCore** > **Path (Spline)** creates a path on the ground.
- 2 Press **Edit Points in Scene View** and draw the centre line (see the editing table in chapter 10). Drag the square handles of a selected point to change the width there.
- 3 Pick a preset in the **Presets** section or set the values in **Path & Grass Removal**.
- 4 Optional: paint a terrain layer or render a material along the path (chapter 12).

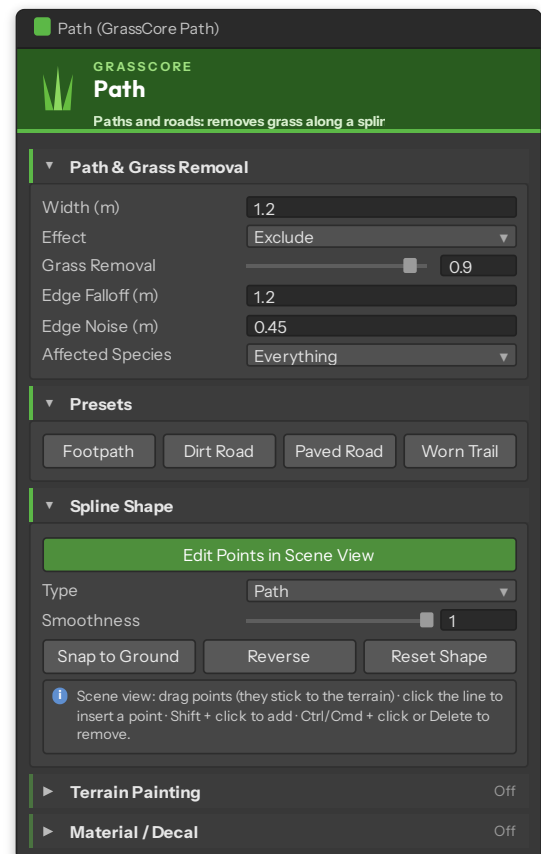


Figure 11.2 – Path inspector.

Important Settings

Setting	Default	Description
Width (m)	3	Width in which grass is removed. Per-point widths multiply it.
Effect	Exclude	<i>Exclude</i> removes grass. <i>Multiply / Override</i> set a density, <i>Include Only</i> makes the affected species grow only along the path.
Grass Removal	1	<i>Exclude</i> : 1 = no grass left on the path, 0.5 = half the density (worn trail). Called <i>Strength</i> for the other effects.
Density	0.3	Target density for <i>Multiply / Override</i> .
Edge Falloff (m)	15	Soft transition beyond the edge where the grass grows back in.
Edge Noise (m)	0.4	Irregular edges for a natural, trodden look (0 = clean edges, eg. paved roads).
Affected Species	Everything	Species affected by the path.

Presets

Preset	Width	Grass Removal	Edge Falloff	Edge Noise	Use
Footpath	12 m	0.9	12 m	0.45 m	Narrow trodden trail, partly overgrown edges.
Dirt Road	4 m	1	2 m	0.6 m	Vehicle track, soft verges.
Paved Road	6 m	1	0.4 m	0 m	Clean edges, no grass.
Worn Trail	2 m	0.55	15 m	0.6 m	Grass thinned out, not removed.

Example Workflow

A trail that widens into a resting place

- 1 Create a path and apply *Footpath*.
- 2 Draw the trail; at the resting place select a point and drag its square handles to a *Width Multiplier* of about 3.
- 3 Raise *Edge Noise* for a more organic outline.
- 4 Paint a dirt terrain layer along the path with *Terrain Painting* ([chapter 12](#)).

Performance Notes

- Paths are spline shapes baked into GPU cells and applied when tiles are built – no per-frame cost.
- Editing a path rebuilds only the chunks under it, live while you drag.
- Very long paths with a small *Resolution (m)* need more GPU cells; the default is fine for most paths.

Common Problems

Problem	Solution
A path needs at least 2 points	Add points with Shift + click.
Path floats above or cuts into the ground	Splines project vertically onto the ground – press Snap to Ground after large terrain edits.
Grass edge too hard / too soft	Adjust <i>Edge Falloff (m)</i> and <i>Edge Noise (m)</i> .
Some species should stay on the path	Untick them in <i>Affected Species</i> .

Terrain Layer Painting & Path Decals

Give paths a real ground surface: paint a Terrain Layer along the path non-destructively, or render a soft-edged material ribbon without touching the terrain.

What it does

Every path has two optional ground surfaces. Both are **off by default** and independent of the grass removal width.

Surface	What it does	Runtime cost
Terrain Painting	Paints a Terrain Layer into the terrain's splat map along the path, with its own width, opacity, soft edge and noise. Repainted automatically when you edit the path.	None – painting happens in the editor.
Material / Decal	Renders a soft-edged material ribbon on top of the ground (colour, texture, normal map or your own material). The terrain is not changed.	One mesh per path.

Terrain painting is **non-destructive**: the original texels under the path are stored in a backup asset (*Path Paint Backup*, folder `Assets/GrassCore/Path Paint`). **Remove Painting** restores the original terrain, and both painting and removal can be undone.

How to set it up

Terrain Painting

- 1 Select the path and open the **Terrain Painting** section.
- 2 Enable **Paint Terrain Layer** and pick a **Terrain Layer** (it is added to the terrain if it is not used there yet).
- 3 Set **Texture Width**, **Opacity**, **Edge Falloff (m)**, **Edge Noise (m)** and **Noise Size (m)**.
- 4 Press **Paint**. With **Auto Repaint** on, the layer is repainted shortly after every change; **Repaint** repaints manually.
- 5 **Save the project** after painting so the terrain changes are stored.
- 6 **Remove Painting** restores the original terrain and switches painting off. **Make Permanent...** keeps the painting and deletes the backup (cannot be restored afterwards).

Terrain painting is edited outside Play mode.

Material / Decal

- 1 Open **Material / Decal** and enable **Use Material / Decal**.
- 2 Assign your own material, or leave it empty and set the built-in material's **Color**, **Texture** and **Normal Map**. Soft edges need the shader **GrassCore/Path Surface** – other shaders ignore the edge fade.
- 3 Set **Width**, **Opacity**, **Edge Falloff**, **Edge Noise**, **Tile Length** and **Height Offset**. After sculpting the terrain press **Rebuild Surface**.

The Scene view shows the grass removal width, the painted texture width with its falloff and the decal width.

Important Settings

Terrain Painting

Setting	Default	Description
Paint Terrain Layer	off	Enables painting (editor, non-destructive).
Terrain Layer	–	Layer painted along the path.
Texture Width	4 m	Width painted at full strength.
Opacity	1	Strength in the middle of the path.
Edge Falloff	15 m	Soft fade into the original terrain on both sides.
Edge Noise / Noise Size	0.5 m / 2.5 m	Irregular edges and the size of the irregularities.

Setting	Default	Description
Follow Point Widths	on	The painted width follows the per-point width multipliers.
Auto Repaint	on	Repaint automatically after changes (editor).

Material / Decal

Setting	Default	Description
Use Material / Decal	off	Draws a material along the path.
Material (optional)	–	Your own material; empty = built-in material with Color / Texture / Normal Map.
Width / Opacity	3 m / 1	Width at full strength and strength in the middle.
Edge Falloff / Edge Noise / Noise Size	0.8 m / 0.35 m / 2 m	Soft, irregular edges.
Tile Length	3 m	Length of one texture repeat along the path.
Height Offset	0.03 m	Lift above the ground (avoids flickering).
Resolution	0.75 m	Vertex spacing: smaller follows the ground more closely.
Follow Point Widths	on	The surface width follows the per-point widths.

Menu commands

Command	Use
Tools › GrassCore › Path Painting › Repaint All Paths	Repaints every painted path.
Tools › GrassCore › Path Painting › Restore Terrain Under All Paths	Removes all path painting and restores the terrain.
Tools › GrassCore › Path Painting › Restore Backups of Deleted Paths	Restores terrain areas painted by paths that were deleted.

Example Workflow

Dirt road through a meadow

- 1 Path with the *Dirt Road* preset: grass removed in 4 m, soft verges.
- 2 *Terrain Painting*: dirt layer, *Texture Width* 3.5 m, *Edge Falloff* 1.5 m, *Edge Noise* 0.6 m – the dirt shows slightly inside the grass edge.
- 3 For a muddy track on top, enable *Material / Decal* with a mud texture, *Width* 2.5 m, *Opacity* 0.7.

Performance Notes

- Terrain painting is editor-only and costs nothing at runtime – the result is ordinary terrain splat data.
- A decal is one mesh with one draw call per path; *Resolution* controls its vertex count.
- Keep the backup assets in your project (they are needed to restore the terrain).

Common Problems

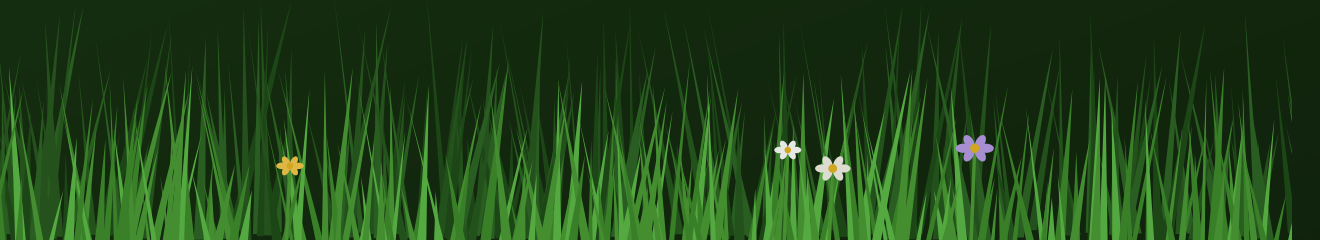
Problem	Solution
Painting is lost after reopening the scene	Save the project after painting – terrain data is an asset.
Painting cannot be removed	Keep the backups in <code>Assets/GrassCore Path Paint</code> (unless you pressed <i>Make Permanent...</i>). For deleted paths use <i>Restore Backups of Deleted Paths</i> .
The path changed since it was painted	Press Repaint , or enable <i>Auto Repaint</i> .
Decal flickers with the terrain	Increase <i>Height Offset</i> .
Decal does not follow the ground	Lower <i>Resolution</i> (smaller vertex spacing). After sculpting the terrain press Rebuild Surface (or call <code>RebuildDecal()</code>).

PART 3

Wind, Interaction & Effects

Bring the grass to life: wind, characters and vehicles, explosions, rotor downwash, mowing, fire, seasons and snow, colour zones and glowing grass.

13	Wind & Wind Interaction	38
14	Player & Object Interaction	40
15	Explosion Shockwaves	43
16	Helicopter & Propeller Downwash	45
17	Mowing, Cut Height & Grass Clippings	47
18	Fire, Burning & Burned Grass	50
19	Seasons & Snow Accumulation	52
20	Dynamic Color Zones	54
21	Bioluminescent & Magic Grass	56



13

WIND, INTERACTION & EFFECTS

Wind & Wind Interaction

A global wind field with gusts that roll across the field, small-scale turbulence and blade flutter – driven from the inspector, a Unity WindZone, other Core products or your own scripts.

What it does

The **wind system** computes one global wind state per frame and animates every blade in the vertex shader – no CPU work per blade. Gusts are large travelling patterns, turbulence adds small chaotic motion and flutter makes thin blades tremble. Each species decides how strongly it reacts.

The same wind drives trees and bushes ([chapter 30](#)), flying grass clippings ([chapter 17](#)) and roughens water surfaces ([chapter 24](#)).

Wind sources, from lowest to highest priority:

- 1 The **base wind** in the World's *Wind* tab.
- 2 A **Unity WindZone** (*Use Wind Zone*): direction, main strength and turbulence follow the zone.
- 3 **External sources** registered through `GrassCoreIntegration` (AtmosphereCore / WorldCore or your own `IGrassCoreWindSource`).
- 4 A **runtime override** from `GrassCoreRuntimeAPI.SetWind`, blended over time.

How to set it up

- 1 Select the world and open the **Wind** tab.
- 2 Set **Direction Degrees** and **Strength**, then shape the motion with gusts and turbulence.
- 3 Optional: enable **Use Wind Zone** to follow a directional WindZone (empty field = first directional WindZone in the scene).
- 4 Fine-tune each species in its **Wind** section: *Strength*, *Flutter*, *Stiffness*, *Phase Variation*.
- 5 Enable **Always Refresh** in the Scene view to preview the animation in edit mode.

Important Settings

World › Wind

Setting	Default	Description
Enabled	on	Enables wind animation.
Direction Degrees	45	Yaw angle (0 = +Z, 90 = +X).
Strength	0.6	Base strength (0 = calm, 1 = breeze, 3 = storm).
Gust Strength	0.8	Additional strength of gusts.
Gust Scale	40 m	World size of the gust patterns.
Gust Speed	6 m/s	Speed at which gusts travel across the field.
Turbulence / Turbulence Scale	0.25 / 6 m	Small-scale chaotic motion and its size.
Flutter Frequency	6	Frequency of blade flutter.
Transition Time	2 s	Time used to blend towards a new target wind.
Use Wind Zone / Wind Zone	off / –	Follow a directional Unity WindZone.
External Sources	on	Accept wind from AtmosphereCore / WorldCore sources registered via GrassCoreIntegration.

The *Current (Live)* section shows the resolved direction and strength of the current frame.

Species › Wind

Setting	Default	Description
Strength	1	Overall response to wind (0–3).
Flutter	0.35	High-frequency flutter (0–2).
Stiffness	2	Higher values keep the lower part of the blade straighter (1–4).
Phase Variation	1	Random phase offset between blades.

Example Workflow

A storm that builds up and calms down

```
GrassCoreWindState storm = GrassCoreRuntimeAPI.GetWind();
storm.strength = 2.6f;
storm.gustStrength = 2f;
storm.turbulence = 1.2f;
GrassCoreRuntimeAPI.SetWind(storm, blendSeconds: 8f); // blend in over 8 s

// ... later
GrassCoreRuntimeAPI.ClearWindOverride(); // back to the inspector / WindZone wind

// Short gust, e.g. from an explosion or a passing train:
GrassCoreRuntimeAPI.SetWindStrengthMultiplier(2.5f);
```

`GrassCoreRuntimeAPI.SampleWind(position)` returns the wind vector at a position – useful to drive particles, flags or audio with the same wind as the grass.

Performance Notes

- The wind costs almost nothing on the CPU: one wind state per frame, uploaded as shader globals.
- Gusts and turbulence are evaluated per vertex. More blade segments (quality preset) mean more vertices, not more wind cost per blade.
- Use *Debug View = Wind* (World › Debug) to visualise the wind bending.

Common Problems

Problem	Solution
Wind does not move in the Scene view	Enable <i>Always Refresh</i> in the Scene view effects menu.
Inspector changes have no effect	A <i>WindZone</i> (<i>Use Wind Zone</i>), an external source or a runtime override has a higher priority. Call <code>ClearWindOverride()</code> or disable the source.
Grass looks too nervous	Lower <i>Turbulence</i> and the species' <i>Flutter</i> ; raise <i>Stiffness</i> .
All blades move in sync	Raise the species' <i>Phase Variation</i> and <i>Gust Scale</i> .

14

WIND, INTERACTION & EFFECTS

Player & Object Interaction

Characters, animals, vehicles, wheels and projectiles push the grass aside and flatten it – with soft recovery and optional persistent trails. One component, no colliders per blade.

What it does

The **GrassCore Interactor** component makes any object bend the grass around it. Interaction runs in a camera-centred interaction texture on the GPU: interactors are stamped into it as swept capsules (so fast objects leave continuous tracks), the grass recovers over time, and flattened grass can stay down as a **persistent trail**.

The same component can also push underwater plants and lily pads aside ([chapter 23](#)), make ripples on water surfaces ([chapter 24](#)), push bushes aside and shake trees ([chapter 31](#)), and knock snow off the grass ([chapter 19](#)).

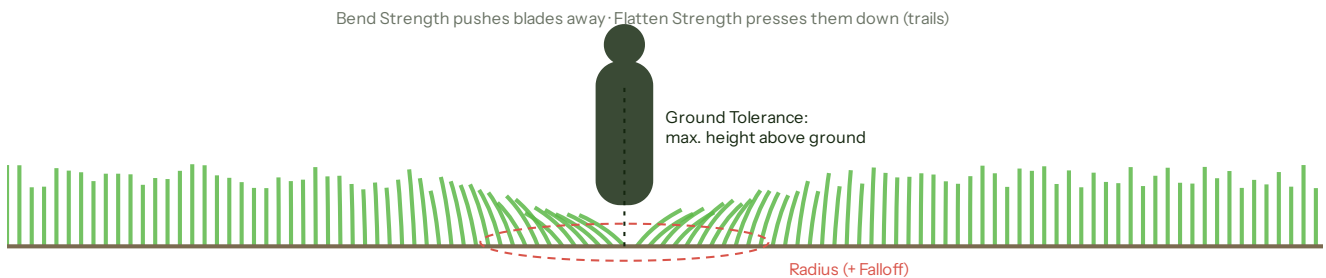


Figure 14.1 – An interactor bends blades away from its position and flattens them. It only acts while it is close to the ground.

How to set it up

- 1 Select your player, NPC, animal or vehicle and run **Tools > GrassCore > Add Interactor to Selection** – or add *GrassCore > Grass Interactor* via Add Component.
- 2 Pick a **Preset**: *Character*, *Animal*, *Vehicle*, *Wheel* or *Projectile*. Choose *Custom* to edit values freely.
- 3 Adjust the **Offset** so the interaction point sits at the feet or wheel contact.
- 4 For vehicles add one interactor per wheel (preset *Wheel*) for realistic tyre tracks.
- 5 Press Play. To preview interaction in edit mode, enable **Simulate In Edit Mode** (World > Interaction).

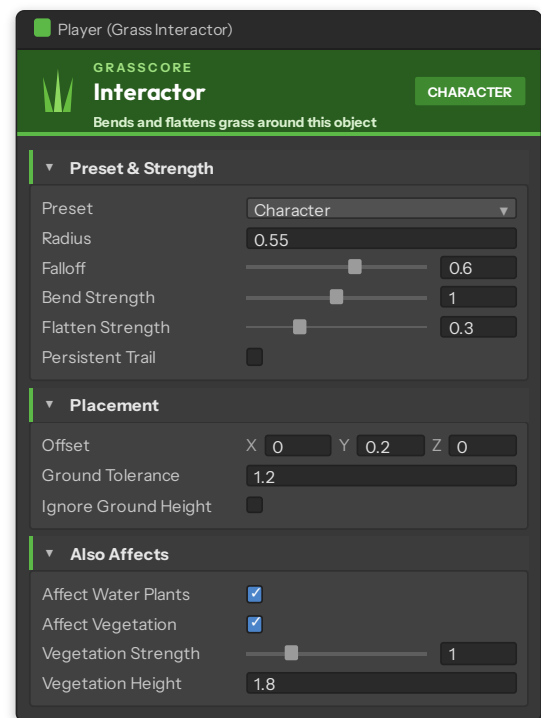


Figure 14.2 – Grass Interactor inspector.

Important Settings

Grass Interactor

Setting	Default	Description
Preset	Character	Applies typical values; <i>Custom</i> keeps your own.
Radius	0.6 m	Radius of influence.
Falloff	0.6	Soft edge of the influence (0 = hard, 1 = fully soft).
Bend Strength	1	How strongly grass is pushed away (0–2).
Flatten Strength	0.35	How strongly grass is flattened (trails).
Persistent Trail	off	Flattened grass stays down until the persistence decays (vehicle tracks, paths).
Offset	0,0,0	Offset from the transform position.
Ground Tolerance	12 m	The interactor only acts while it is at most this high above the ground (also the reach above the position for water plants).
Ignore Ground Height	off	Skip the ground height test (always interact).
Affect Water Plants	on	Also push underwater plants and lily pads.
Affect Vegetation	on	Also push bushes aside and shake trees.
Vegetation Strength / Height	1 / 18 m	Mass / force for trees and bushes (character 1, vehicle 2–3) and the height of the body.

Presets

Preset	Radius	Falloff	Bend	Flatten	Persistent	Ground Tol.	Veg. Strength / Height
Character	0.55	0.6	1	0.3	no	12	1 / 18
Animal	0.8	0.6	0.9	0.25	no	12	0.9 / 11
Vehicle	18	0.35	14	0.9	yes	2.5	2.5 / 17
Wheel	0.45	0.25	12	1	yes	0.9	0.6 / 0.6
Projectile	0.35	0.8	15	0	no	1	15 / 0.35

World › Interaction › Bending & Trails

Setting	Default	Description
Enabled	on	Grass interaction (bending, trails, impulses).
Simulate In Edit Mode	off	Preview interactors without entering Play mode.
Bend Recovery	3 /s	How fast bent grass springs back.
Flatten Recovery	0.25 /s	How fast flattened grass stands up. Low values leave long-lasting trails.
Persistence Decay	0.02 /s	How fast persistent trails fade. 0 = permanent until cleared.

The size and resolution of the interaction area come from the quality preset (*Interaction Range*, *Interaction Resolution*; [chapter 4](#)). Per species, *Bend Response* and *Flatten Response* scale the effect.

Example Workflow

A car leaving tyre tracks

- 1 Add a *Vehicle* interactor to the car body (strong bending around the car) – or skip it for a subtle look.
- 2 Add an empty child at each wheel contact point with a *Wheel* interactor: small radius, full flattening, persistent trail.
- 3 World › Interaction: *Flatten Recovery* 0.15 and *Persistence Decay* 0.01 for tracks that last for minutes.
- 4 From code, stamp a one-off trail (e.g. a sliding body): `GrassCoreRuntimeAPI.AddInteraction(from, to, radius: 0.5f, bend: 1f, flatten: 0.8f, persistent: true)`.

Performance Notes

- Up to **64 interactors** are evaluated per frame; each is a tiny capsule stamp into one texture – the cost does not depend on the number of blades.
- Only blades inside the camera-centred interaction area (*Interaction Range*) react, and only LOD0 and LOD1 blades evaluate interaction.
- Interactors far above the ground (more than *Ground Tolerance*) are skipped.
- *Interaction Resolution* defines the texture detail; higher values give finer footprints at a small GPU cost.

Common Problems

Problem	Solution
Nothing bends	Interaction runs in Play mode (or <i>Simulate In Edit Mode</i>). Check World › Interaction › <i>Enabled</i> and the interactor's <i>Ground Tolerance / Offset</i> .
Grass far away does not react	Only the interaction area around the camera is simulated – raise <i>Interaction Range</i> (Custom quality).
Trails disappear too quickly	Enable <i>Persistent Trail</i> and lower <i>Flatten Recovery / Persistence Decay</i> .
Footprints look blocky	Raise <i>Interaction Resolution</i> or the interactor's <i>Falloff</i> .
Remove all trails (e.g. on level load)	<code>GrassCoreRuntimeAPI.ClearInteraction()</code> .

Explosion Shockwaves

Explosions send an expanding ring through the grass that pushes every blade outward; behind the front the grass rises back up softly. Frame-rate independent, evaluated on the GPU.

What it does

A **shockwave** is a ring that expands from a point to its *Radius* over its *Duration*. Blades at the wave front are pushed away from the centre and optionally flattened; behind the front they recover over the *Recovery Distance*. Shockwaves are evaluated analytically every frame, so they look the same at any frame rate and do not leave lasting marks in the trail texture.

Shockwaves also affect underwater plants, shake trees and bushes, knock snow off and – with *Explosion Splashes* on – make a splash on water surfaces. Up to **32 shockwaves** can run at once; when the limit is reached the oldest or weakest is replaced.

How to set it up

From a prefab

- 1 Add *GrassCore* › *Shockwave Source* to your explosion prefab.
- 2 Keep **Trigger On Enable** on: every time the prefab is spawned or taken from a pool, a shockwave starts at its position.
- 3 Tune the **Shockwave** section.

For testing

GameObject › **GrassCore** › **Shockwave Source** creates a test object. In Play mode press **Trigger Now** in its *Trigger* section. The World › Interaction tab also has a *Test Shockwave in Front of Camera* button.

From code

```
void OnProjectileExploded(Vector3 hitPoint)
{
    GrassCoreRuntimeAPI.TriggerShockwave(hitPoint,
        new GrassCoreShockwaveSettings(radius: 12f, strength: 1.3f, duration: 1.2f, waveWidth: 2f));
}
```

Important Settings

Setting	Default	Description
Radius	14 m	Final radius the wave front reaches.
Strength	12	How strongly the grass is pushed away (1 = strong, 2 = extreme; 0–3).
Duration	13 s	Lifetime of the wave; the front decelerates and the push fades over this time.
Wave Width	2.5 m	Width of the wave front (steepness of the leading edge).
Recovery Distance	5 m	Distance behind the front over which the grass rises back. Larger = slower, softer recovery.
Flatten	0.35	Extra flattening at the wave front (0 = bending only).
Max Height Above Ground	1	Explosions higher than this (as a fraction of the radius) have no effect. 0 disables the check.
Trigger On Enable	on	Shockwave Source: trigger when the object is enabled.

The component also offers `Trigger()` and `TriggerAt(position)` for scripts.

Example Workflow

Grenade with a matching effect on trees and water

- 1 Explosion prefab: *Shockwave Source* with *Radius* 10, *Strength* 15, *Duration* 1.

- 2 Trees and bushes shake automatically when *Interaction* is on (World › Trees). For damage add `GrassCoreRuntimeAPI.DamageTrees(center, radius, damage)` (chapter 32).
- 3 Near water the shockwave makes a splash (World › Surfaces › Water Surfaces › *Explosion Splashes*).
- 4 To burn the grass as well, call `GrassCoreRuntimeAPI.IgniteGrass(center, radius: 4f)` (chapter 18).

Performance Notes

- Shockwaves are a small list evaluated per blade inside the interaction area – no textures are written.
- The cost grows with the number of active waves, not with their radius. Short durations keep the list short.
- Only blades within the camera-centred interaction area react.

Common Problems

Problem	Solution
Explosion in the air has no effect	Lower the explosion or raise <i>Max Height Above Ground</i> (0 disables the check).
Effect too weak far from the centre	Raise <i>Radius</i> and <i>Duration</i> ; the push fades over the lifetime.
Explosions far away have no effect	They must be inside the interaction area around the camera (<i>Interaction Range</i>).
Grass pops back up	Increase <i>Recovery Distance</i> for a softer recovery.

16

WIND, INTERACTION & EFFECTS

Helicopter & Propeller Downwash

Helicopters, drones and other rotors press the grass down and blow it outward – strongest under the rotor, with travelling gusts, swirl and an irregular edge that fades with height.

What it does

The **Downwash Source** turns a rotor hub into a local storm. Under the rotor the grass is pressed down; around it the grass is blown outward in gust packets that travel away from the centre. The effect fades with the rotor's height above the ground, and a banking rotor blows the grass ahead of it. Downwash also pushes water plants and lily pads and knocks snow off.

Up to **16 rotors** are evaluated per frame, analytically in the interaction pass (inside the camera-centred interaction area).

How to set it up

- 1 Add *GrassCore > Downwash Source (Rotor)* to each rotor hub – or create a test object with **GameObject > GrassCore > Downwash Source (Rotor)**.
- 2 Drive **Throttle** (0–1) from your engine or rotor RPM. *Response Time* smooths spin-up and spin-down.
- 3 Keep **Follow Rotor Tilt** on so a banking helicopter blows the grass ahead of it.
- 4 Adjust **Radius**, **Strength** and **Max Height** to the size of the aircraft.

From code – call every frame while the rotor runs:

```
GrassCoreRuntimeAPI.ApplyDownwash(rotor.position, radius: 10f, strength: 1.5f, turbulence: 0.6f, rotorId: 1);  
// or with all settings and the rotor's up axis:  
GrassCoreRuntimeAPI.ApplyDownwash(rotor.position, rotor.up, GrassCoreDownwashSettings.Default, throttle: 1f, rotorId: 1);
```

Important Settings

Setting	Default	Description
Radius	10 m	Radius of the affected area on the ground (grows slightly with height).
Strength	14	How strongly the grass is blown outward (1 = strong, 2+ = violent; 0–4).
Flatten	0.6	How strongly the grass is pressed down, strongest under the rotor.
Turbulence	0.55	Travelling gusts, swirl and a wobbling outline (0 = smooth circle).
Gust Scale / Gust Speed	3 m / 9 m/s	Size of the gust packets and their outward speed.
Max Height	25 m	Height above the ground at which the downwash has faded out completely.
Throttle	1	Rotor power (0 = off, 1 = full).
Response Time	0.6 s	Spin-up / spin-down time.
Follow Rotor Tilt	on	Tilt the wash with the rotor.
Offset	0,0,0	Rotor hub offset in local space.

Example Workflow

Helicopter landing

- 1 Helicopter prefab: one *Downwash Source* on the main rotor hub, *Radius* 12, *Strength* 16, *Max Height* 30.
- 2 Your flight script sets `throttle` from the engine state – on the ground with the engine off it is 0.
- 3 During the landing the grass flattens more and more as the helicopter descends; when the engine stops it springs back.
- 4 A tandem-rotor aircraft gets one source per rotor (each with its own `rotorId` when driven from code).

Performance Notes

- Downwash is evaluated analytically – no textures are written and the cost is independent of the radius.
- Sources with *Throttle* 0 or above *Max Height* have no effect.
- Only grass inside the interaction area around the camera reacts.

Common Problems

Problem	Solution
No effect	Is <i>Throttle</i> above 0 and the rotor below <i>Max Height</i> ? Is World › Interaction <i>Enabled</i> ?
Effect disappears when calling from code	<code>ApplyDownwash</code> applies to the current frame only – call it every frame.
Wash looks like a perfect circle	Raise <i>Turbulence</i> .
Grass blown in the wrong direction	Check the rotor's up axis, or turn off <i>Follow Rotor Tilt</i> .

Mowing, Cut Height & Grass Clippings

Lawn mowers, tractors and string trimmers cut the grass to any height and leave clean, persistent tracks – optionally with GPU grass clippings flying out of the deck.

What it does

A **GrassMower** component cuts the grass under its cutting area while it moves in Play mode. The cut height is stored in a **cut map** per mown cell, so tracks persist while chunks stream in and out and can be saved. Optional **grass clippings** are GPU particles thrown out of the deck, carried by the wind and landing on the ground.

How a cut plant looks depends on the species' **Cut Style**:

Cut Style	Result
Auto	Procedural blades are cut through, custom meshes shrink.
Cut Through	Vertices above the cut are clipped to the cut height: blunt, freshly cut ends.
Shrink	The whole plant is scaled down to the cut height (flowers, clumps and cards keep their shape).

How to set it up

- 1 **GameObject** › **GrassCore** › **Mower (Cutting Tool)** creates a test mower – or select your vehicle and run **Tools** › **GrassCore** › **Add Mower to Selection**.
- 2 Choose a **Preset**: *Lawn Mower*, *Riding Mower*, *Tractor* or *String Trimmer*.
- 3 The object's forward axis (blue) is the driving direction; move the cutting area under the deck with **Offset**.
- 4 Set **Cut Height** (0 = removed, 0.1 = stubble, 0.5 = half, 1 = unchanged).
- 5 Optional: enable **Emit Clippings** in *Grass Clippings*.
- 6 Press Play and drive. The Scene view shows the cutting area and the discharge direction.

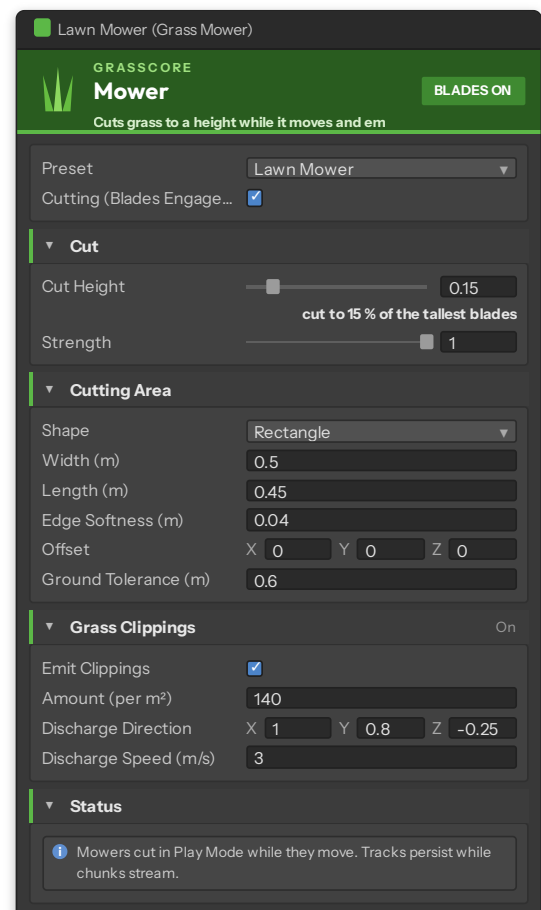


Figure 17.1 – Grass Mower inspector.

Important Settings

Grass Mower

Setting	Default	Description
Preset	Lawn Mower	Applies typical values; edited values switch the preset to <i>Custom</i> .
Cutting (Blades Engaged)	on	Off = drive without mowing.
Cut Height	0.15	Normalised height of the tallest blades (0 = removed, 1 = unchanged).
Strength	1	Strength of the cut.
Shape	Rectangle	<i>Rectangle</i> (Width × Length) or <i>Circle</i> (Radius).
Width / Length / Radius	0.5 / 0.45 / 0.25 m	Size of the cutting area.
Edge Softness	0.04 m	Soft edge of the track; small values give crisp stripes.
Offset	0,0,0	Centre of the cutting area relative to the transform.
Ground Tolerance	0.6 m	Only cut while at most this high above the ground (jumps, lifted decks).
Ignore Ground Height	off	Cut regardless of the height above the ground.

Presets

Preset	Shape / Size	Cut Height	Discharge
Lawn Mower	0.5 × 0.45 m deck	0.15	Side
Riding Mower	11 × 0.7 m deck	0.2	Side
Tractor	2.4 × 1 m mulcher / flail mower	0.2	Rear
String Trimmer	Round head, radius 0.22 m	0.08	Sprayed forward

Grass Clippings (per mower)

Setting	Default	Description
Emit Clippings	off	GPU clippings while cutting.
Amount (per m²)	140	Clippings per m² of fully cut, dense grass.
Discharge Direction / Speed	(1, 0.8, -0.25) / 3 m/s	Direction in local space (x right, y up, z forward) and speed.
Spread / Inherit Velocity	0.7 / 0.4	Random spread and the share of the mower's velocity.
Length / Lifetime	0.02–0.05 m / 16–3.2 s	Size and lifetime (including time on the ground).
Spin	22 rad/s	Spin while flying.
Fresh / Dry Colour	green / straw	Each clipping blends between both.

World › Interaction › Mowing

Setting	Default	Description
Cut Map Resolution	Medium	Low 128² (25 cm at 32 m cells, 16 KB per mown cell), Medium 256² (12.5 cm, 64 KB), High 512² (6 cm, 256 KB).
Max Mown Cells	256	Cells that keep mowing at once; when full, the cell farthest from the camera grows back.
Capacity / Max Spawned per Frame	8192 / 600	Clippings alive at once (GPU ring buffer) and spawned per frame.
Gravity / Air Drag / Wind Influence	7 / 2.6 / 0.6	Physics of the clippings.

Per species: **MowResponse** (0 = never cut, e.g. stones or protected plants) and **Cut Style**.

Example Workflow

Mowing gameplay with save games

```
// A sword swing or a spell cuts a circle:
GrassCoreRuntimeAPI.MowGrass(position, radius: 2f, cutHeight: 0.1f);
GrassCoreRuntimeAPI.MowGrassLine(a, b, width: 1.2f, cutHeight: 0.15f);

// Grass grows back over time:
GrassCoreRuntimeAPI.RegrowMownGrass(position, radius: 5f, amount: 0.05f);

// Save / load the mown state:
GrassCoreMowingSnapshot save = GrassCoreRuntimeAPI.ExportMowing();
GrassCoreRuntimeAPI.ImportMowing(save);
```

After teleporting a mower, call `ResetTrack()` on it so no track is drawn between the old and the new position.

Performance Notes

- **Zero cost when unused:** the mowing system exists only while a cutting mower is active or mown grass / clippings exist.
- A stationary mower dispatches nothing; stamps only cover the texels under the cutting area.
- Memory grows with the number of mown cells (see *Cut Map Resolution*). Lower *Max Mown Cells* on memory-limited platforms.
- Clippings stop simulating when the last one has expired; the buffers are released after a few seconds idle.
- `ExportMowing()` reads the cut maps back from the GPU – call it when saving, not every frame.

Common Problems

Problem	Solution
Mower does not cut	It cuts in Play mode while moving. Check <i>Cutting</i> , <i>Ground Tolerance</i> and the <i>Offset</i> (cutting area on the ground?).
Tracks look blurry	Raise <i>Cut Map Resolution</i> or lower <i>Edge Softness</i> . The inspector warns when the area is small compared to the cut map detail.
A species should not be cut	Set its <i>Mow Response</i> to 0.
Flowers look chopped	Use <i>Cut Style = Shrink</i> for flowers and custom meshes.
Old tracks grow back	<i>Max Mown Cells</i> is full – raise it, or accept that far cells regrow.

18

WIND, INTERACTION & EFFECTS

Fire, Burning & Burned Grass

Set the grass on fire: the flames spread outward, every blade glows, chars and burns down, the charred stubble disappears – and the burnt area stays free of grass.

What it does

A grass fire starts at an ignition point and spreads outward at *Spread Speed* until it fills its area. Every blade passes through the same stages:

- 1 **Burning** – the blade glows with flame and ember colours (HDR, blooms) and flickers.
- 2 **Charring and burning down** – over *Burn Duration* the blade turns black and shrinks to stubble.
- 3 **Removal** – the charred stubble disappears at *Remove Speed*.
- 4 **Burnt ground** – the area stays free of grass as a **persistent runtime edit** that survives chunk streaming and is saved with `ExportGrassEdits()`.

Fires can be circles or spline shapes (an *Area* burns the enclosed area; a *Path* burns a strip along the line). Up to **32 fires** can burn at once. Trees and bushes inside a fire are charred and lose their leaves when *Fire* is enabled for them (World › Trees › *Shared GrassCore Systems* and the profile's *Burnable*).

How to set it up

Burn Source component

- 1 **GameObject › GrassCore › Burn Source** creates a fire object – or add *GrassCore › Burn Source* to a fire prefab. For a spline-shaped fire use **GameObject › GrassCore › Spline Burn Area**.
- 2 Set the **Fire** parameters (radius, intensity, durations, spread).
- 3 **Ignite On Enable** starts the fire when the object is enabled (spawned or pooled prefabs).
- 4 Optional: assign a **Fire Light**; its intensity follows the fire's brightness (*Light Intensity* × glow).
- 5 In Play mode press **Ignite Now** (section *Timeline & Test*). A progress bar shows the fire's state, and the header badge shows **BURNING**.

From code

```
int fireId = GrassCoreRuntimeAPI.IgniteGrass(hitPoint,
    new GrassCoreBurnSettings(radius: 8f, intensity: 3f, burnDuration: 4f, removeSpeed: 0.3f)
    {
        spreadSpeed = 2f, // m/s, 0 = the whole area ignites at once
        edgeNoise = 0.6f // irregular burn scar
    });

if (GrassCoreRuntimeAPI.TryGetFire(fireId, out GrassCoreBurnInfo info))
    fireLight.intensity = 4f * info.glow; // drive lights, audio or VFX
```

Important Settings

Fire (Burn Source / GrassCoreBurnSettings)

Setting	Default	Description
Radius	6 m	Radius of the burnt area (circle fires).
Intensity	3	Brightness of the flames (HDR multiplier, 0–8). Values above 1 bloom.
Burn Duration	4 s	How long each blade burns: flames, charring and burning down.
Remove Speed	0.35 /s	How fast the charred stubble is removed (0.25 = gone after 4 s).
Spread Speed	2.5 m/s	Speed of the fire front. 0 = the whole area ignites at once.

Setting	Default	Description
Edge Softness	0.3	Soft edge of the burnt area.
Edge Noise	0.5	Irregularity of the outline (0 = perfect circle).
Stubble Height	0.25	Height of the charred stubble relative to the blade.
Affected Species	Everything	Species that burn.
Shape / Spline	Circle	Circle (<i>Radius</i>) or a spline area / path; spline fires start at the object's position.

World › Interaction › Fire

Setting	Default	Description
Enabled	on	Enables grass fires.
Flame Colour / Ember Colour	orange / red	Colours of hot flames and embers.
Char Colour	near black	Albedo of charred grass.
Flicker Speed / Amount	1 / 1	Flame flicker (0 = steady glow).

`GrassCoreBurnInfo` (from `TryGetFire`) contains the centre, radius, front radius, elapsed time, lifetime, progress, **glow** (0–1), *burning* and *completed*. `GrassCoreRuntimeAPI.ActiveFireCount` and `CancelAllFires()` complete the API; `World.Fire.BurnCompleted` is raised when a fire has finished.

Example Workflow

A wildfire started by a lightning strike

- On impact: `IgniteGrass(point, new GrassCoreBurnSettings(radius: 25f, intensity: 3.5f, burnDuration: 5f, removeSpeed: 0.2f) { spreadSpeed = 1.2f, edgeNoise = 0.8f })`.
- Each frame: `TryGetFire(id, out info)` → place a fire particle ring at `info.frontRadius`, set the volume of a fire sound from `info.glow`.
- When saving: `GrassCoreRuntimeAPI.ExportGrassEdits()` stores the burnt areas; `ImportGrassEdits(snapshot)` restores them.

Performance Notes

- Only chunks touched by a fire do extra work; the fire state is evaluated for the blades of those chunks.
- After a fire has finished, the burnt ground is an ordinary runtime edit – no ongoing cost.
- The number of simultaneous fires is limited to 32.

Common Problems

Problem	Solution
Fire is disabled on the GrassCore World	Enable World › Interaction › Fire › <i>Enabled</i> .
Nothing burns	Fires run in Play mode. Check <i>Affected Species</i> and that grass is inside the radius.
No glow / bloom	Enable Bloom in your post-processing Volume; raise <i>Intensity</i> .
Burnt areas grow back after changing quality	Runtime edits are cleared when the tile resolution changes; restore them with <code>ImportGrassEdits</code> .
Trees are not charred	Enable <i>Fire</i> in World › Trees › <i>Shared GrassCore Systems</i> and <i>Burnable</i> in the profile.

Seasons & Snow Accumulation

One value moves the whole landscape through the year – fresh spring green, dry summer straw, autumn colours, winter with snow settling on the blade tips. Players and vehicles knock the snow off; it slowly rebuilds.

What it does

The **season** is a single cyclic value: 0 = spring, 0.25 = summer, 0.5 = autumn, 0.75 = winter, 1 = spring again. Each season has a **look** that tints the grass at the root and the tip, mixes a second tip colour in patches and per blade, and defines how much snow lies at that time of year.

Snow settles on the blade tips. The snow line creeps down the blade with *Snow Coverage* and loaded blades sag. With *Snow Follows Season* the snow only appears in winter and melts into spring; without it *Snow Amount* applies all year (eg. a permanently snowy mountain level).

Snow interaction: players, animals, vehicles, shockwaves and rotor downwash knock the snow off inside the interaction area. After *Snow Regrow Delay* it rebuilds over *Snow Regrow Time*.

Trees and bushes follow the season as well: leaves change colour, *Deciduous* profiles drop their leaves in late autumn and regrow them in spring, seasonal flowers bloom in spring and seasonal fruits appear from late summer to autumn ([chapter 30](#)).

How to set it up

- 1 Select the world and open the **Season** tab.
- 2 Drag **Season** or press **Spring**, **Summer**, **Autumn** or **Winter**. The year bar shows the colours and snow over the year. In Play mode the buttons blend over 3 seconds.
- 3 Open **Season Looks** to edit the four looks. *Strength* 0 keeps the material colours (the default spring look).
- 4 Set **Snow Amount**, **Snow Coverage** and **Snow Colour** in *Snow*; tune the interaction with *Snow Sensitivity*, *Snow Regrow Delay* and *Snow Regrow Time*.
- 5 Per species: **Season Response** (0 = evergreen), **Snow Response** (0 = never snowy, eg. reeds in water).

```
GrassCoreRuntimeAPI.SetSeason(GrassCoreSeason.Winter, blendSeconds: 10f);
GrassCoreRuntimeAPI.SetSnowAmount(0.6f, blendSeconds: 5f);
float snow = GrassCoreRuntimeAPI.GetEffectiveSnow(); // snow after season scaling (0..1)
```

Important Settings

Setting	Default	Description
Enabled	on	Seasonal colours and snow.
Season	0	Time of the year (cyclic): 0 spring, 0.25 summer, 0.5 autumn, 0.75 winter.
Spring / Summer / Autumn / Winter look	–	<i>Root Colour</i> , <i>Tip Colour</i> , <i>Tip Variation</i> (second tip colour in patches), <i>Strength</i> (how much the look replaces the material colours), <i>Snow</i> (snow at this time of year).
Snow Amount	1	Amount of snow on the grass (0–1).
Snow Follows Season	on	Scale <i>Snow Amount</i> by the season. Off = <i>Snow Amount</i> applies all year.
Snow Coverage	0.45	How far down the blade snow can settle (fraction from the tip).
Snow Colour	white-blue	Colour of the snow.
Snow Sensitivity	1	How easily interaction knocks the snow off (0–3).
Snow Regrow Delay	6 s	Time the snow stays off after a disturbance.
Snow Regrow Time	20 s	Time the snow needs to rebuild completely.

`SetSeason(value, blendSeconds, forward)` blends forward through the year by default (winter → spring), or backwards with `forward = false`.

Example Workflow

A day-to-day season cycle in a survival game

- 1 Keep the default looks and let your calendar call `SetSeason(day / 365f, blendSeconds: 30f)` once per in-game day.
- 2 Reeds and water plants keep their colour: set their *Season Response* and *Snow Response* to 0.
- 3 For a blizzard event: `SetSnowAmount(1f, 20f)` and a stronger wind ([chapter 13](#)); afterwards blend back to the season's snow.

Performance Notes

- Seasons and snow are evaluated per vertex from a few global values – no extra passes, no extra memory per blade.
- Snow interaction uses the existing interaction area ([chapter 14](#)).
- Changing the season at runtime causes no tile rebuilds.

Common Problems

Problem	Solution
No snow in winter	Check <i>Snow Amount</i> and the <i>Winter</i> look's <i>Snow</i> . The <i>Snow on the grass</i> stat in the Season tab shows the effective value.
Interaction is disabled: snow is not knocked off	Enable World › Interaction › <i>Enabled</i> .
Some plants should stay green	Set their species' <i>Season Response</i> to 0.
Seasonal colours too strong	Lower the look's <i>Strength</i> .

20

WIND, INTERACTION & EFFECTS

Dynamic Color Zones

Paint the world with gameplay: dry patches, swamps, poisoned ground that glows, burnt areas, snow or magic zones – with soft, noisy edges, created in the editor or at runtime.

What it does

A **ColorZone** changes the look of the grass inside a circle or spline shape: colour, glow, wetness, snow and blade height. Overlapping zones blend – colours by *Priority*, glow adds up. Zones can move, fade in and fade out, which makes them ideal for gameplay (a spreading poison cloud, a magic circle under the player, a burnt crater after an explosion).

All zones near the camera are blended into small textures once when something changes, so the cost per blade is constant – no matter how many zones exist. Up to **64 zones** near the camera are blended.

Preset	Look
Dry Patch	Straw colour, slightly shorter grass.
Swamp	Dark, wet and glossy grass, noisy edge.
Poison	Purple tint with a green glow, shorter grass.
Burnt	Nearly black, burnt down to 35 % height.
Snow	Snow on the blade tips, independent of the season.
Magic	Blue tint with a strong blue glow.

How to set it up

In the editor

- 1 **GameObject** › **GrassCore** › **Color Zone** (circle) or **GameObject** › **GrassCore** › **Spline Color Zone** (area or path).
- 2 Pick a **Preset** – or *Custom* to edit the look freely.
- 3 Shape it in **Shape** (radius or spline, *Spline Falloff*, optional *Use Transform Scale*) and **Look**.
- 4 Move the transform to move the zone; disable the component to remove it. *Fade In Time* fades it in when enabled.

At runtime

```
int zone = GrassCoreRuntimeAPI.CreateColorZone(position, GrassCoreColorZonePreset.Poison, radius: 6f, fadeInSeconds: 1f);
GrassCoreRuntimeAPI.MoveColorZone(zone, newPosition);
GrassCoreRuntimeAPI.RemoveColorZone(zone, fadeOutSeconds: 3f);
```

`ModifyColorZone(id, settings)` changes a zone's look, `TryGetColorZone(id, ...)` reads it back. The Season tab shows the number of zones near the camera and has **Create Color Zone** and **Clear Zones Created From Code** buttons.

Important Settings

Zone look (GrassCoreColorZoneSettings)

Setting	Description
Radius	Radius (m) of the zone
Falloff	Soft edge (0 = hard edge, 1 = fades from the centre).
Edge Noise	Irregular outline (0 = perfect circle).
Intensity	Overall strength – fade a zone in or out with it.
Color	Grass colour inside the zone; alpha = how much it replaces the grass colour.
Emission	Glow (poison, magic). Black = no glow.

Setting	Description
Wetness	Darker, glossier look (swamps, puddles).
Snow	Snow on the blade tips, independent of the season.
Height	Blade height inside the zone (1 = unchanged, lower = stunted or burnt down).
Priority	Zones with a higher priority are blended over lower ones (colour).

Color Zone component and world settings

Setting	Default	Description
Shape	Circle	Circle (<i>Radius</i>) or Spline (area or path).
Spline Falloff	3 m	Spline: soft edge inside the outline (areas) or beyond the width (paths).
Use Transform Scale	off	Scale the radius with the transform's largest horizontal scale.
Fade In Time	0 s	Fade-in time when enabled.
World › Season › Color Zones › Enabled	on	Enables colour zones.
Range	300 m	Distance around the camera in which zones are shown (capped by the render distance).
Resolution	512	Zone texture resolution (256 / 512 / 1024). Texel size = $2 \times \text{range} / \text{resolution}$.

Per species, **Color Zone Response** (0 = ignores zones) controls the effect; trees and bushes use their profile's *Color Zone Response*.

Example Workflow

Aspreading poison cloud

- 1 When the cloud spawns: `id = CreateColorZone(pos, GrassCoreColorZonePreset.Poison, 2f, 0.5f)`.
- 2 Each second, read the settings with `TryGetColorZone`, increase `radius` and call `ModifyColorZone(id, settings)`.
- 3 When the cloud is gone: `RemoveColorZone(id, fadeOutSeconds: 5f)` – the grass slowly recovers.

Performance Notes

- The cost per blade is one texture lookup at the root, independent of the number of zones.
- The zone textures are rebuilt only when something changes – not every frame.
- A smaller *Range* or *Resolution* reduces memory; a larger resolution gives crisper small zones.

Common Problems

Problem	Solution
Zone not visible	Is it within <i>Range</i> of the camera? Is <i>Intensity</i> > 0 and the colour's alpha > 0?
Small zones look blurry	Raise <i>Resolution</i> or lower <i>Range</i> .
Glow not visible	Enable Bloom; raise <i>Emission</i> .
A species ignores zones	Its <i>Color Zone Response</i> is 0.

Bioluminescent & Magic Grass

Fantasy meadows that glow at night: emissive blade tips with a soft pulse breathing across the field, glowing footsteps, auras around the player and rings of light travelling through the grass.

What it does

Glow has two layers:

- 1 **Material glow** – enable *Bioluminescence* › *Glow* in a GrassCore/Grass material. The blade tips become emissive with their own colour and a soft pulse whose phase drifts across the field, so light "breathes" through the meadow. Each blade varies slightly.
- 2 **Reactive glow** – the grass lights up around **Glow Sources** (player auras, orbs, lanterns), leaves a glowing **trail** that spreads and fades, and reacts to **light waves** – rings of light travelling outward.

The sample species **Glowgrass** ([Samples/Species/Glowgrass](#)) is ready to use.

How to set it up

- 1 Use *Glowgrass*, or enable **Glow** in your species' material (section *Bioluminescence*).
- 2 Set **Glow Colour**, **Glow Intensity** and **Glowing Tip Length**; shape the pulse with *Pulse Speed*, *Pulse Amount* and *Pulse Scale (m)*.
- 3 For reactive glow add *GrassCore* › *Glow Source* to the player (or run **Tools** › **GrassCore** › **Add Glow Source to Selection**). Set the *Reactive Glow* and *Reactive Glow Colour* in the material.
- 4 Enable **Bloom** in your URP post-processing Volume for the halo.
- 5 In Play mode press **Emit Light Wave** on the Glow Source, or the *Test Light Wave from Camera* button in World › Interaction.

```
// Glowing area for the current frame (call every frame), with an afterglow trail:
GrassCoreRuntimeAPI.AddGlow(player.position, radius: 3f, intensity: 1.5f, trail: 0.7f);

// A ring of light travelling outward:
GrassCoreRuntimeAPI.EmitGlowWave(position, range: 20f, intensity: 1.5f, speed: 7f, width: 2.5f);
```

Important Settings

Material (GrassCore/Grass › Bioluminescence)

Property	Default	Description
Glow	off	Enables the emissive tips.
Glow Colour (HDR)	cyan-green	Colour of the glowing tips.
Glow Intensity	2	Brightness (0–20). 0 = grass only glows around glow sources.
Glowing Tip Length	0.35	Part of the blade that glows.
Pulse Speed / Amount / Scale (m)	0.25 / 0.5 / 12	Soft pulse drifting across the field.
Per-Blade Variation	0.5	Random brightness per blade.
Reactive Glow Colour (HDR) / Reactive Glow	blue / 3	Colour and strength of the response to glow sources, trails and waves.

Glow Source

Setting	Default	Description
Radius / Intensity / Falloff	3 m / 15 / 0.7	Glowing area around the source.
Trail	0.6	Afterglow left behind (0 = none).
Speed Response / Full Speed	0 / 5 m/s	1 = glow only while moving (brighter when faster).
Wave Interval	0 s	Emit a light wave every N seconds (0 = off).

Setting	Default	Description
Wave	20 m, 7 m/s, 2.5 m, 15	Range, speed, ring width and intensity of the light waves.
Max Height Above Ground	3 m	Only glow when this close to the ground (0 = no limit).
Offset	0,0,0	Offset from the transform.

World › Interaction › Bioluminescent Glow

Setting	Default	Description
Enabled	on	Reactive glow (sources, trails, waves). The material's own glow and pulse work regardless.
Afterglow Decay	0.5 /s	How fast the afterglow fades (0 = never).
Spread	15 m ² /s	How fast the afterglow spreads into the surrounding grass.
Max Glow	4	Upper limit where many sources overlap.

Example Workflow

A night meadow that reacts to the player

- 1 Species: *Glowgrass* with *Glow Intensity* 0 – the meadow is dark until something touches it.
- 2 Player: *Glow Source* with *Radius* 2.5, *Trail* 0.8, *Speed Response* 0.7 – walking leaves a glowing path.
- 3 Every 6 seconds a light wave: *Wave Interval* = 6.
- 4 A magic spell: `EmitGlowWave(target, range: 30f)` plus a *Magic* colour zone (chapter 20).

Performance Notes

- Material glow is a shader keyword: only materials with *Glow* enabled pay its small, constant cost.
- Reactive glow uses the interaction area; up to 32 glow sources and 32 light waves are evaluated per frame.
- Bloom is a post-processing cost of your camera, not of GrassCore.

Common Problems

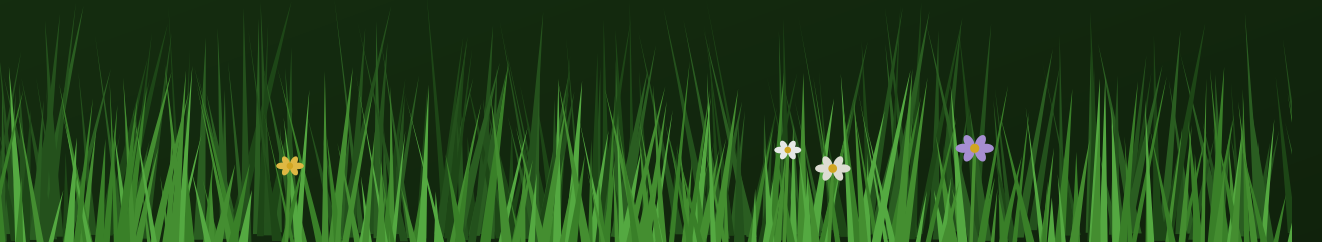
Problem	Solution
No species material has Glow enabled	Enable <i>Glow</i> in the material (GrassCore/Grass › Bioluminescence) or use the <i>Glowgrass</i> sample.
No halo around the glow	Enable Bloom in the Volume; HDR must be enabled on the URP asset.
Glow source has no effect	Is it within <i>Max Height Above Ground</i> ? Is World › Interaction › Glow <i>Enabled</i> ? Is <i>Reactive Glow</i> > 0 in the material?
Trails never fade	<i>Afterglow Decay</i> is 0.

PART 4

Water

Lakes, rivers and seas that remove grass under water, wet the shores, grow underwater plants and render interactive water surfaces.

22	Water Volumes, Lakes & Rivers	59
23	Underwater Vegetation & River Current	62
24	Interactive Water – Ripples, Wakes & V-Waves	65



22

WATER

Water Volumes, Lakes & Rivers

Lakes, ponds, rivers and sea levels that remove the grass under water, create a wet band along the shore and – optionally – render a full water surface with reflections, refraction, foam and river flow.

What it does

A **GrassCore Water Volume** describes water in the scene:

Shape	Use
Box	A flat rectangle at the object's height, or an <i>Infinite</i> sea level.
Spline › Lake / Area	A free-form lake or pond drawn along the shore, flat surface at the <i>Water Level</i> .
Spline › River	A river drawn along its bed from source to mouth; the surface follows the spline plus <i>River Depth</i> . Start → End is the flow direction.

Every water volume removes grass under the water and creates a **wet band** along the shore (width: World › Surfaces › *Shoreline Wetness Width*). Species react with their *Water Clearance*, *Shoreline Width / Density* and *Aquatic* settings ([chapter 6](#)). Two optional features are off by default and free until you enable them:

- **Water Surface** – rendered water with reflections, refraction, depth colour, foam, shore waves, wind waves on lakes and flowing rivers; interactive waves ([chapter 24](#)).
- **Underwater Vegetation** – sea grass, kelp, reeds and lily pads ([chapter 23](#)).

How to set it up

- 1 Choose **GameObject** › **GrassCore** › **Water Volume** (box), **GameObject** › **GrassCore** › **Water** › **Spline Lake** or **GameObject** › **GrassCore** › **Water** › **River**. Lakes and rivers created from the menu already have their surface enabled.
- 2 **Lake**: draw the shore with *Edit Points in Scene View*, then set the **Water Level (Y)** – or press **Level from Shoreline Points**.
- 3 **River**: draw the river along its bed from the source to the mouth (use **Reverse** to flip the direction). Set **River Depth**. River points have their own height: select a point and use the Move gizmo (green arrow = up / down); *Height Above Ground*, *Snap Point to Ground* and *Smooth Heights* help. Arrows in the Scene view show the current.
- 4 Enable **Depth Texture** and **Opaque Texture** on the URP asset for the water surface (the inspector offers a button).

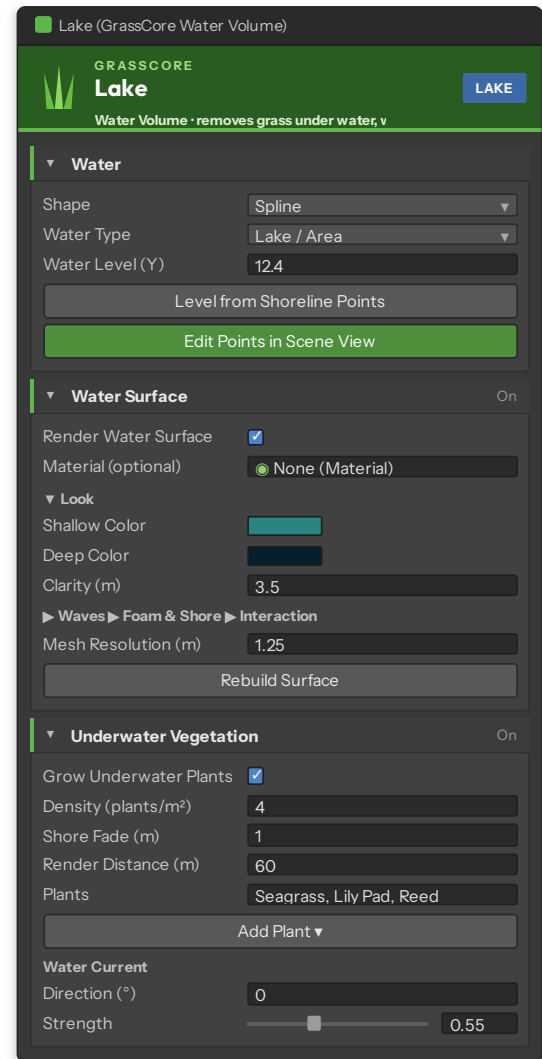


Figure 22.1 – Water Volume inspector (lake).

Important Settings

Water

Setting	Default	Description
Shape	Box	Box or Spline.
Size / Infinite	100 × 100 m / off	Box: size, or an infinite sea level.
Water Type	Lake / Area	Spline: <i>Lake / Area</i> (flat surface) or <i>River</i> (line with a width).
River Depth	0.8 m	River: height of the surface above the spline points (the points lie on the river bed).
Water Level (Y)	object Y	Height of the flat surface (this object's Y position).

Water Surface

Setting	Default	Description
Render Water Surface	off	Off = no meshes, textures, dispatches or draws.
Material (optional)	–	A custom material uses its own values; the settings it overrides are ignored.
Look › Shallow / Deep Color	teal / dark blue	Colour of shallow water and of deep water (also the absorption).
Clarity	3.5 m	How deep you can see before the water takes the deep colour.
Smoothness / Reflection Strength / Refraction	0.94 / 1 / 0.35	Reflection sharpness, strength and distortion.

Setting	Default	Description
Waves › Wave Strength / Scale / Wind Response	0.5 / 5 m / 1	Small surface waves; the GrassCore wind roughens lakes.
Foam & Shore › Foam Intensity, Foam Color, Shore Foam Width, Shore Waves, Edge Softness	1, white, 0.5, 0.35, 0.35	Foam along shores and obstacles, waves rolling towards the shore, soft shoreline blending.
Interaction › Interactive Waves / Wave Height	– / 1	Waves from interactors on this surface (chapter 24).
River Flow	14 m/s ...	Flow speed, downhill and narrows acceleration, bend and shallow turbulence, base turbulence, river foam, and curves for speed, turbulence and foam along the river.
Mesh Resolution	125 m	Vertex spacing of the surface mesh.

Rebuild Surface rebuilds the mesh after sculpting the terrain under the water.

World › Surfaces › Water Surfaces

Setting	Default	Description
Enabled (Master)	on	Master switch for all water surfaces.
Reflections	Screen Space	<i>Screen Space</i> (SSR with probe fallback) or <i>Probes Only</i> (cheapest).
Layer	4	Rendering layer of the water surfaces.
Interactive Waves	on	Ripples, wakes and boat waves (only simulated while something touches the water). See chapter 24 .

Example Workflow

A mountain river flowing into a lake

- 1 Create a *Spline Lake*, draw the shore and press *Level from Shoreline Points*.
- 2 Create a *River*; draw it from the spring down to the lake. Select the last point and lower it to the lake level with the Move gizmo; press *Smooth Heights*.
- 3 River Flow: raise *Slope Acceleration* so the steep part rushes, *Shallow Turbulence* for white water over rocks; use *Speed Along River* to slow it down near the mouth.
- 4 Add *Shore Reeds* to the world – their *Shoreline Density* = 2 makes them grow along both shores.

Performance Notes

- Grass removal and the wet shore band are part of the tile build – no per-frame cost.
- Water surfaces cost one draw per visible surface; meshes are rebuilt only when the shape changes. *Reflections = Probes Only* is the cheapest option.
- Very large lakes are coarsened automatically; a larger *Mesh Resolution* reduces vertices further.
- Up to 8 spline water shapes are evaluated per chunk tile.

Common Problems

Problem	Solution
Depth Texture / Opaque Texture is disabled on the URP asset	Press Enable Depth & Opaque Texture on the URP Asset . Without them water has no refraction, depth colour or soft shores.
Infinite box volumes have no surface mesh	Use a finite box or a spline lake for a visible surface.
Water surfaces are switched off on the GrassCore World	Enable World › Surfaces › Water Surfaces › <i>Enabled (Master)</i> .
River flows the wrong way	Press Reverse – the flow goes from the first to the last point.
Grass still grows at the shoreline	Raise the species' <i>Water Clearance</i> or lower its <i>Shoreline Density</i> .

23

WATER

Underwater Vegetation & River Current

Sea grass, water weeds, kelp forests, reeds and floating lily pads grow in their own depth zones and sway in the water current. Swimmers and boats push them aside.

What it does

Every water volume can grow **underwater plants**. A plant list defines which plants grow; each plant grows only inside its own **water depth range**, so plants form natural zones by themselves – reeds in the shallows, lily pads floating, sea grass in the middle, kelp in deep water.

Every water plant grows only inside its own Depth Range – zones form automatically.

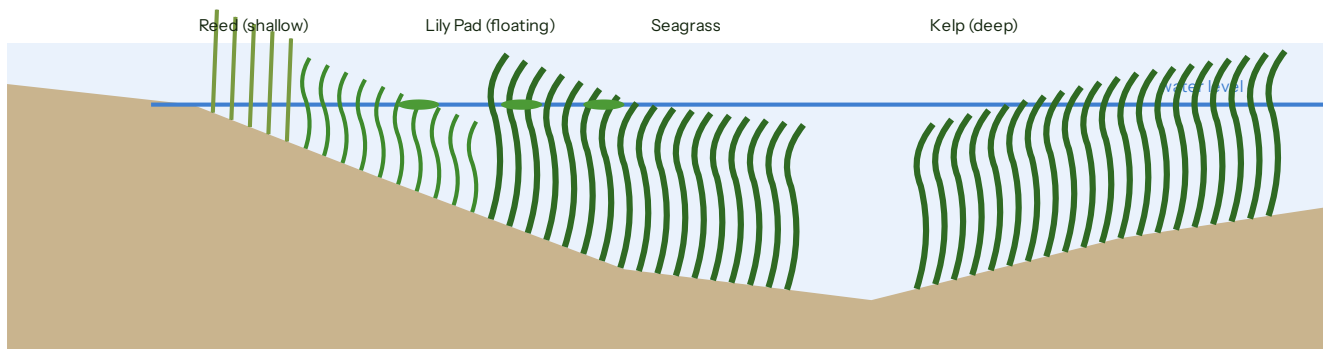


Figure 23.1 – Depth zones form automatically from the plants' Depth Ranges.

Plants sway in the **water current**: lakes and boxes use a direction, rivers take the direction from the spline (start → end) and are stronger in the middle. Interactors push plants aside ([chapter 14](#)), explosions and rotor downwash affect them too, and they drift back over *Interaction Recovery*.

Underwater vegetation is a separate rendering path. It exists only while at least one active water volume grows plants – otherwise there are no buffers, dispatches, draws or updates. The terrain grass is not affected either way.

How to set it up

- 1 Select a water volume and open **Underwater Vegetation**. Enable **Grow Underwater Plants**.
- 2 Press **Add Plant** to add an existing water plant or create a new one from a preset (*Seagrass*, *Water Weed*, *Kelp*, *Reed*, *Lily Pad*). Ready-made assets are in [Samples/WaterPlants](#).
- 3 Set the plants' **Weight** (share among the plants that fit a depth). Without plants, the *Default Plant* settings of the volume are used.
- 4 Set **Density**, **Shore Fade** and **Render Distance**.
- 5 Configure the **Water Current**. For rivers keep **Follow River** on.
- 6 For own plants: **Assets** > **Create** > **GrassCore** > **Water Plant** – built-in ribbon, built-in lily pad or your own mesh (pivot at the root, +Y up), optional material with shader **GrassCore/Underwater**.

Important Settings

Underwater Vegetation (per water volume)

Setting	Default	Description
Grow Underwater Plants	off	Off = zero cost.
Density (plants/m²)	4	Plants per m² close to the camera, shared by all plants of the list.
Shore Fade (m)	1	Plants thin out and shrink towards the water's edge.

Setting	Default	Description
Render Distance (m)	60	Distance up to which plants are drawn (density thins out towards it).
Interaction	on	Plants react to interactors, explosions and downwash.
Plants	–	Up to 8 water plants with weights per volume.

Water Current

Setting	Default	Description
Follow River	on	Rivers: the current follows the spline, curves included.
Direction (°)	0	Lakes, boxes and rivers without <i>Follow River</i> (0 = +Z / north, 90 = +X / east).
Strength	0.55	How strongly the current bends the plants (0 = upright, 1 = almost flat).
Speed	0.6	Speed of the sway waves travelling downstream.
Turbulence	0.35	Irregular sideways movement.
Influence	1	Overall influence of the current (0 = no movement).
Centre Boost	0.5	Rivers: how much faster the current is in the middle than at the banks.

Water Plant asset

Setting	Description
Shape	<i>Ribbon</i> (built-in leaf), <i>Lily Pad</i> (floating leaf, size = Width Range) or <i>Custom Mesh</i> .
Placement / Orientation	<i>Bed</i> (rooted, swaying) or <i>Floating</i> (bobbing); <i>Across Current</i> (ribbons stream flat) or <i>Random</i> .
Height / Width / Water Depth Range	Size and the depth range in which the plant grows.
Stay Below Surface	Keep the plant under the water (off for reeds).
Root / Tip Colour, Colour Variation	Colours (lily pads: centre / rim).
Bend / Sway / Interaction Response	Reaction to the current, swaying and interaction (0–2).
Cast Shadows	Usually off under water; useful for reeds and lily pads.

Preset	Height	Water depth	Notes
Seagrass	0.3–0.9 m	0.2–5 m	Ribbons across the current.
Water Weed	0.4–14 m	0.4–6 m	Thin and very soft, random orientation.
Kelp	2.5–6 m	3–30 m	Wide, slow, brown-yellow.
Reed	1.2–2.2 m	0–0.7 m	Grows above the surface, stiff, casts shadows.
Lily Pad	floating, 0.25–0.55 m wide	0.3–2.5 m	Floats and bobs on the surface.

World › Surfaces › Underwater Vegetation

Setting	Default	Description
Enabled (Master)	on	Master switch (each volume enables its plants separately).
Material	–	Optional shared material (GrassCore/Underwater).
Max Instances	131072	Plants drawn per camera.
Interaction Recovery	12 /s	How fast pushed plants drift back. Lower = longer lanes behind boats.

Example Workflow

A pond with zones

- 1 Spline lake with *Grow Underwater Plants* on.
- 2 Add *Reed*, *Lily Pad*, *Seagrass* and *Water Weed*. The depth ranges sort them automatically from the shore to the deep middle.
- 3 Weights: *Seagrass* 1, *Water Weed* 0.4 – both fit the middle depths and share it.

- 4 Add a *Grass Interactor* to the boat (preset *Vehicle*, pivot near the waterline): it pushes lily pads aside and leaves a lane that closes slowly.

Performance Notes

- Zero cost when no volume grows plants.
- One draw call per water plant type; up to 16 plant types and 32 water volumes are drawn at once.
- *Render Distance* and *Density* are the main cost controls; *Max Instances* caps the total.
- Plant interaction exists only while the world's interaction runs and a volume has *Interaction* on.

Common Problems

Problem	Solution
<i>Underwater vegetation is switched off on the GrassCore World</i>	Enable World › Surfaces › Underwater Vegetation › <i>Enabled (Master)</i> .
No plants appear	Is the water deep enough for the plants' <i>Water Depth Range</i> ? Check <i>Render Distance</i> .
<i>The material must use the shader GrassCore/Underwater</i>	Assign a material with that shader, or leave the material empty.
Boat does not push plants	Add a <i>Grass Interactor</i> with <i>Affect Water Plants</i> on; it reaches from <i>Radius</i> below to <i>Ground Tolerance</i> above its position.
<i>Interaction is disabled on the GrassCore World, so water plants are not pushed</i>	Enable World › Interaction.

Interactive Water – Ripples, Wakes & V-Waves

Swimmers, animals, falling objects and boats make ripples, splashes and wakes on GrassCore water surfaces. Boats push V-shaped bow waves and leave a foamy, turbulent stern wake.

What it does

When *Interactive Waves* is on, GrassCore simulates a wave field around the camera. **Water Interactors** add waves where they touch the water: ripples while standing, waves and a fading wake while moving, a splash when something falls in. Boats are modelled as a pointed hull: they create **V-shaped bow waves** (clearly visible when the boat is faster than the *Wave Speed*), side displacement waves and a foamy, turbulent **stern wake**. Rivers carry the waves downstream.

GrassCore Interactors (the grass component) also make ripples when they are in the water, and explosions near the water make a splash. The simulation only runs while something touches the water – an untouched lake costs nothing.

How to set it up

- 1 Make sure the water volume has **Render Water Surface** on and the surface's **Interactive Waves** enabled (section *Interaction*).
- 2 Add *GrassCore > Water Interactor* to swimmers, animals, physics objects and boats.
- 3 Pick a **Preset**: *Character*, *Animal*, *Physics Object* (uses the Rigidbody's velocity and mass), *Boat* or *Custom*.
- 4 Boats: set **Hull Length** and **Half Beam**; the **Bottom Offset** is the keel relative to the transform.
- 5 Press Play – or enable *Simulate In Edit Mode* (World > Surfaces > Water Surfaces).

```
GrassCoreRuntimeAPI.AddWaterSplash(position, radius: 0.6f, strength: 1f); // e.g. a stone hitting the water
```

Important Settings

Water Interactor

Setting	Default	Description
Preset	Character	Values come from the preset; choose <i>Custom</i> to edit them.
Radius / Half Beam	0.35 m	Radius of the part in the water; boats: half the hull width.
Hull Length	0	Boats: hull length along the forward axis (0 = round object).
Height	18 m	Vertical size from the bottom point upwards.
Bottom Offset	0,0,0	Bottom point (feet, keel) relative to the transform.
Strength	1	Wave strength.
Foam	0.4	Foam left in the wake.
Bow Wave / Stern Turbulence	1 / 0.6	Boats: V-shaped bow wave and propeller wash.
Use Rigidbody	on	Velocity and mass from an attached Rigidbody (heavier = larger waves).

Preset	Radius	Length	Height	Strength	Foam
Character	0.35	0	18	1	0.35
Animal	0.45	0	13	1	0.35
Physics Object	0.5	0	1	12	0.5
Boat	11 (half beam)	5	12	14	1

In Play mode the inspector shows how much of the object is submerged (*Submerged*, also `Submersion` / `IsInWater` from code).

World › Surfaces › Water Surfaces

Setting	Default	Description
Interactive Waves	on	Enables the wave simulation.
Simulate In Edit Mode	off	Run the simulation outside Play mode.
Range (m) / Resolution	96 m / 512	Size and resolution of the simulated area around the camera (256 / 512 / 1024).
Wave Speed	2.2 m/s	Boats faster than this leave a V-shaped wake.
Wave Damping	0.3 /s	How quickly waves die out.
Wake Fade	0.18 /s	How quickly wakes and foam trails fade (lower = longer trails).
Grass Interactors Make Ripples	on	GrassCore Interactors in the water also make waves.
Explosion Splashes	on	Shockwaves on or near the water make a splash.

Example Workflow

A speedboat crossing a lake

- 1 Boat with a Rigidbody: *Water Interactor*, preset *Boat*, *Hull Length* 6, *Half Beam* 12, *Bottom Offset* at the keel.
- 2 World › Water Surfaces: *Wave Speed* 2.2 – at full speed the boat is much faster and draws a clear V wake.
- 3 *Wake Fade* 0.1 for a long foam trail behind the boat.
- 4 Add a *Grass Interactor* (preset *Vehicle*) to push lily pads aside ([chapter 23](#)).

Performance Notes

- The simulation runs at a fixed rate only while something touches the water; afterwards dispatches stop.
- *Resolution* is the main cost: 512 is a good default, 1024 for close-up water gameplay.
- *Range* defines the simulated area – waves outside it are not simulated.

Common Problems

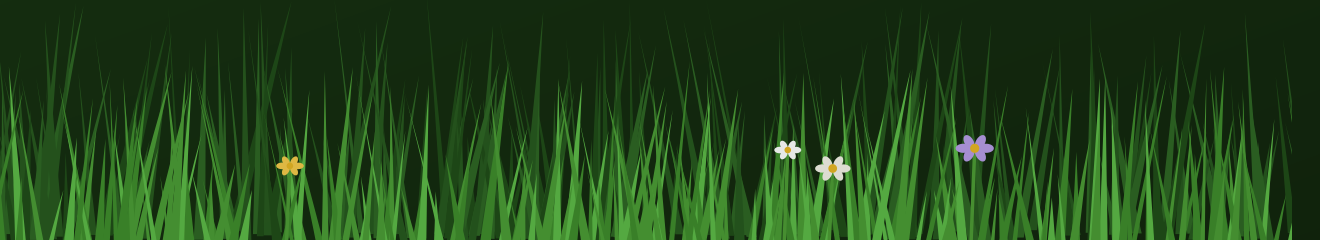
Problem	Solution
No waves	The surface must be rendered with <i>Interactive Waves</i> on, and the simulation runs in Play mode (or <i>Simulate In Edit Mode</i>).
Interactive waves are switched off on the GrassCore World	Enable World › Surfaces › Water Surfaces › <i>Interactive Waves</i> .
Boat makes no V wake	The boat must be faster than <i>Wave Speed</i> ; lower <i>Wave Speed</i> or raise <i>Bow Wave</i> .
Waves look blocky	Raise <i>Resolution</i> or lower <i>Range</i> .
Values cannot be edited	They come from the preset – choose <i>Custom</i> .

PART 5

Trees & Bushes

The optional tree and bush modules: procedural and custom plants, LODs and billboards, placement, wind, flowers and fruits, interaction, colliders and destroyable trees.

25	Trees & Bushes – Overview	68
26	Procedural Trees & Bushes	70
27	Custom Tree & Bush Meshes	73
28	Tree & Bush LODs and Billboards	75
29	Manual Placement & Vegetation Areas	77
30	Tree & Bush Wind, Flowers & Fruits	80
31	Tree & Bush Interaction and Optional Colliders	82
32	Destroyable Trees, Tree Damage & Falling Trees	84



25

TREES & BUSHES

Trees & Bushes – Overview

Optional tree and bush modules rendered by the GrassCore World – GPU-culled, with LODs and billboards, wind, seasons, interaction and destruction. They are never scattered automatically and cost nothing until you place them.

What it does

Trees and bushes are separate from the grass pipeline: they have their own components, profiles, compute shader, shaders and draw calls, but they share the GrassCore systems – wind, seasons, snow, fire, colour zones and interaction.

Building block	Purpose	Chapter
Vegetation Profile (asset)	One tree or bush type: procedural parameters or custom meshes, materials, wind, LODs, flowers, fruits, interaction, colliders, destruction.	chapter 26 , chapter 27
Tree & Bush Instances (Manual)	Place, move, rotate, scale and erase single plants in the Scene view.	chapter 29
Vegetation Area	Generate an exact number of plants inside a box, ellipse or spline – deterministic, seeded.	chapter 29
World › Trees tab	Module switches, LOD bias, distance, shadows, billboards, shared systems, features and status.	this chapter

Trees and bushes are **never placed automatically** – only where you place them by hand or define an area. With no instances, or with both modules switched off, nothing runs: no buffers, no dispatches, no draws.

How to set it up

- 1 Choose **GameObject › GrassCore › Vegetation › Tree & Bush Instances (Manual)**. It starts with the default tree and bush in its palette.
- 2 In its inspector select **Place**, pick a profile and click on the ground in the Scene view.
- 3 For forests use **GameObject › GrassCore › Vegetation › Vegetation Area (Box)** or (*Spline*), add entries and press **Generate**.
- 4 Pick profiles from **Samples/Vegetation** (38 ready-made trees and bushes) or create your own with **Assets › Create › GrassCore › Tree Profile / Bush Profile**.
- 5 Tune the global behaviour in **World › Trees**.

Important Settings

World › Trees

Section / Setting	Default	Description
Modules › Trees / Bushes	on / on	Off = no instances of that kind are processed or rendered (no cost).
Rendering › LOD Bias	1	Multiplier of all LOD distances (higher = detailed LODs further away; 0.25–4).
Distance Scale	1	Multiplier of the profiles' maximum render distances.
Shadows	LOD0 And LOD1	Which mesh LODs cast shadows: Off, LOD0, LOD0 And LOD1, All Mesh LODs (billboards never cast shadows).
Billboards (Impostors)	on	Far billboard impostors for profiles that enable them.
Shared GrassCore Systems	all on	Wind, Seasons, Snow, Fire, Color Zones for trees and bushes.
Features › Interaction / Max Reacting Plants	on / 512	Reaction to interactors and impulses; maximum plants reacting at once (chapter 31).
Flowers / Fruits	on / on	Show the flowers and fruits of profiles that have them (off = their draws are skipped).
Colliders / Collider Distance / Collider Layer	off / 80 m / 0	Primitive colliders near the camera and interactors (chapter 31).

Section / Setting	Default	Description
Destroyable Trees	on	Particles, physics, effects, max falling trees and debris (chapter 32).
Placement	–	<i>Add Manual Instances, Add Area (Box), Add Area (Spline).</i>
Status	–	Placement components and instances, rendered instances, render sets, draws per camera, GPU memory, interaction, colliders, destroyable trees.

Example Workflow

A park scene

- 1 *Tree & Bush Instances (Manual)* with *Old Oak*, *Japanese Maple* and *Round Bush*: place a few hero trees by hand.
- 2 A *Vegetation Area (Spline)* along the edge of the park with *Beech* (count 20) and *Loose Shrub* (count 40).
- 3 World › Trees: *Shadows* = LOD0 And LOD1, *Billboards* on.
- 4 Players get a *Grass Interactor* – it bends the grass and pushes bushes aside.

Performance Notes

- **Zero cost when unused:** the vegetation system exists only while instances of an enabled kind exist.
- Instances are uploaded once when something changes. Every frame the GPU culls them per camera (frustum, distance, LOD with dithered cross-fade) and draws one indirect call per render set, LOD and submesh, plus one billboard draw per set.
- *LOD Bias*, *Distance Scale* and *Shadows* are the main quality / cost controls.
- The *Status* section and the Profiler window show instances, draws and GPU memory.

Common Problems

Problem	Solution
Trees do not appear	Check World › Trees › <i>Trees / Bushes</i> , the profile's <i>Max Distance</i> and that the placement component contains instances.
No GrassCore World in the scene: trees and bushes are rendered by the world	Add a GrassCore World.
Trees pop between LODs	Increase the profile's <i>LOD Cross-Fade</i> or <i>LOD Bias</i> .
Trees cost too much	Lower <i>Distance Scale</i> , <i>LOD Bias</i> or <i>Shadows</i> ; enable billboards.

26

TREES & BUSHES

Procedural Trees & Bushes

Build trees and bushes from parameters – broadleaf trees and conifers, natural, dense, wild, mounded and upright shrubs – or start from 36 species presets, from young oaks to sakura and berry bushes.

What it does

A **Vegetation Profile** with *Meshes = Procedural* builds its geometry from parameters: trunk, branches, twigs and leaf cards for trees; stems and leaves for bushes. GrassCore generates all LODs, wind data and – optionally – flowers and fruits. *Variants* and *Seed* create several natural variations of the same species; instances pick one.

The inspector preview shows the plant with all LODs (drag to rotate), and the **Geometry** section lists vertex and triangle counts per LOD.

How to set it up

- 1 Create a profile with **Assets > Create > GrassCore > Tree Profile** or *Bush Profile* (or *Vegetation Profile (Tree or Bush)*).
- 2 Press **Apply SpeciesPreset...** and pick one of the 36 presets – or start from the default parameters.
- 3 Shape the plant in the **Procedural Tree / Procedural Bush** section.
- 4 Adjust **Materials, Wind, LOD & Rendering** and the optional sections.
- 5 Add the profile to a *Tree & Bush Instances* palette or a *Vegetation Area* (chapter 29).

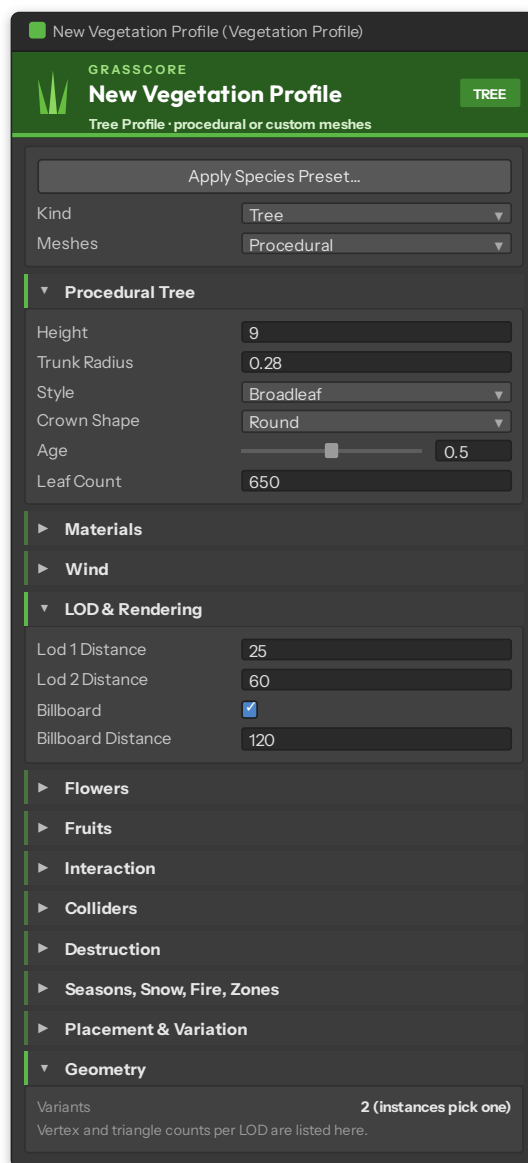


Figure 26.1 – Vegetation Profile inspector.

Species presets

Group	Presets
Broadleaf Trees	Oak, Old Oak, Young Oak, Birch, Beech, Maple, Poplar, Weeping Willow, Acacia
Conifers	Spruce, Fir, Pine, Cypress, Young Spruce
Japanese	Sakura (Cherry Blossom), Japanese Maple, Azalea Mound, Cloud-Pruned Shrub
Fruit Trees	Apple Tree, Cherry Tree, Orange Tree, Lemon Tree
Bushes	Round Bush, Hedge Shrub, Tall Bush, Dense Boxwood, Loose Shrub, Wild Bush
Flowering Bushes	Rhododendron (Pink), Spiraea (White), Forsythia (Yellow), Hydrangea (Blue), Lilac (Purple)
Berry Bushes	Raspberry, Blueberry, Redcurrant

Ready-made profiles for all presets are in [Samples/Vegetation](#), plus *Oak (Destroyable)* and *Spruce (Destroyable)*.

Important Settings

Procedural Tree

Setting	Default	Description
Style	Broadleaf	<i>Broadleaf</i> (rounded crown) or <i>Conifer</i> (straight trunk, whorls of drooping branches, cone-shaped crown).
Height / Trunk Radius	9 m / 0.28 m	Total height and trunk radius at the base.
Crown Start / Crown Radius	0.38 / 3.4 m	Where the crown starts (fraction of the height) and its horizontal radius.
Branch Count / Twigs Per Branch / Branch Angle	9 / 3 / 55°	Main branches, side twigs and the angle from vertical.
Leaf Count / Leaf Size	650 / 0.55 m	Leaf cards on LOD0 (each shows a cluster of leaves) and their size.
Gnarl	0.35	Crookedness of trunk and branches.
Crown Shape	Round	Round, Oval, Columnar, Conical, Spreading, Umbrella, Weeping, Layered, Irregular.
Age	0.5	Young (slim, few branches) to old (thick trunk with root flare, heavy low limbs, wide crown, bare twigs).
Leaf Shape	Auto	Auto, Broad, Small, Maple, Long, Needles (texture of the default leaf material).
Variation	0	Natural differences between variants and seeds.
Variants / Seed	2 / 1	Different random trees from the same parameters (1-4); instances pick one.

Procedural Bush

Setting	Default	Description
Height / Radius	13 m / 11 m	Size of the bush.
Stem Count	6	Woody stems from the base.
Leaf Count / Leaf Size	380 / 0.28 m	Leaf cards on LOD0 and their size.
Flatness / Fill	0.25 / 0.35	0 = round dome, 1 = flat; how much leaves fill the inside.
Style	Natural	Natural, Dense (clipped, hedge, topiary), Wild (long shoots), Mounded (Japanese cloud pruning), Upright.
Openness	0	0 = evenly leafy, 1 = loose clumps with gaps.
Leaf Shape / Variation / Variants / Seed	Auto / 0 / 2 / 1	As for trees.

Materials and general settings

Setting	Default	Description
Trunk (Stem) / Leaves / Flowers / Fruits Material	empty	Empty = GrassCore defaults. Materials with other shaders are converted to GrassCore/Vegetation automatically.
Default Bark / Leaf Colour	brown / green	Colours of the default materials.
Color Variation	0.18	Random brightness / hue variation of the leaves per instance.
Scale Variation	0.9–11	Random scale multiplier per instance.
Spacing Radius	1m	Personal space used by Vegetation Areas.
Align To Ground	0	Tilt instances towards the ground normal (0 = upright).
Season Response / Deciduous / Snow Response / Color Zone Response / Burnable	1/ off / 1/ 1/ on	Shared systems (only when enabled in World › Trees). <i>Deciduous</i> : bare in winter.

Example Workflow

A Japanese garden

- 1 Profiles from presets: *Japanese Maple* (Layered crown), *Sakura*, *Cloud-Pruned Shrub*, *Azalea Mound*.
- 2 *Sakura*: Flowers on with *Seasonal* – it blossoms in spring only ([chapter 30](#)).
- 3 Raise *Variants* to 3 and *Variation* to 0.4 for natural differences between the maples.
- 4 Place them manually around a pond ([chapter 29](#)); *Deciduous* on for the maple so it is bare in winter.

Performance Notes

- *Leaf Count* is the main geometry control – check the **Geometry** section after changes.
- Each variant is a separate render set; 2–3 variants are usually enough.
- LOD1, LOD2 and billboards are generated automatically ([chapter 28](#)).
- Procedural meshes are generated in memory – no mesh assets are written.

Common Problems

Problem	Solution
All trees look identical	Raise <i>Variants</i> and <i>Variation</i> , enable <i>Random Rotation</i> and scale ranges in the placement.
Crown too dense / too expensive	Lower <i>Leaf Count</i> and raise <i>Leaf Size</i> .
Leaves do not change in autumn	Enable <i>Seasons</i> in World › Trees and keep <i>Season Response</i> > 0.
Bush looks like a ball	Try <i>Style Wild</i> or <i>Mounded</i> , raise <i>Openness</i> .

Custom Tree & Bush Meshes

Bring your own tree and bush models – with LOD Groups, multiple materials and optional hand-authored wind data – and use them with GrassCore's culling, LODs, billboards, wind, seasons and interaction.

What it does

With *Meshes = Custom Meshes* a profile renders your own meshes for LOD0, LOD1 and LOD2 (missing LODs reuse the previous one), plus an optional billboard. Each submesh gets a **role** that decides how it is rendered:

Submesh role	Rendering
Auto	Detected from the material name (bark / trunk / wood vs leaf / foliage / needle) and the submesh order.
Trunk	Opaque, bends with trunk and branches.
Leaves	Alpha clipped, two-sided, flutters.
Flowers	Alpha-clipped cards tinted with the Flowers colours; switchable with the world's <i>Flowers</i> feature.
Fruits	Opaque, tinted with the Fruits colours; switchable with the world's <i>Fruits</i> feature.
Ignore	Not rendered.

Any material works: materials with other shaders (URP Lit, imported materials) are converted to **GrassCore/Vegetation** automatically (base map, colour, normal map, cutoff).

Wind data

Mode	How it works
Simple (<i>Wind Data = Automatic</i>)	Works immediately: wind weights are derived from the vertex height and the distance to the trunk axis – the base barely moves, the crown more, leaf submeshes flutter.
Advanced (<i>Vertex Colors</i>)	Vertex colours hold the wind data: R = wind influence, G = branch flexibility, B = leaf flutter, A = phase. Generate Tree Wind Data / Generate Bush Wind Data writes them into mesh copies that you can refine in a DCC tool.

How to set it up

- 1 Create a profile (**Assets > Create > GrassCore > Tree Profile**) and set **Meshes = Custom Meshes**.
- 2 Fastest: drag your model or prefab into **Import Model / Prefab** and press **Import**. GrassCore reads LOD Groups, submeshes and materials.
- 3 Or assign **LOD0 Mesh** (required), **LOD1 Mesh** and **LOD2 Mesh** manually.
- 4 Fix the orientation with **Mesh Rotation** (eg. -90° X for Z-up exports) and the size with **Mesh Scale**.
- 5 Check the **Submesh** roles and the materials.
- 6 Optional: switch *Wind Data* to Advanced and press *Generate ... Wind Data*.
- 7 Optional: **Bake Billboard Textures** stores the billboard views as PNGs next to the profile ([chapter 28](#)).

Important Settings

Setting	Default	Description
LOD0 / LOD1 / LOD2 Mesh	–	LOD0 = full detail; missing LODs reuse the previous mesh.
Mesh Rotation / Mesh Scale	0,0,0 / 1	Applied to the custom meshes.
Submesh roles	Auto	One role per submesh of LOD0; other LODs use the same order.
Wind Data (Mode)	Automatic	Simple (automatic) or Advanced (vertex colours).

Setting	Default	Description
Wind Strength / Trunk Sway / Branch Bend / Leaf Flutter / Motion Speed	1/1/1/1/1	Wind response (chapter 30).
Colliders › Generate Tree / Bush Collider	–	Fits editable primitive collider shapes to the mesh (chapter 31).

Recommended geometry budgets

Trees are instanced many times, but far fewer than grass blades. These are guidelines for real-time games; use the profile's **Geometry** section to compare with the procedural trees.

Plant	LOD0	LOD1	LOD2	Notes
Large tree	4 000–12 000 tris	30–50 % of LOD0	10–20 % of LOD0	Billboard beyond ~120 m.
Small tree	2 000–6 000 tris	30–50 %	10–20 %	
Bush	500–3 000 tris	30–50 %	10–20 %	Bushes usually need no billboard.
Hero tree (few instances)	up to 25 000 tris	30–40 %	10–15 %	Keep the count low.

Prefer alpha-clipped leaf cards with cluster textures over modelled leaves, keep the number of submeshes / materials small (each submesh is one draw per LOD), and author clean LODs with similar silhouettes for smooth cross-fades.

Example Workflow

Importing a tree from an asset pack

- 1 Drag the tree prefab (with its LOD Group) into *Import Model / Prefab*, press *Import*.
- 2 The roles are detected from the material names; set any unclear submesh to *Trunk* or *Leaves* manually.
- 3 The pack's model is Z-up: *Mesh Rotation X* = –90.
- 4 Keep *Wind Data* = *Automatic* first. If the trunk bends too much, switch to *Advanced*, press *Generate Tree Wind Data* and paint the vertex colours in your DCC tool.
- 5 *Bake Billboard Textures* for a stable far LOD.

Performance Notes

- Every submesh is drawn per LOD in use – combine materials where possible.
- Large LOD0 meshes multiplied by many close instances dominate the cost; use *LOD1 Distance* and *LOD Bias* to switch early.
- Converted materials are created at runtime; no extra assets are written.

Common Problems

Problem	Solution
LOD0 is empty: the first assigned mesh is used for it	Assign the LOD0 mesh.
Tree lies on its side	Set <i>Mesh Rotation</i> (eg. –90° X).
Leaves render opaque	Set the submesh role to <i>Leaves</i> (alpha clipped, two-sided).
Trunk sways like rubber	Lower <i>Trunk Sway</i> , or use <i>Advanced</i> wind data with low R values near the base.
Flowers / fruits of the model cannot be switched off	Set their submeshes to the <i>Flowers / Fruits</i> role.

28

TREES & BUSHES

Tree & Bush LODs and Billboards

Three mesh LODs with dithered cross-fades and an optional billboard impostor for the far distance keep forests detailed up close and cheap in the distance.

What it does

Every tree and bush instance is assigned a level of detail on the GPU, per camera:

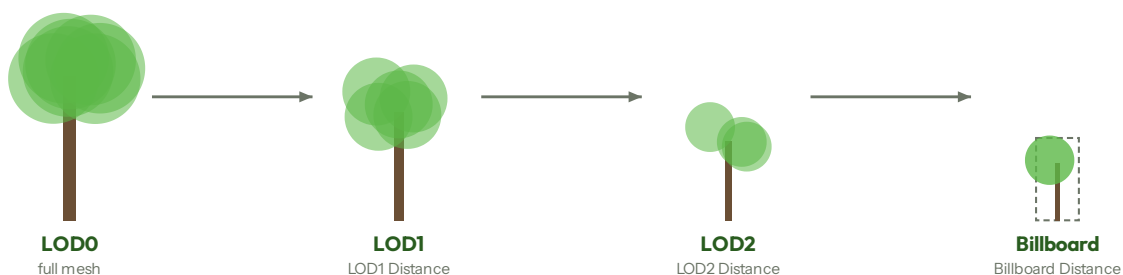


Figure 28.1 – Mesh LODs and the billboard impostor of a tree.

Level	Distance	Content
LOD0	up to <i>LOD1Distance</i>	Full mesh.
LOD1	up to <i>LOD2Distance</i>	Reduced mesh (generated for procedural profiles).
LOD2	up to <i>Billboard Distance</i>	Lowest mesh.
Billboard (LOD3)	up to <i>Max Distance</i>	Impostor quad: 8 baked views (albedo + normals), dithered between the two nearest views.

Distances are for instance scale 1 – larger instances switch later. Between two LODs both are rendered over a short **cross-fade** window with complementary dithering, so switches are soft. The world's *LOD Bias* multiplies all LOD distances, *Distance Scale* the maximum distances.

Procedural billboards are baked automatically on the GPU when the trees are first rendered. For custom meshes you can bake them once into textures.

How to set it up

- 1 Open **LOD & Rendering** in the profile.
- 2 Set **LOD1Distance** and **LOD2Distance**.
- 3 Enable **Billboard (LOD3)** for trees and set **Billboard Distance** and **Billboard View Size (px)**.
- 4 Set **Max Distance** and **LOD Cross-Fade**.
- 5 Custom meshes: press **Bake Billboard Textures** to bake the 8 views into PNG textures next to the profile; otherwise they are baked automatically at runtime. Baked atlases can also be assigned to *Baked Albedo* / *Baked Normals*.
- 6 Use the preview in the inspector to inspect each LOD.

Important Settings

Setting	Default	Description
LOD1Distance (m)	25	Where LOD1 starts.
LOD2Distance (m)	60	Where LOD2 starts.
Billboard (LOD3)	on	Far billboard impostor (trees).
Billboard Distance (m)	120	Where the billboard starts.

Setting	Default	Description
Billboard View Size (px)	256	Pixel size of one of the 8 baked views.
Baked Albedo / Normals	–	Optional baked atlases (4 × 2 views); baked automatically when empty.
Max Distance (m)	600	Maximum render distance (× the world's <i>Distance Scale</i>).
LOD Cross-Fade	0.12	Width of the dithered cross-fade (fraction of the switch distance).
Shadows	On	<i>On</i> (world setting decides how many LODs), <i>LOD0 Only</i> or <i>Off</i> . Billboards never cast shadows.
World › LOD Bias / Distance Scale	1 / 1	Global multipliers.

Example Workflow

Tuning a forest for a console target

- 1 World › Trees: *LOD Bias* 0.8, *Shadows* LOD0 And LOD1.
- 2 Large trees: *Billboard Distance* 90 m, *Max Distance* 450 m.
- 3 Bushes: *Billboard* off, *Max Distance* 120 m, *Shadows* LOD0 Only.
- 4 Compare *draws per camera* and *GPU memory* in World › Trees › Status before and after.

Performance Notes

- LOD selection and culling run in a compute shader – no CPU cost per instance.
- Billboards are the cheapest representation: a single quad per tree.
- Shadows are often the biggest cost of a forest: limit them to LOD0 / LOD1.
- A wider *LOD Cross-Fade* renders two LODs for longer; keep it small.

Common Problems

Problem	Solution
Visible pop between LODs	Raise <i>LOD Cross-Fade</i> slightly or make LOD silhouettes more similar.
Billboard looks wrong (dark / wrong colours)	For custom meshes bake the billboard again after changing materials; check the submesh roles.
Trees disappear too early	Raise <i>Max Distance</i> or the world's <i>Distance Scale</i> ; check the camera's far clip plane.
Billboards cast no shadows	By design – billboards never cast shadows.

Manual Placement & Vegetation Areas

Place single trees exactly where you want them, or fill an area with an exact number of plants – deterministic, with spacing rules, slope and height limits.

What it does

Two components store tree and bush instances in the scene:

Component	Use
Tree & Bush Instances (Manual) (<code>GrassCoreVegetationLayer</code>)	Place, edit and erase single plants with the mouse. Instances stay exactly where they are put.
Vegetation Area (<code>GrassCoreVegetationArea</code>)	Generates plants inside a Box, Ellipse or Spline – with an exact <i>Count</i> per entry or a <i>Total Count</i> split by weight. Same seed and settings = same result.

Instances are stored relative to the GameObject, so moving, rotating or scaling the object moves the plants with it. Undo works for everything.

How to set it up

Manual placement

- GameObject > GrassCore > Vegetation > Tree & Bush Instances (Manual).**
The palette starts with the default tree and bush; drag more profiles onto *Add Profile*.
- Choose a tool: **Place** (click the ground), **Edit** (click an instance, then move / rotate / scale it with W / E / R; Delete removes it) or **Erase** (click to delete). *Off* returns to normal selection.
- Random Rotation*, *Random Scale* and *Keep on Ground* apply while placing and moving.
- Snap All to Ground** after terrain edits; **Clear All** removes every instance (with confirmation).

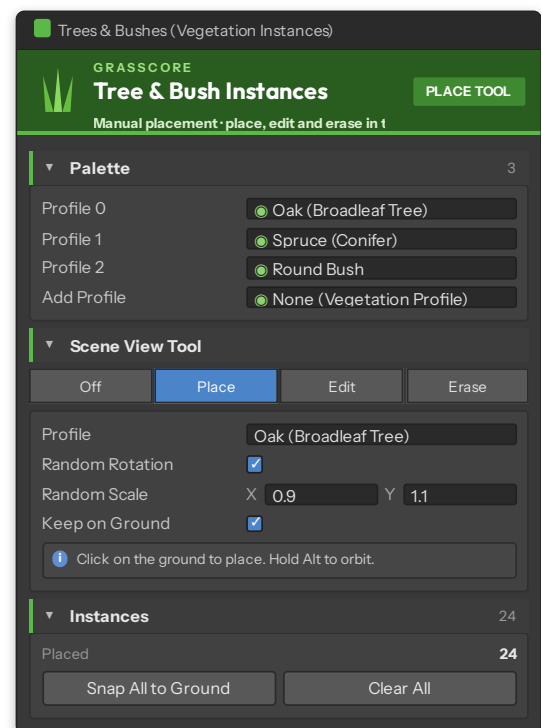


Figure 291 – Tree & Bush Instances inspector.

Vegetation Areas

- 1 **GameObject** › **GrassCore** › **Vegetation** › **Vegetation Area (Box)** or (*Spline*) – draw the outline in the Scene view for splines.
- 2 Add entries with **Add Tree**, **Add Bush**, **Add Empty** (then assign any profile) and set each **Count** – or switch *Distribution* to **Weighted** and set a *Total Count* and weights.
- 3 Set the **Placement Rules** and the **Seed**.
- 4 **Generate** places the instances, **Regenerate** rolls a new seed, **Clear** removes them. With **Auto Regenerate** the area updates shortly after you stop editing.
- 5 Each entry shows *placed / requested* – if fewer fit, enlarge the area or lower the count or distance.

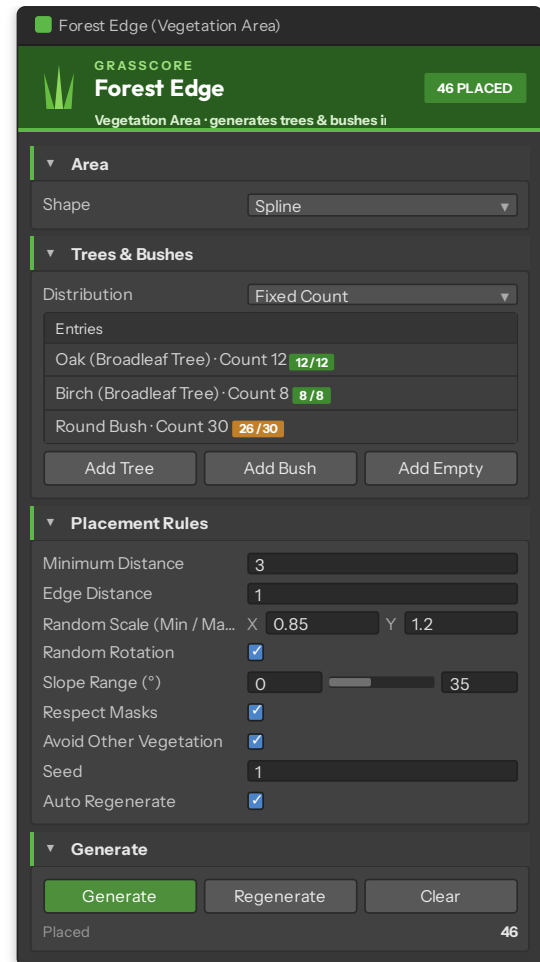


Figure 29.2 – Vegetation Area inspector.

Important Settings

Vegetation Area

Setting	Default	Description
Shape / Size	Box / 40 × 40 m	Box, Ellipse (size in the transform's orientation) or Spline.
Distribution	Fixed Count	<i>Fixed Count</i> per entry or <i>Weighted</i> (Total Count split by weight).
Total Count	40	Weighted distribution: number of instances.
Minimum Distance	3 m	Minimum spacing between instances (profiles' <i>Spacing Radius</i> also applies).
Edge Distance	1m	Distance kept from the outline.
Random Scale (Min / Max)	0.85–12	Random scale per instance.
Random Rotation / Random Tilt	on / 2°	Rotation around the up axis and additional tilt.
Height Range / Slope Range	±10 000 m / 0–35°	Allowed ground height and slope.
Respect Masks	on	Keep out of <i>Exclude</i> mask volumes, paths and water.
Avoid Other Vegetation	on	Keep the spacing to trees / bushes of other areas and manual instances.
Seed	1	Same seed + same settings = same positions.
Auto Regenerate	on	Regenerate automatically in the editor after changes.

Tree & Bush Instances

Setting	Description
Palette	Profiles that can be placed (instances refer to them by index; removing a profile removes its instances after a confirmation).

Setting	Description
Scene View Tool	Off, Place, Edit, Erase.
Random Rotation / Random Scale / Keep on Ground	Applied while placing; <i>Keep on Ground</i> keeps moved instances on the ground.

From code, `GrassCoreVegetationLayer` offers `Add`, `Set`, `SetProfile`, `RemoveAt`, `Clear` and `SnapToGround`; areas offer `Generate()`, `Regenerate()` and `Clear()`.

Example Workflow

A mixed forest with a clearing and a path

- 1 Draw a *Vegetation Area (Spline)* around the forest.
- 2 *Weighted* distribution, *Total Count* 400: *Spruce* weight 3, *Beech* 2, *Round Bush* 4.
- 3 Put an *Exclude* mask volume over the clearing and a path through the forest – with *Respect Masks* on, no tree grows there.
- 4 Place a few hero trees manually in the clearing with *Tree & Bush Instances*; *Avoid Other Vegetation* keeps the area's trees away from them after the next *Generate*.

Performance Notes

- Generation runs in the editor (or when you call `Generate()`), never per frame.
- Rendering cost depends mainly on the visible instances and their LODs; instances outside the view are culled on the GPU.
- *Auto Regenerate* can be switched off for very large areas to avoid waiting while editing.

Common Problems

Problem	Solution
Nothing could be placed	Check the ground under the area (terrain or colliders), <i>Height / Slope Range</i> , <i>Edge Distance</i> and masks.
Fewer instances than requested	Enlarge the area, lower <i>Minimum Distance</i> or the count. Profiles with a large <i>Spacing Radius</i> need more room.
Settings changed since the last generation – press Generate	Press <i>Generate</i> or enable <i>Auto Regenerate</i> .
Trees float after terrain sculpting	<i>Snap All to Ground</i> (manual instances) or <i>Generate</i> (areas).

30

TREES & BUSHES

Tree & Bush Wind, Flowers & Fruits

Hierarchical wind – trunk sway, bending branches, fluttering leaves – driven by the same GrassCore wind as the grass, plus flowers and fruits that sit naturally on the foliage and follow the seasons.

What it does

Wind

Trees and bushes move in three layers, all from the GrassCore wind ([chapter 13](#)):

Layer	Effect	Profile setting
Trunk sway	Slow sway of the whole plant (bushes: <i>Whole Bush Sway</i>).	Trunk Sway
Branch bend	Branches and thin twigs bend with phase variation.	Branch Bend
Leaf flutter	Fast, fine fluttering of the leaves.	Leaf Flutter

Procedural profiles write their wind data into the mesh automatically. Custom meshes use *Simple* mode (derived from height and distance to the trunk axis) or *Advanced* mode with vertex colours ([chapter 27](#)).

Flowers and fruits

Procedural profiles can grow **flowers** (alpha-clipped blossom cards) and **fruits** (low-poly round, oval, cherry or berry shapes on stalks). They sit on the foliage – outside, throughout, at twig tips or in clusters – and fruits hang from twigs inside the crown, never floating outside the foliage. They follow the wind and thin out on LOD1 / LOD2. With **Seasonal**, flowers bloom only in spring and fruits appear only from late summer to autumn. Custom meshes mark submeshes as *Flowers* or *Fruits* ([chapter 27](#)).

The world's **Flowers** and **Fruits** switches (World › Trees › Features) hide them for all plants – their draws are skipped, so there is no cost.

How to set it up

- 1 Open the profile's **Wind** section. Adjust *Wind Strength*, *Trunk Sway*, *Branch Bend*, *Leaf Flutter* and *Motion Speed*.
- 2 Make sure *Wind* is enabled in World › Trees › *Shared GrassCore Systems*.
- 3 Open **Flowers** and/or **Fruits**, enable them, set *Amount*, *Size*, *Colour / Colour 2*, *Randomness*, *Distribution* (fruits also *Shape*) and *Seasonal*.
- 4 The flowering and fruit presets (*Sakura*, *Rhododendron*, *Apple Tree*, *Raspberry*, ...) already use these settings.

Important Settings

Wind (profile)

Setting	Default	Description
Wind Data (Mode)	Automatic	Custom meshes: Simple (automatic) or Advanced (vertex colours).
Wind Strength	1	Overall response to the GrassCore wind (0–3).
Trunk Sway / Whole Bush Sway	1	Low-frequency sway of the whole plant.
Branch Bend	1	Bending of branches and thin twigs.
Leaf Flutter	1	Fast, fine fluttering of the leaves.
Motion Speed	1	Speed of branch and leaf motion (0.2–3).

Flowers / Fruits (profile)

Setting	Flowers default	Fruits default	Description
Enabled	off	off	Generate flowers / fruits (procedural profiles).
Amount	0.5	0.5	Relative to the foliage (flowers: 1= one card per two leaf cards; fruits: 1= plenty).
Size	0.18 m	0.08 m	Flower card size / fruit diameter.
Colour / Colour 2	pink / white	red / green	Two colours; each flower or fruit varies between both.
Randomness	0.5	0.5	Variation of size, colour and orientation / position.
Distribution	Outer	Hanging	Outer, Throughout, Branch Tips, Clustered, Hanging.
Shape	–	Round	Fruits: Round, Oval, Cherry, Berry.
Seasonal	off	off	Flowers: spring only. Fruits: late summer to autumn.
Flowers / Fruits Material	–	–	Empty = default blossom / fruit material tinted with the colours.

Example Workflow

An orchard through the year

- 1 Profile from the *Apple Tree* preset; *Flowers* on and *Seasonal*; *Fruits* on and *Seasonal*.
- 2 Place the trees in rows with a *Vegetation Area (Box)*, *Minimum Distance* 6 m.
- 3 Drive the season ([chapter 19](#)): blossoms in spring, green leaves in summer, red apples in late summer, coloured leaves in autumn, bare branches in winter if *Deciduous* is on.

Performance Notes

- Wind is evaluated in the vertex shader – no CPU cost per plant.
- Flowers and fruits are extra submeshes: one draw per render set and LOD when present. Switching them off in the world skips these draws completely.
- Small fruits thin out faster on LOD1 / LOD2.

Common Problems

Problem	Solution
Trees do not move	Enable <i>Wind</i> in World › Trees › <i>Shared GrassCore Systems</i> ; check the profile's <i>Wind Strength</i> .
Leaves move but the trunk bends unnaturally (custom mesh)	Lower <i>Trunk Sway</i> or use Advanced wind data.
No flowers / fruits	Enabled in the profile and in World › Trees › <i>Flowers / Fruits</i> ? With <i>Seasonal</i> they only appear in their season.
Fruits look too large	Lower <i>Size</i> (fruit diameter in metres).

31

TREES & BUSHES

Tree & Bush Interaction and Optional Colliders

Players walking through bushes push the branches aside, vehicles make trees sway, explosions shake whole forests – simulated on the GPU. Primitive colliders are available when your gameplay needs them.

What it does

Interaction

The same **GrassInteractor** that bends the grass also affects trees and bushes (with *Affect Vegetation* on). Bushes are pushed aside where something moves through the foliage and spring back; trees sway from strong contacts and impacts. Shockwaves and `AddVegetationImpulse` shake the plants around a point.

Each plant that is touched gets a GPU **spring slot** until it has settled. No interactor near any plant means no upload, no dispatch and no shader cost.

Colliders (optional, off by default)

With **Colliders** enabled on the world, plants close to the camera and to interactors get primitive colliders (capsules, boxes, spheres) in Play mode. They are pooled GameObjects – never mesh colliders – created only within the *Collider Distance*.

Collider Mode	Shapes
Auto	Trees: a capsule along the trunk. Bushes: none.
None	No colliders.
Trunk	Capsule along the trunk (bushes: around the stems).
Trunk And Branches	Trunk capsule plus capsules along the largest branches.
Body	One capsule around the whole plant (solid bushes, small trees).
Custom	The shapes listed in the profile; Generate Tree / Bush Collider fits them to the plant.

How to set it up

- 1 World › Trees › Features: keep **Interaction** on.
- 2 Add a *Grass Interactor* to players, NPCs, animals and vehicles; keep **Affect Vegetation** on and pick a preset (or set *Vegetation Strength* and *Vegetation Height*).
- 3 Tune each profile's **Interaction** section.
- 4 For colliders: World › Trees › Features › **Colliders** on; choose a *Collider Distance* and *Collider Layer*; set the profile's **Colliders** mode (or *Custom* + *Generate ... Collider*). *Is Trigger* makes them detect without blocking (eg. bushes that slow the player down).

```
GrassCoreRuntimeAPI.AddVegetationImpulse(explosionCenter, radius: 15f, strength: 2f); // shake trees and bushes
GrassCoreRuntimeAPI.SetVegetationCollidersEnabled(true);
```

Important Settings

Profile › Interaction

Setting	Default	Description
React to Interactors	on	React to GrassCore interactors and impulses (needs Interaction on the world).
Push Strength	1	How far branches and leaves are pushed aside where something moves through the foliage.
Impact Response	1	How much the whole plant sways on contacts and impacts.
Spring Stiffness (Hz)	2	How fast it swings back (0.3–5).
Damping	0.3	0 = long springy swinging, 1 = settles without overshoot.

Setting	Default	Description
Max Lean (°)	18	Maximum lean of the whole plant.

Profile › Colliders and World › Trees › Features

Setting	Default	Description
Mode	Auto	See the table above.
Is Trigger	off	Trigger colliders (detect, do not block).
Shapes	–	Custom shapes: Capsule (x = radius, y = height), Box (size), Sphere (x = radius), with centre and rotation.
World › Max Reacting Plants	512	Plants reacting at the same time (GPU spring slots, 64 B each).
World › Colliders / Collider Distance / Collider Layer	off / 80 m / 0	Colliders within this distance of the camera and the vegetation interactors.

`GrassCoreRuntimeAPI.TryGetTree(collider, out tree)` resolves a physics hit on a vegetation collider to the tree it belongs to (chapter 32).

Example Workflow

Bushes that react and slow the player down

- 1 Bush profiles: *Push Strength* 15, *Damping* 0.4.
- 2 World › Trees: *Colliders* on, *Collider Distance* 40 m.
- 3 Bush profiles: *Colliders = Body*, *Is Trigger* on. Your character controller reduces its speed while inside a trigger.
- 4 Tree profiles: *Colliders = Trunk* – players cannot walk through trunks.

Performance Notes

- Interaction cost scales with the number of reacting plants (capped by *Max Reacting Plants*), not with the number of trees.
- Colliders exist only in Play mode, only near the camera and interactors, and are pooled. Keep *Collider Distance* as small as your gameplay allows.
- Turning *Interaction* or *Colliders* off in the world removes their cost entirely.

Common Problems

Problem	Solution
Bushes do not react	World › Trees › <i>Interaction</i> on? Interactor with <i>Affect Vegetation</i> on? Profile <i>React to Interactors</i> on?
Trees react too strongly	Lower <i>Impact Response</i> and <i>Max Lean</i> , or the interactor's <i>Vegetation Strength</i> .
Player walks through trunks	Enable World › Trees › <i>Colliders</i> and choose a collider mode in the profile.
Colliders missing far away	By design – they exist only within <i>Collider Distance</i> of the camera and interactors.

32

TREES & BUSHES

Destroyable Trees, Tree Damage & Falling Trees

Chop, blast and fell trees: every hit shakes the crown and drops leaves, at zero health the tree breaks at its break height – the stump stays with a real cut surface, the top falls with physics and fades out.

What it does

Trees whose profile has **Destruction > Destroyable** enabled can be damaged from any source – an axe, a vehicle, an explosion or your own weapon code:

- 1 **Hit** – the crown and upper trunk swing in the hit direction (*Hit Wobble*), leaves fall from the crown and wood chips fly from the impact point.
- 2 **Felled** – at zero health the tree breaks at its *Break Height*. The **stump** stays in place with a cut surface built from the real trunk cross-section.
- 3 **Falling** – the top part falls, preferably in the direction of the last or strongest hits: with **Physics** (Rigidbody and colliders, created only at the moment of felling) or as a cheap scripted fall.
- 4 **Landing** – a burst of leaves and twigs; after *Fallen Tree Lifetime* the top part fades out. The stump stays.

There are no per-tree scripts, updates or damage checks: damage arrives through the API, and the destruction system is created on the first hit. Bushes are never destroyable. Destruction works in Play mode.

Samples: **Oak (Destroyable)** and **Spruce (Destroyable)** in `Samples/Vegetation`. The demo player chops destroyable trees with the left mouse button (*GrassCoreDemoAxe*).

How to set it up

- 1 Open a tree profile's **Destruction** section and enable **Destroyable**.
- 2 Set **Health**, **Auto Break Height** (or a manual *Break Height*), the falling behaviour and the particles / effects.
- 3 Make sure World > Trees > **Destroyable Trees** is on.
- 4 Apply damage from your code:

```
// Melee / axe: ray against the trunks - no colliders needed.
if (GrassCoreRuntimeAPI.RaycastTree(new Ray(eye, forward), 3f, out GrassCoreTreeHit hit))
    hit.tree.ApplyDamage(25f, hit.point, forward);

// Nearest tree around a position:
if (GrassCoreRuntimeAPI.TryGetTree(position, 1.5f, out GrassCoreTree tree))
    tree.ApplyDamage(10f, position, direction);

// Explosion with falloff - trees fall away from the centre:
GrassCoreRuntimeAPI.DamageTrees(center, radius: 12f, damage: 150f);

// React to events:
GrassCoreTreeEvents.Felled += info => SpawnLogs(info.position, info.fallDirection);
```

Important Settings

Profile > Destruction

Setting	Default	Description
Destroyable	off	The tree can be damaged and felled. Off = no damage checks, buffers, particles or physics for this profile
Health	100	Hit points, scaled with the instance size (a tree twice as large has twice the health).
Auto Break Height / Break Height (m)	on / 0.7	Break height chosen automatically (above the root flare, below the crown) or set manually.
Hit Wobble	1	How much crown and upper trunk swing on each hit.

Setting	Default	Description
Physics	on	The top falls with physics; off = scripted fall.
Fall Direction Randomness	0.15	0 = exactly in the direction of the strongest / last hits.
Fallen Tree Lifetime (s)	30	Time the fallen top lies on the ground before it fades.
Fade Out (s)	2.5	Duration of the fade-out.
Falling Leaves / Leaves per Hit	on / 14	Leaves falling from the crown on each hit.
Wood Chips	on	Chips fly from the impact point.
Landing Burst	on	Leaves and twigs burst out when the tree lands.

World › Trees › Destroyable Trees

Setting	Default	Description
Destroyable Trees	on	Master switch for all trees.
Particles (Falling Leaves)	on	GPU leaf particles on hits (no GameObjects).
Physics (Falling Trees)	on	Off = scripted falls for all trees.
Effects (Chips, Landing Burst)	on	Extra effects.
Max Falling Trees	8	Fallen top parts existing at once (pooled; the oldest is removed).
Max Debris Particles	2048	Debris particles alive at once (ring buffer, 48 Beach).

API overview

Member	Purpose
<code>GrassCoreTree.ApplyDamage(damage, hitPosition, hitDirection)</code>	Damage from any source; returns false when the tree is not destroyable, already felled, the feature is off or in edit mode.
<code>Fell(direction, animate)</code> / <code>Restore()</code>	Fell at once (<code>animate = false</code> : stump only, no events) / full health, no stump.
<code>IsValid</code> , <code>IsDestroyable</code> , <code>IsFelled</code> , <code>Health</code> , <code>MaxHealth</code> , <code>Position</code> , <code>BreakPoint</code> , <code>TrunkRadius</code> , <code>Height</code> , <code>Profile</code>	State of the tree handle.
<code>GrassCoreRuntimeAPI.RaycastTree</code> , <code>TryGetTree(position, radius, ...)</code> , <code>TryGetTree(collider, ...)</code>	Find trees by ray, position or physics hit.
<code>DamageTrees(center, radius, damage)</code>	Explosion damage with falloff.
<code>RestoreAllTrees()</code> , <code>GetFelledTrees(list)</code> , <code>SetTreeFelled(position)</code>	Restore all trees; save / load felled trees.
<code>GrassCoreTreeEvents</code>	<code>Damaged</code> , <code>Felled</code> , <code>Landed</code> , <code>FallenTreeRemoved</code> , <code>Restored</code> .

Example Workflow

Lumberjack gameplay with save games

- 1 Profiles *Oak (Destroyable)* and *Spruce (Destroyable)*; *Health* 120.
- 2 The axe script uses `RaycastTree` and `ApplyDamage(30, hit.point, forward)`; a hit sound plays on `GrassCoreTreeEvents.Damaged`.
- 3 On `Felled`: spawn log items at `info.position` along `info.fallDirection`.
- 4 Saving: `GetFelledTrees(list)` stores the stump positions; loading calls `SetTreeFelled(position)` for each (no animation or events).
- 5 Regrowth after some in-game days: `tree.Restore()` or `RestoreAllTrees()`.

Performance Notes

- Nothing runs until the first hit; profiles without *Destroyable* have no destruction cost at all.
- Physics bodies exist only for falling top parts and are pooled (*Max Falling Trees*).
- Leaves, chips and bursts are GPU particles in one ring buffer – one draw call while any is alive.
- Switching off *Particles*, *Physics* or *Effects* in the world reduces the cost further.

Common Problems

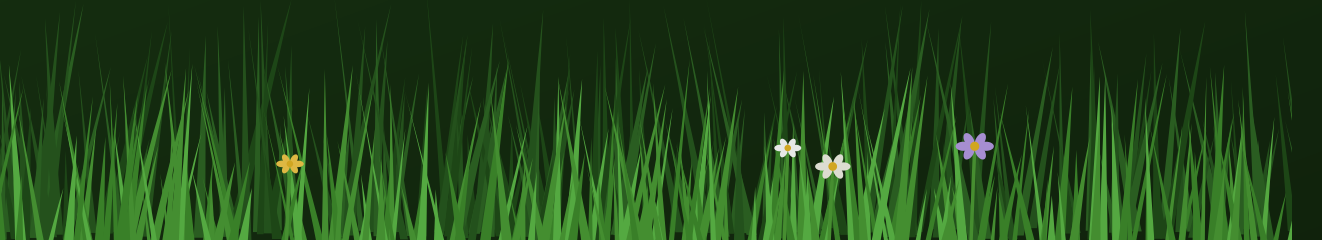
Problem	Solution
<code>ApplyDamage</code> returns false	The profile is not <i>Destroyable</i> , the tree is already felled, World › Trees › <i>Destroyable Trees</i> is off, or you are in edit mode.
<code>RaycastTree</code> finds nothing	Increase the max distance; the ray is tested against the trunks of destroyable trees.
Trees fall into walls	Enable <i>Physics</i> so the top collides; lower <i>Fall Direction Randomness</i> for predictable falls.
Fallen trees vanish too early	Raise <i>Fallen Tree Lifetime</i> and <i>Max Falling Trees</i> .
Tree handles invalid after a floating-origin shift	Fetch the handles again after <code>OnOriginShifted</code> .

PART 6

Performance & Reference

Optimise for your target hardware, debug problems quickly and look up answers, API examples, best practices and limits.

33	Performance & Optimization	88
34	Debugging & Common Problems	91
35	Frequently Asked Questions	93
36	Runtime API – Short Examples	95
37	Best Practices	99
38	Reference & Support	101



Performance & Optimization

How GrassCore spends CPU and GPU time, which settings matter most, how GPU instancing and culling work, and how to keep very large scenes fast.

What it does

GrassCore is GPU-driven. Understanding where the time goes helps you choose the right settings:

Cost	Scales with	Main controls
CPU per frame	Chunk bookkeeping and tile builds of newly visible chunks (budgeted). Not with the number of blades.	<i>Chunk Size, Tile Builds Per Frame, Render Distance</i>
GPU generation	Candidate positions in visible chunks – thanks to the density lattice this follows what you see, not the view distance.	<i>Blades / m², Density Scale, Halving Distance, Full Density Distance</i>
GPU rendering	Rendered blades × triangles per blade, plus shadow passes.	<i>Max Instances, Blade Segments, species Max Distance, custom mesh budgets, Shadows</i>
GPU memory	Tile cache (chunks × <i>Tile Resolution</i>), instance budget, optional systems in use.	<i>Tile Resolution, Max Instances, Render Distance</i>

GPU instancing and culling

GrassCore never creates GameObjects or renderers per blade or per tree:

- 1 Chunk culling** (per camera): only chunks inside the camera frustum and render distance are processed.
- 2 Generation** (compute shader): each visible chunk enumerates only the density levels that can survive at its distance; every candidate is tested against the surface tile, filters, masks and a fine frustum test (with a margin close to the camera for shadows).
- 3 LOD bucketing**: surviving blades are sorted into buckets per species and LOD.
- 4 Indirect rendering**: one `RenderMeshIndirect` per bucket in use draws all instances; shadows, depth and depth-normals use the same draws. The draw arguments are written on the GPU – there are **no CPU readbacks** and **no per-frame allocations**.

Trees and bushes follow the same principle with their own cull kernel: per-instance frustum, distance and LOD selection with dithered cross-fade, then one indirect draw per render set, LOD and submesh.

How to set it up

- 1** Choose a quality preset per platform ([chapter 4](#)). Start with **High** on desktop GPUs, **Medium** on laptops and consoles, **Low** on low-end hardware.
- 2** Open **Tools > GrassCore > Profiler** and play your scene. Watch *Blades, Dropped, Candidates per blade, Tile slots used* and *GPU memory*.
- 3** Adjust in this order:
 - 1 Halving Distance** – the most effective far-field control.
 - 2 Shadows / Shadow Distance** – LOD0 only, or off.
 - 3 Render Distance** – and the species *Max Distance* for small details such as flowers.
 - 4 Density** – *Blades / m²* or *Density Scale*.
 - 5 Blade Segments** – fewer segments per LOD.
- 4** If *Dropped* is non-zero, raise *Max Instances* or lower the density.

Important Settings

Setting	Impact	Recommendation
Halving Distance	Very high	Lower values thin the far field quickly. Keep <i>Fade Width</i> large enough to avoid visible rings.
Shadows	High	<i>LOD0</i> only; <i>Off</i> for low-end and for sparse species (flowers).
Render Distance	High	Match it to your camera and fog. Use fog to hide the end of the grass.
Blades / m ² , Density Scale	High	Fewer, wider blades (<i>Far Width Expansion</i>) often look as dense.
Blade Segments	Medium	3–5 segments for <i>LOD0</i> are enough for most grass.
Max Instances	Memory	Budget of rendered blades per camera; too low drops blades.
Tile Resolution	Memory / detail	64 is a good default; 32 for low-end.
Tile Builds Per Frame	Spikes	Lower it if streaming causes hitches, raise it if chunks appear late.
Chunk Size	CPU	32 m default; larger chunks for very large view distances.
Render In Scene View / Reflections	Editor / extra cameras	Reflection cameras are off by default – keep them off unless needed.

Recommended geometry budgets

Content	LOD0	LOD1	LOD2
Procedural grass blade (High)	9 tris	5 tris	1 tri
Custom grass clump	12–40 tris	6–16 tris	2–6 tris
Procedural flower (built-in)	28–44 tris	18–28 tris	6–7 tris
Custom flower / small plant	24–60 tris	12–30 tris	4–8 tris
Bush (custom mesh)	500–3 000 tris	30–50 %	10–20 %
Tree (custom mesh)	2 000–12 000 tris	30–50 %	10–20 %, then billboard

Tips for large scenes

- Cost depends on the view distance, not the world size – a 10 km terrain with 250 m grass distance renders about as fast as a small one.
- Use **fog** and the quality *Render Distance* together: grass fading into fog hides the far edge naturally.
- Optional systems cost nothing until used (Figure 1.2). Use them only where your gameplay needs them.
- For worlds larger than a few kilometres use a **floating origin**: when your system shifts the world so that position `offset` becomes the origin, call `GrassCoreRuntimeAPI.OnOriginShifted(offset)`. Placement stays identical.
- After scripted terrain or mesh changes, rebuild only the affected region: `GrassCoreRuntimeAPI.RefreshRegion(bounds)`.
- Teleports (anchor jumps larger than the render distance) use a burst budget for tile builds automatically.
- Several cameras each run their own generation and draws. Give the grass its own *Layer* (World › Quality › Rendering) and remove that layer from the culling mask of helper cameras that do not need grass.

Example Workflow

From 9 ms to 3 ms on a laptop GPU

- Starting point: *High*, shadows *LOD0*, render distance 250 m.
- Customise *This Preset*; *Halving Distance* 25 → 16; *Fade Width* 10 → 12.
- Shadow Distance* 40 → 20; flowers *Shadows* = *Off*.
- Species *Tall Grass*: *Max Distance* 120 m; flowers *Max Distance* 60 m.
- Blade Segments* 5 / 3 / 1 → 4 / 2 / 1.
- Check *Candidates per blade* and *Blades* in the Profiler after each step.

Performance Notes

- Measure in a build or in the Game view: the Scene view renders grass as well while it is visible.

- *Candidates per blade* close to 1 means very little wasted work; the Profiler also shows how much work is saved compared to a full-frustum grid.
- GPU memory is shown in the World inspector (*Quality › Resolved*), the Debug tab, the Profiler and the statistics panel.

Common Problems

Problem	Solution
Hitches when moving fast	Lower <i>Tile Builds Per Frame</i> or raise <i>Chunk Size</i> .
Grass appears late after teleports	Raise <i>Tile Builds Per Frame</i> ; teleports already trigger a burst budget.
Dropped blades	Raise <i>Max Instances</i> or reduce density / distance.
High GPU memory	Lower <i>Tile Resolution</i> , <i>Max Instances</i> or <i>Render Distance</i> ; check optional systems (mowing cells, water, trees) in the Profiler.
Visible rings at distance	Raise <i>Fade Width</i> and <i>Halving Distance</i> ; raise the species' <i>Far Width Expansion</i> .

Debugging & Common Problems

Debug views, Scene view gizmos, the GrassCore overlay, the Profiler window and statistics panels – and a checklist for the most common problems.

What it does

GrassCore offers several tools to see what is happening:

Tool	Where	Shows
Debug View	World › Debug, Scene view overlay, <code>SetDebugView</code>	Colours the grass: <i>LOD</i> (red LOD0, green LOD1, blue LOD2), <i>Species</i> (by index), <i>Density Fade</i> (red = fading out), <i>Interaction</i> (bending / flattening), <i>Wind</i> (wind bending).
Scene Gizmos	World › Debug	<i>Draw Chunks</i> (orange = building, yellow = rebuilding, green shades = density level), <i>Only Visible Chunks</i> , <i>Draw LOD Rings</i> (LOD, full-density and render-distance rings), <i>Draw Interaction</i> (interaction area, active shockwaves and fires).
GrassCore Scene overlay	Scene view › Overlays menu › <i>GrassCore</i>	Debug view, chunk and LOD ring toggles, blades, chunks, pending tiles and GPU memory.
Profiler window	Tools › GrassCore › Profiler or Window › Analysis › GrassCore Profiler	Blades per LOD and species with history, culling efficiency, streaming, CPU / GPU timings, memory and every optional system.
World › Debug › Statistics	World inspector	Chunks, tile builds, blades, LOD split, candidates, fires, camera views, GPU memory, CPU time; <i>Rebuild All Tiles</i> , <i>Open Profiler Window</i> .
Statistics Overlay	GameObject › GrassCore › Statistics Overlay	Text overlay in the Game view (toggle with F9, legacy input).
Demo Statistics panel	Tools › GrassCore › Create Demo UI	Canvas panel with tiles and an LOD bar (F9 or close button); replaces the text overlay while active.

How to set it up

- 1 Select the world, open **Debug** and choose a *Debug View*.
- 2 Enable *Draw Chunks* and *Draw LOD Rings* to see the streaming grid and LOD distances in the Scene view.
- 3 Open the **GrassCore** overlay from the Scene view's Overlays menu for quick access.
- 4 Open the **Profiler** window while playing.
- 5 In builds, add the Statistics Overlay or the demo UI's statistics panel.

Important Settings

Profiler window sections

Section	Values
Blades	History graph, rendered (primary camera), dropped (instance budget), LOD split bar, blades per species.
Culling efficiency	Visible / active chunks, candidate threads, candidates per blade, full-frustum grid equivalent, work saved vs. full grid.
Streaming	Tile slots used, pending tile builds, tiles built this frame, chunk size, lattice and density levels.
CPU / GPU	CPU tick and render time, camera views, GPU memory, interactors / shockwaves / downwash, colour zones, season / snow, glow, fires, edited chunks, spline shapes, underwater, water surfaces and waves, mowing and clippings, trees and bushes.

Validation messages

The World inspector shows warnings at the top. The most important ones:

Message	Meaning / fix
The active render pipeline is not URP	Assign a URP asset.
Another GrassCoreWorld is active	Only one world renders – remove the other one.
No enabled species	Add or enable a species in the Species tab.
No grass surface found	Add a terrain (auto-detected) or a Mesh Surface Provider.
Main camera far clip ... is shorter than the grass render distance	Raise the camera's far clip plane or lower the render distance.
... blades were dropped because the instance budget is full	Raise <i>Max Instances</i> or reduce density.
All chunk tile slots are in use	Raise the chunk size or reduce the render distance.

Example Workflow

Finding out why grass is missing in one spot

- 1 *Debug View = Species*: is the spot coloured at all? If not, the surface or a mask removes it.
- 2 Select nearby mask volumes, paths, water volumes and Grass Vegetation Areas – their Scene view gizmos show their influence.
- 3 Check the species' *Placement Filters* (slope, elevation, terrain layers) and *Water Clearance*.
- 4 *Draw Chunks*: is the chunk built (green) or still building (orange)?
- 5 Press *Rebuild All Tiles* after external changes (for example terrain edits from a script).

Performance Notes

- Statistics are collected only while a tool requests them (inspector, overlay, profiler, panel) – closing them removes the small readback cost.
- Gizmos are editor-only.

Common Problems

Symptom	Fix
Nothing renders	Check the World warnings: URP active? Species assigned? Surface found? Layer included in the camera's culling mask?
Grass stops too early	Camera far clip, species <i>Max Distance</i> , quality <i>Render Distance</i> .
Grass only near the camera in edit mode	Streaming follows the Scene view camera in edit mode and the main camera in Play mode (or the <i>Streaming Anchor</i>).
Wind frozen in the Scene view	Enable <i>Always Refresh</i> in the Scene view.
Grass through buildings	Add a Mesh Surface Provider in <i>Blocker</i> mode or an <i>Exclude</i> mask volume.
Burnt / cut areas grow back after a quality change	Runtime edits are cleared when the tile resolution changes; restore them with <code>ImportGrassEdits</code> .
Grass flickers between frames	Two worlds or a camera without depth? Remove duplicate worlds; check the camera's depth texture settings for water.
Grass dark or unlit	Check the material (GrassCore/Grass) and <i>Receive Shadows</i> ; in Deferred, grass is a forward-only pass.
Pink grass	The shader failed to compile or URP is not active – check the Console and the render pipeline asset.
Console: [GrassCore] ... warning on start	The world reports why it is unsupported (no compute shaders, missing resources).

Frequently Asked Questions

Short answers to the questions new GrassCore users ask most often.

What it does

This chapter collects quick answers. Each answer points to the chapter with the details.

How to set it up

Do I have to bake or paint grass? No. Grass is generated on the GPU from the surfaces, species and masks every time a chunk streams in. Nothing is baked or stored per blade ([chapter 1](#)).

Does GrassCore work with the Built-in Render Pipeline or HDRP? No. GrassCore requires Unity 6 and URP 17+ ([chapter 2](#)).

Does it work with Forward+, Deferred and multiple cameras? Yes. In Deferred the grass is drawn in a forward-only pass. Every game camera renders grass while *Render In Game Cameras* is on and its culling mask contains the world's *Layer* ([chapter 4](#)).

How many species can I use? Up to 8 species per world, including species that only grow inside Grass Vegetation Areas. Local overrides of areas do not use species slots ([chapter 9](#)).

Can I use my own grass or flower meshes? Yes – *Blade Source = Custom Meshes* with up to three LODs ([chapter 7](#)). Or use the built-in procedural flowers ([chapter 8](#)).

Can grass grow on meshes instead of terrains? Yes – add a Mesh Surface Provider. Rocks and buildings can remove grass as blockers ([chapter 5](#)).

How do I make paths and roads? With a GrassCore Path: grass removal with falloff, optional terrain layer painting and material decals ([chapter 11](#), [chapter 12](#)).

Important Settings

Which setting should I change first for performance? The quality preset, then *Halving Distance*, then shadows ([chapter 33](#)).

Why is my grass gone after changing the quality preset at runtime? It is not gone – tiles are rebuilt over a few frames. Runtime edits (burnt or cut areas) are cleared when the tile resolution changes; save and re-import them with `ExportGrassEdits()` / `ImportGrassEdits()` ([chapter 18](#)).

Do optional systems cost performance when I do not use them? No. Grass Vegetation Areas, mowing, water surfaces, underwater plants, trees and bushes and tree destruction only allocate resources while they are used.

How many interactors, shockwaves and fires are supported? 64 interactors, 32 shockwaves, 16 rotor downwash sources, 32 fires, 32 glow sources and 32 light waves at the same time ([chapter 38](#)).

Example Workflow

Can I save cut, burnt and mown grass in save games? Yes: `ExportGrassEdits()` / `ImportGrassEdits()` for removed and burnt grass, `ExportMowing()` / `ImportMowing()` for mown grass, `GetFelledTrees()` / `SetTreeFelled()` for felled trees ([chapter 36](#)).

Can AtmosphereCore or my own weather system drive the wind? Yes – implement `IGrassCoreWindSource` (or `IGrassCoreWeatherSource`) and register it with `GrassCoreIntegration` ([chapter 36](#)).

Does GrassCore work with OceanCore or my own water? Yes – implement `IGrassCoreWaterProvider` and register it; grass is removed under that water and gets a wet shoreline ([chapter 36](#)).

Can trees be cut down? Yes, with *Destroyable* tree profiles and the damage API ([chapter 32](#)).

Performance Notes

Does the size of my terrain matter? Hardly. Cost follows the view distance and what is visible, not the size of the world ([chapter 33](#)).

Does the Scene view cost performance? Yes, while it is visible it renders grass too. Measure in the Game view or in a build.

| Common Problems

The grass does not move in the editor. Enable *Always Refresh* in the Scene view ([chapter 13](#)).

My character does not bend the grass. Add a Grass Interactor; interaction runs in Play mode or with *Simulate In Edit Mode* ([chapter 14](#)).

The Setup Wizard opens every time I start Unity. Untick *Show at startup* in the wizard's footer ([chapter 2](#)).

Where do I get help? E-mail Contact@RedicionStudio.com, Discord <https://discord.gg/RkyXNetrR>, or [Tools](#) › [GrassCore](#) › [Support](#).

Runtime API – Short Examples

Everything you can do in the inspector you can also drive from gameplay code. The static `GrassCoreRuntimeAPI` class is the single entry point; integration interfaces connect your own wind, weather and water systems.

What it does

`GrassCoreRuntimeAPI` (namespace `GrassCore`) is a static façade over the active GrassCore World. It hides all GPU details: you pass world positions, radii and strengths, GrassCore takes care of chunks, tiles and buffers. Every call is safe when no world exists – it simply does nothing or returns `false`, `-1` or a default value.

The optional integration layer lives in `GrassCore.Integration`: your own systems implement small interfaces (`IGrassCoreWindSource`, `IGrassCoreWeatherSource`, `IGrassCoreWaterProvider`, `IGrassCoreEnvironmentController`) and register them with the static `GrassCoreIntegration` class. This is also how `OceanCore`, `AtmosphereCore` and `WorldCore` connect – GrassCore has no hard dependency on any of them.

How to set it up

- 1 Add `using GrassCore;` (and `using GrassCore.Integration;` for adapters) to your script.
- 2 Check `GrassCoreRuntimeAPI.IsAvailable` before relying on results – it is `true` when a world is active and the platform supports compute shaders.
- 3 Call the API from `Start`, `Update` or your gameplay events. Do not call it from `Awake` of objects that are created before the world is enabled.

```
using GrassCore;
using UnityEngine;

public class GrassQualityMenu : MonoBehaviour
{
    public void OnQualityChanged(int index) // 0 = Low ... 3 = Ultra
    {
        if (!GrassCoreRuntimeAPI.IsAvailable) return;
        GrassCoreRuntimeAPI.SetQuality((GrassCoreQualityLevel)index);
    }

    public void OnGrassToggled(bool on) => GrassCoreRuntimeAPI.SetRenderingEnabled(on);
    public void OnDensityChanged(float v) => GrassCoreRuntimeAPI.SetDensityMultiplier(v); // 0.05 - 4
    public void OnDistanceChanged(float m) => GrassCoreRuntimeAPI.SetRenderDistance(m); // < 0 = use preset
}
```

Important Settings

API overview

Topic	Members
State	<code>World</code> , <code>IsAvailable</code> , <code>SetRenderingEnabled</code> , <code>IsRenderingEnabled</code> , <code>Statistics</code> , <code>RebuildAll</code> , <code>RefreshRegion(bounds)</code> , <code>OnOriginShifted(offset)</code> , <code>TryGetSurfaceHeight</code>
Quality	<code>SetQuality</code> , <code>GetQuality</code> , <code>SetCustomQuality(profile)</code> , <code>SetDensityMultiplier</code> , <code>SetRenderDistance</code> , <code>SetDebugView</code>
Wind	<code>GetWind</code> , <code>SetWind(state, blendSeconds)</code> , <code>ClearWindOverride</code> , <code>SetWindStrengthMultiplier</code> , <code>SampleWind(position)</code>
Interaction	<code>AddRadialImpulse</code> , <code>AddInteraction(from, to, ...)</code> , <code>ClearInteraction</code> , <code>TriggerShockwave</code> , <code>ApplyDownwash</code>
Grass edits	<code>RemoveGrass</code> , <code>RestoreGrass</code> , <code>ModifyDensity</code> (circle or spline, optional species mask), <code>ClearGrassEdits</code> , <code>ExportGrassEdits</code> , <code>ImportGrassEdits</code>
Mowing	<code>MowGrass</code> , <code>MowGrassLine</code> , <code>RegrowMownGrass</code> , <code>ClearMowing</code> , <code>HasMownGrass</code> , <code>ExportMowing</code> , <code>ImportMowing</code>
Fire	<code>IgniteGrass</code> (circle or spline), <code>TryGetFire</code> , <code>ActiveFireCount</code> , <code>CancelAllFires</code>
Seasons & snow	<code>SetSeason</code> (0–4 or <code>GrassCoreSeason</code>), <code>GetSeason</code> , <code>SetSnowAmount</code> , <code>GetEffectiveSnow</code>

Topic	Members
Colour zones	CreateColorZone, MoveColorZone, ModifyColorZone, RemoveColorZone, TryGetColorZone
Glow	AddGlow, EmitGlowWave
Species & areas	GetSpeciesIndex, SetSpeciesEnabled, SetSpeciesDensity, AddSpecies, GrassAreas, CreateGrassArea, SetGrassAreaSpecies
Water	AddWaterSplash, SetWaterSurfacesEnabled, SetWaterSimulationEnabled, AreWaterSurfacesActive
Trees & bushes	SetTreesEnabled, SetBushesEnabled, RemoveVegetation, BurnVegetation, RestoreVegetation, AddVegetationImpulse, SetVegetation...Enabled (interaction, flowers, fruits, colliders), VegetationInstanceCount, InteractingVegetation
Tree destruction	RaycastTree, TryGetTree, DamageTrees, RestoreAllTrees, GetFelledTrees, SetTreeFelled, SetTreeDestruction...Enabled, FallingTrees, GrassCoreTreeEvents

Integration interfaces (GrassCore.Integration)

Interface	Members	Register with
IGrassCoreWindSource	int WindPriority, bool TryGetWind(out GrassCoreWindState) – the source with the highest priority that returns true wins.	GrassCoreIntegration.RegisterWindSource
IGrassCoreWeatherSource	Wetness (0–1), RainIntensity (0–1), WindMultiplier (1= calm, > 1= storm)	RegisterWeatherSource
IGrassCoreWaterProvider	Bounds WaterBounds, int WaterVersion, bool TryGetWaterHeight(position, out height)	RegisterWaterProvider
IGrassCoreEnvironmentController	bool TryGetQuality(out GrassCoreQualityLevel) – GrassCore follows this quality instead of the world setting.	SetEnvironmentController

Always unregister in OnDisable (UnregisterWindSource, UnregisterWeatherSource, UnregisterWaterProvider, ClearEnvironmentController).

Example Workflow

Explosion: shockwave, burnt grass and shaking trees

```
void Explode(Vector3 p)
{
    GrassCoreRuntimeAPI.TriggerShockwave(p, radius: 14f, strength: 1.4f);
    GrassCoreRuntimeAPI.IgniteGrass(p, radius: 4f);
    GrassCoreRuntimeAPI.CreateColorZone(p, GrassCoreColorZonePreset.Burnt, radius: 6f, fadeInSeconds: 0.5f);
    GrassCoreRuntimeAPI.AddVegetationImpulse(p, radius: 15f, strength: 2f);
    GrassCoreRuntimeAPI.DamageTrees(p, radius: 12f, damage: 150f);
}
```

Building placement: remove grass, restore it on demolition

```
GrassCoreRuntimeAPI.RemoveGrass(building.position, radius: 6f, falloff: 0.2f);
// Only thin out tall grass (species index 2) around the camp:
GrassCoreRuntimeAPI.ModifyDensity(camp, 12f, 0.3f, GrassCoreEditMode.Multiply,
    GrassCoreSpeciesMask.Single(2));
// Later:
GrassCoreRuntimeAPI.RestoreGrass(building.position, 6f);
```

Weather: blend to a storm and back

```
GrassCoreWindState storm = GrassCoreRuntimeAPI.GetWind();
storm.strength = 2.5f;
storm.gustStrength = 1.5f;
storm.turbulence = 1f;
GrassCoreRuntimeAPI.SetWind(storm, blendSeconds: 8f);
// ...
GrassCoreRuntimeAPI.ClearWindOverride(); // back to the world / WindZone wind
```

Seasons over a game year

```
GrassCoreRuntimeAPI.SetSeason(GrassCoreSeason.Autumn, blendSeconds: 30f);
GrassCoreRuntimeAPI.SetSnowAmount(0.8f, blendSeconds: 60f);
```

Save games

```
// Save
GrassCoreEditSnapshot edits = GrassCoreRuntimeAPI.ExportGrassEdits(); // removed / burnt / painted
GrassCoreMowingSnapshot mown = GrassCoreRuntimeAPI.ExportMowing(); // cut heights
var stumps = new List<Vector3>();
GrassCoreRuntimeAPI.GetFelledTrees(stumps); // felled trees
// → write edits, mown and stumps with your save system (the snapshots are [Serializable] data classes)

// Load
GrassCoreRuntimeAPI.ImportGrassEdits(edits); // false + warning if chunk size / tile resolution differ
GrassCoreRuntimeAPI.ImportMowing(mown);
foreach (Vector3 s in stumps) GrassCoreRuntimeAPI.SetTreeFelled(s);
```

Floating origin

```
// Your floating-origin system moved the world so that 'offset' became the new origin:
GrassCoreRuntimeAPI.OnOriginShifted(offset); // placement and edits stay identical
```

Custom wind source (for example your own weather system)

```
using GrassCore;
using GrassCore.Integration;
using UnityEngine;

public class MyWeatherWind : MonoBehaviour, IGrassCoreWindSource
{
    public float strength = 1f;
    public int WindPriority ⇒ 10;

    void OnEnable() ⇒ GrassCoreIntegration.RegisterWindSource(this);
    void OnDisable() ⇒ GrassCoreIntegration.UnregisterWindSource(this);

    public bool TryGetWind(out GrassCoreWindState wind)
    {
        wind = GrassCoreWindState.Default;
        wind.directionDegrees = transform.eulerAngles.y;
        wind.strength = strength;
        return true;
    }
}
```

Custom water provider (your own ocean or lake)

```
public class MyOceanAdapter : MonoBehaviour, IGrassCoreWaterProvider
{
    public float seaLevel = 0f;
    int m_Version;

    public Bounds WaterBounds ⇒ new Bounds(new Vector3(0f, seaLevel, 0f), new Vector3(20000f, 50f, 20000f));
    public int WaterVersion ⇒ m_Version;

    public bool TryGetWaterHeight(Vector3 p, out float h) { h = seaLevel; return true; }

    public void SetSeaLevel(float y) { seaLevel = y; m_Version++; } // affected chunks rebuild

    void OnEnable() ⇒ GrassCoreIntegration.RegisterWaterProvider(this);
    void OnDisable() ⇒ GrassCoreIntegration.UnregisterWaterProvider(this);
}
```

Reading statistics in your own UI

```
void OnEnable() ⇒ GrassCoreRuntimeAPI.World?.RequestStatistics(true);
void OnDisable() ⇒ GrassCoreRuntimeAPI.World?.RequestStatistics(false);

void Update()
{
    GrassCoreStats s = GrassCoreRuntimeAPI.Statistics;
    if (s ≠ null && s.GpuCountersValid)
        label.text = $"{s.InstancesRendered:N0} blades · {s.VisibleChunks} chunks";
}
```

Performance Notes

- API calls queue small GPU stamps or update a few constants. They are cheap and allocation-free – calling them every frame (for example `ApplyDownwash` for a helicopter) is intended.
- `RemoveGrass`, `ModifyDensity` and spline edits write into the affected chunk tiles only; nothing is rebuilt globally.

- `RebuildAll()` rebuilds every tile – use `RefreshRegion(bounds)` after local terrain or mesh changes instead.
- `Statistics` triggers a small asynchronous GPU readback only while someone requests it (`RequestStatistics(true)`). Release the request when your UI is hidden.
- Integration interfaces are queried only when needed: wind and weather sources once per frame, water providers when chunk tiles inside `WaterBounds` are built.

Common Problems

Problem	Solution
Calls have no effect	No active world, or <code>IsAvailable</code> is <code>false</code> (no compute shader support). Call from <code>Start</code> or later.
<code>ImportGrassEdits</code> returns <code>false</code>	The snapshot was saved with a different chunk size or tile resolution (quality preset). Restore the same quality before importing.
Grass under my water does not disappear	Increase <code>WaterVersion</code> after changing the water, and make sure <code>WaterBounds</code> contains the shoreline.
Custom wind is ignored	World › Wind › <i>External Sources</i> is off, another source has a higher <code>WindPriority</code> , <code>TryGetWind</code> returns <code>false</code> – or a <code>SetWind</code> override is active (it takes precedence; call <code>ClearWindOverride</code>).
<code>AddSpecies</code> returns <code>-1</code>	All 8 species slots are used.
Tree handles become invalid	Handles do not survive floating-origin shifts or vegetation rebuilds – fetch them again.

Best Practices

Proven ways to set up GrassCore projects that look great, stay fast on every platform and are easy to maintain in a team.

What it does

This chapter summarises the recommendations from the previous chapters into a checklist for production projects. Follow it when you start a new project and again before you ship.

How to set it up

Project structure

- Keep your own species, vegetation profiles and water plants in a project folder (for example `Assets/MyGame/GrassCore`). Duplicate the sample assets instead of editing them, so updates of GrassCore never overwrite your work.
- Use **one GrassCore World per scene**. Additive scenes should not bring their own world.
- Name species after their look (*Meadow Grass*, *Shore Reeds*) – the name appears in the Profiler and in Grass Vegetation Areas.
- Put paths, mask volumes, water volumes and Grass Vegetation Areas under a common parent object (for example *Environment*) to keep the hierarchy readable.

Workflow order

- 1 World and surfaces – terrain or meshes, quality preset (chapter 4, chapter 5).
- 2 Base species – one or two grass species that cover the whole terrain (chapter 6).
- 3 Variation – Grass Vegetation Areas, flowers, colour zones (chapter 9, chapter 8, chapter 20).
- 4 Structure – paths, masks, water (chapter 11, chapter 10, chapter 22).
- 5 Trees and bushes – Vegetation Layers and Vegetation Areas (chapter 29).
- 6 Life – wind, interactors, effects (chapter 13, chapter 14).
- 7 Performance pass on the target hardware (chapter 33).

Important Settings

Look

Recommendation	Why
Use few base species with strong variation instead of many similar species.	Height, width and colour variation already break up repetition; every species slot costs a draw bucket.
Let flowers grow in Grass Vegetation Areas or with low density.	Flowers are most convincing in clusters; spreading them evenly looks artificial.
Match the grass colour to the terrain layer underneath.	The gap between blades shows the ground – similar colours make the meadow look denser.
Use <i>Far Width Expansion</i> instead of high far-field density.	Wider distant blades fill the image at a fraction of the cost.
Combine the render distance with fog.	The end of the grass disappears naturally.

Performance

Recommendation	Why
Choose the quality preset per platform first, then fine-tune with <i>Customise This Preset</i> .	Each preset sets 14 values (density, distances, LOD, shadows, interaction, memory, geometry, streaming) that fit together.
Keep shadows on LOD0 (or off) and switch them off for sparse species.	Shadow passes repeat the grass draws.

Recommendation	Why
Give small details (flowers, small custom meshes) a short <i>Max Distance</i> .	They are invisible at distance anyway.
Put the grass on its own <i>Layer</i> (World › Quality › Rendering) and remove that layer from the culling mask of helper cameras (minimap, UI, render textures).	Each camera that renders grass runs its own generation and draws.
Use <code>RefreshRegion</code> instead of <code>RebuildAll</code> after scripted changes.	Only the affected chunks are rebuilt.
Release statistics requests when your debug UI is hidden.	Statistics readbacks are only needed while someone looks at them.

Gameplay

Recommendation	Why
One interactor per character; one per wheel for vehicles.	Realistic tracks at minimal cost.
Use <i>Persistent Trail</i> only where trails matter (vehicles, paths).	Normal interaction recovers by itself and needs no bookkeeping.
Save edits, mowing and felled trees with the export functions (chapter 36).	Runtime changes are not stored in the scene.
Use the presets of shockwaves, fires, colour zones and mowers as a starting point.	They are balanced to look right without tuning.

Example Workflow

Shipping checklist

- 1 Open the World inspector – no warnings at the top.
- 2 Play the scene with the **Profiler** window open – *Dropped* = 0, *Tile slots used* below the maximum.
- 3 Test every quality preset on the slowest target device.
- 4 Teleport across the map and drive at full speed – no visible streaming gaps.
- 5 Test a save game round trip with edits, mowing and felled trees.
- 6 Check every camera that should (or should not) render grass.
- 7 Remove the Statistics Overlay and the demo UI from release scenes (or keep them behind a debug switch).
- 8 Build for each platform and check the Console for *[GrassCore]* warnings.

Performance Notes

- Measure in builds or in the Game view – the Scene view also renders grass while it is visible.
- Change one setting at a time and compare the Profiler values before and after.
- Optional systems (areas, mowing, water surfaces, underwater plants, trees and bushes, destruction) only cost something while they are used – enable them where your gameplay needs them, not "just in case".

Common Problems

Problem	Best practice
Updates overwrite my changes	Work on duplicates of the sample assets in your own folder.
Every scene looks different	Share species and profiles between scenes; keep scene-specific changes in Grass Vegetation Areas.
Performance drops late in production	Profile early on the target hardware and keep a budget for blades and vegetation instances.
Team members change the same world	Keep species and profiles as separate assets – they can be edited without touching the scene.

38

PERFORMANCE & REFERENCE

Reference & Support

Every menu command, component, asset type, shader and limit of GrassCore at a glance – plus where to get help.

What it does

This chapter is a quick lookup. Use it to find a menu command, to check which component does what, or to look up a limit before you design a feature around it.

How to set it up

Tools menu – Tools › GrassCore

Command	Action
Setup Wizard	Getting Started, Quick Setup (world, surfaces, species, quality) and Support – including Open Documentation (this PDF).
Create GrassCore World	Creates a configured world in the open scene.
Create Demo Scene	Builds a demo scene: generated terrain, several species, Grass Vegetation Areas, a lake, a path, a playable character, a vehicle and the demo UI.
Create Demo UI	Adds the Canvas demo UI and statistics panel to the open scene.
Add Terrain Providers to All Terrains	Adds a Terrain Surface Provider to every terrain.
Add Mesh Surface / Mesh Blocker to Selection	Grass grows on / is removed by the selected meshes.
Add Interactor / Burn Source / Downwash Source / Glow Source / Mower to Selection	Adds the component to all selected objects.
Profiler	Opens the GrassCore Profiler window (also Window › Analysis › GrassCore Profiler).
Path Painting › Repaint All Paths / Restore Terrain Under All Paths / Restore Backups of Deleted Paths	Terrain layer painting of paths (chapter 12).
Documentation	Selects the Quick Start guide in the Project window.
Support	Opens the Support page of the Setup Wizard.

GameObject menu – GameObject › GrassCore

Command	Creates
GrassCore World	The world (renderer, streaming, all systems).
Path (Spline)	A path that removes grass and can paint the terrain.
Spline Mask Area / Mask Volume	Density masks (box, sphere, cylinder or spline).
Grass Vegetation Area	A spline area with its own species, densities and overrides.
Water Volume / Water › Spline Lake / Water › River	Water that removes grass, wets shores, grows water plants and renders surfaces.
Shockwave Source / Burn Source / Spline Burn Area / Downwash Source (Rotor) / Glow Source / Mower (Cutting Tool)	Effect sources.
Color Zone / Spline Color Zone	Tint, dryness and wetness zones.
Vegetation › Tree & Bush Instances (Manual) / Vegetation Area (Box) / Vegetation Area (Spline)	Trees and bushes.
Statistics Overlay	Text statistics in the Game view.

Assets menu – Assets › Create › GrassCore

Asset	Purpose
Grass Species	Blades, flowers or custom meshes with look, placement, LOD, wind and interaction settings (chapter 6).
Vegetation Profile (Tree or Bush), Tree Profile, Bush Profile	Tree and bush definitions (chapter 25).
Water Plant	Underwater plants and lily pads (chapter 23).

Important Settings

Components

Component	Purpose	Chapter
GrassCore World	Central component: species, quality, streaming, all systems.	chapter 4
Terrain Surface Provider / Mesh Surface Provider	Surfaces grass grows on (or is blocked by).	chapter 5
Grass Vegetation Area	Local species, densities and overrides.	chapter 9
Mask Volume	Include, exclude or scale density.	chapter 10
Path	Grass removal with falloff, terrain painting, decals.	chapter 11
Grass Interactor	Bends and flattens grass, pushes plants, makes ripples.	chapter 14
Shockwave Source / Downwash Source (Rotor)	Explosions and rotor wind.	chapter 15, chapter 16
Grass Mower	Cuts grass to a cut height, emits clippings.	chapter 17
Burn Source	Starts grass fires.	chapter 18
Color Zone	Local colour, dryness and wetness.	chapter 20
Glow Source	Bioluminescent glow and light waves.	chapter 21
Water Volume / Water Interactor	Water, underwater plants, interactive waves.	chapter 22, chapter 24
Vegetation Area / Vegetation Instances (Manual)	Tree and bush placement.	chapter 29
Statistics Overlay	Runtime statistics.	chapter 34

Shaders

Shader	Used for
GrassCore/Grass	Grass blades, custom grass meshes and procedural flowers.
GrassCore/Vegetation	Trees and bushes (procedural and custom meshes).
GrassCore/Water	Water surfaces.
GrassCore/Underwater	Underwater plants and lily pads.
GrassCore/Path Surface	Path decal material.
Hidden/GrassCore/...	Internal: clippings, surface capture, tree debris, vegetation billboards.

Compute shaders and internal shaders are loaded from `Runtime/Resources/GrassCore` – do not move or rename them.

Limits

Feature	Limit
Species per world	8
Grass Vegetation Area override sets per scene	24
Grass Vegetation Areas with overrides	255
LOD levels per species	3
Grass Interactors	64
Shockwaves / rotor downwash sources	32 / 16

Feature	Limit
Fires (burn areas)	32
Colour zones	64
Glow sources / glow waves	32 / 32
Mask volumes per chunk tile / mask textures	32 / 16
Terrain layers used for filters	16
Water volumes with underwater plants	32
Water plant types (per world / per volume)	16 / 8
Spline water shapes per chunk tile	8
Spline shapes baked at once (paths, areas, masks, water, colour zones, burn areas)	256
Interactive water sources per frame	64
Trees and bushes (instances)	4 000 000

Folder structure

Folder	Content
Runtime	Runtime scripts (assembly <i>GrassCoreRuntime</i>), the grass shader and <code>Resources/GrassCore</code> (compute shaders, internal shaders, defaults).
Editor	Inspectors, Setup Wizard, Profiler window, menus (assembly <i>GrassCoreEditor</i>).
Samples/Species	Grass and flower species with materials (Meadow Grass, Tall Grass, Dry Steppe Grass, Shore Reeds, Glowgrass, Wildflowers, tulips, poppies, lavender ...).
Samples/Vegetation	38 tree and bush profiles, including <i>Oak (Destroyable)</i> and <i>Spruce (Destroyable)</i> .
Samples/WaterPlants	Kelp, Lily Pad, Reed, Seagrass, Water Weed.
Samples/Demo	Demo scene builder, demo UI and demo scripts (player, vehicle, axe).
Documentation	This PDF, Quick Start, API reference and architecture notes.

Example Workflow

Looking up a limit before designing a feature

- 1 You plan a battle with many simultaneous explosions.
- 2 Limits: 32 shockwaves at the same time – the oldest wave is replaced when more are triggered.
- 3 For larger battles, combine shockwaves with `AddRadialImpulse` for distant explosions and `CreateColorZone(..., Burnt, ...)` for lasting scorch marks.

Performance Notes

- The limits are fixed buffer sizes chosen so that the per-frame cost stays constant – none of them grows with the size of your world. Optional systems allocate their buffers only when first used.

Common Problems

Most problems are answered in [chapter 34](#) and [chapter 35](#). If yours is not, contact us:

- E-mail:** Contact@RedicionStudio.com · **Discord:** <https://discord.gg/RkyXNetrR> · **In Unity:** [Tools](#) · [GrassCore](#) · [Support](#)
- Asset Store:** <https://assetstore.unity.com/packages/slug/414712>

Please include your Unity and URP version, the GrassCore version (shown in the Setup Wizard and in this document's footer), a screenshot of the World inspector warnings and – if possible – the GrassCore Profiler values. If you enjoy GrassCore, a review on the Asset Store helps us a lot. Thank you!