Infohazard.HyperNav

1.1.5

Generated by Doxygen 1.9.5

1 HyperNav Documentation

1.1 Table of Contents

- HyperNav Documentation
  - Table of Contents
  - Introduction
  - Documentation and Support
  - License
  - Installation
    - Prerequisites
    - Asset Store
    - Package Manager (Git URL or Submodule)
  - Setup
    - General Setup
    - SRP Setup
  - Demos
    - HyperNavDemo (Pathfinding)
    - HyperNavAvoidanceDemo (Pathfinding Avoidance)
    - HyperNavAvoidanceScaleDemo (Many-Agent Standalone Avoidance)
    - HyperNavFloatingOriginDemo (Floating Origin System)
  - Features Guide
    - Volume Setup
    - Pathfinder Setup
    - Agent Setup
    - Spline Agent Setup
    - Avoidance
    - Moving Volumes and Floating Origin

1.2 Introduction

HyperNav is a navmesh-like system for implementing volume-based navigation and 3D obstacle avoidance. Unlike a navmesh, which is bound to walkable surfaces, HyperNav creates volumes in which characters can navigate on all three axes. I developed it for Astral Horizon, an FPS with 6-DOF gameplay, and it is best suited to similar games (where characters with the ability to fly need to navigate enclosed spaces).

1.3 Documentation and Support

`API Docs`

`Tutorial Playlist`

`Discord`

1.4 License

HyperNav uses the standard Unity Asset Store per-seat license for tools.

1.5 Installation

1.5.1 Prerequisites

HyperNav depends on the Infohazard.Core library, which you can get from the Asset Store or as a Package Manager package from `Github`. Regardless of how you install it, you should do so before importing HyperNav. You can see more information about the core library `here`

HyperNav also has three additional package manager dependencies: burst, collections, and editorcoroutines. However, these will automatically be installed when you import it, so no need to do anything here.

1.5.2 Asset Store

The main way to install HyperNav is through the asset store. Just install it as you would any other asset, and make sure you allow Unity to update package manager dependencies.

1.5.3 Package Manager (Git URL or Submodule)

Because HyperNav is a paid asset, the Github repository is not open source. However, if you wish to contribute to HyperNav and have purchased a seat, please feel free to `email me` and I can look into getting you read access to the repository.

1.6 Setup

1.6.1 General Setup

The only setup required beyond installation is to add references to the Infohazard.HyperNav assembly if you are using an assembly definition. If you are using the default assemblies (such as Assembly-CSharp), nothing is needed here.

1.6.2 SRP Setup

If you are using a scriptable render pipeline (URP, HDRP, etc) and wish to run the demos, you will need to upgrade the materials using your render pipeline's material upgrade system. The materials you'll need to upgrade are in `Assets/Plugins/Infohazard/Demos/Infohazard.HyperNav/Materials` and `Assets/Plugins/Infohazard/Demos/Shared Demo Assets/Materials`.

1.7 Demos

The following demo scenes are provided to cover the various features of HyperNav.

1.7.1 HyperNavDemo (Pathfinding)

This scene, located at `Assets/Plugins/Infohazard/Demos/Infohazard.HyperNav/Scenes/↩ HyperNavDemo.unity`, demonstrates how to set up NavVolumes and a NavPathfinder, and use NavAgent and SplineNavAgent to find paths. Run the demo and click anywhere on the right side of the screen to set a destination, and use the WASD keys to rotate the camera on the left side of the screen.

1.7.2 HyperNavAvoidanceDemo (Pathfinding Avoidance)

This scene, located at `Assets/Plugins/Infohazard/Demos/Infohazard.HyperNav/Scenes/↩ HyperNavAvoidanceDemo.unity`, demonstrates how you can use avoidance along with NavAgents to avoid agents getting stuck in a narrow maze. Try turning off avoidance by setting the agents' avoidance weights to zero to see the difference it makes.

1.7.3 HyperNavAvoidanceScaleDemo (Many-Agent Standalone Avoidance)

This scene, located at `Assets/Plugins/Infohazard/Demos/Infohazard.HyperNav/Scenes/↩ HyperNavAvoidanceScaleDemo.unity`, deomonstrates how you can use avoidance without a NavAgent. It also shows that the avoidance system performs well even with a huge number of agents all avoiding one another.

1.7.4 HyperNavFloatingOriginDemo (Floating Origin System)

This scene, located at `Assets/Plugins/Infohazard/Demos/Infohazard.HyperNav/Scenes/↩ HyperNavFloatingOriginDemo.unity`, deomonstrates how you can use HyperNav with a floating origin system. This is useful in very large scenes where you need to shift all objects such that the player remains close to the origin.

1.8 Features Guide

1.8.1 Volume Setup

The first step to setup navigation in your scene is to create a NavVolume, using the menu item Tools > Infohazard > Create > Nav Volume.

1.8.1.1 NavVolume Parameters Configuration There are a lot of parameters in NavVolume, which you can read about in full in the `API Docs`. The most important properties that you will probably want to configure here are:

- Bounds: The boundaries of the volume, which you can edit visually.
- BlockingLayers: The layers that are considered impassible to agents.
- MaxAgentRadius: Paths in this volume must be able to accommodate an agent of this radius.
- VoxelSize: Size of voxel samples when baking, which is a tradeoff between accuracy and performance (both baking and at runtime).

1.8.1.2 Baking Once your parameters are configured, you can bake the NavVolume! Just hit the "Bake" button and wait for it to finish (it should be fairly quick). If the baking is taking a long time, consider breaking your volume up into multiple smaller volumes. By default, when baking is done, you should see the baked results rendered as a blue mesh in the scene view when the volume is selected.

1.8.1.3 External Links If you are using multiple volumes, you most likely will want agents to be able to find paths between them. This is accomplished using external links. Simply use the "Generate External Links" button after all volumes are baked, or, for convenience, "Generate All External Links" to generate links on all loaded NavVolumes. Once the external links are generated, you should see them as green lines in the scene view when a volume is selected.

Note that if you re-bake a volume, you will need to regenerate not only its external links but the external links of all of its neighbors. If you don't do this, you will get links leading to the wrong region, and potentially paths that cross through impassible areas. The "Generate All External Links" button comes in handy here.

1.8.2 Pathfinder Setup

Before you can start pathfinding, you also need to create a NavPathfinder, using the menu item Tools > Infohazard > Create > Nav Pathfinder. You can create a NavPathfinder in each scene that needs one, or you can create one that persists through level loading.

1.8.2.1 NavPathfinder Parameters Configuration Like with volumes, there are a lot of parameters in NavPathfinder, which you can read about in full in the `API Docs`. The most important properties that you will probably want to configure here are:

- PathfindingMode: Where paths are calculated. To enjoy the performance boost of the C# Job System and Burst, I recommend leaving this as Worker Thread (Job).
- MaxConcurrentJobs: How many pathfinding jobs can be running at once. This should be kept fairly low. Additional jobs will be kept in a queue.
- MaxCompletionFrames: How many frames pathfinding jobs are allowed to take to complete. I recommend starting this at three, as that is the max allowed when using high-performance TempJob memory. If your paths are consistently taking longer than three frames (this will print warning messages in the console), increase it as needed.

1.8.3 Agent Setup

The final step to start pathfinding is to set up agents. A NavAgent is the easiest way to start finding paths from code. Simply add the NavAgent to your GameObject, set its Destination, and read its DesiredVelocity.

1.8.3.1 NavAgent Parameters Configuration You can find full parameter documentation in the `API Docs`. The main paremeters you'll want to configure for NavAgent are:

- Acceptance: How close the agent must get to a waypoint before it is considered arrived. If acceptance is too low, the agent may not be able to feasibly reach its destination (depending on your movement system).
- AccelerationEstimate: How fast the agent is able to accelerate. This value is used so that it knows when to slow down. You can set this dynamically if your character's acceleration changes.
- SampleRadius: Radius outside the agent to search for a NavVolume. If no NavVolume is found within this radius, pathfinding cannot occur.

1.8.3.2 Agent Usage With your agent set up, you'll need to reference it in code. Your movement code should set NavAgent.Destination to start finding a path, and use NavAgent.DesiredVelocity to determine the direction and speed to move in. You can check NavAgent.Arrived to see if the agent has reached its destination or not.

1.8.4 Spline Agent Setup

If you wish to use the spline path system to achieve smoother navigation paths, simply replace the NavAgent with a SplineNavAgent. SplineNavAgent provides the same properties as NavAgent (as it is a subclass) with some additional properties.

1.8.4.1 SplineNavAgent Paremeters Configuration SplineNavAgent's parameters are documented in the `API Docs`. In addition to the parameters offered by NavAgent, SplineNavAgent contains the following parameters that you may want to set:

- TangentScale: This value controls the curviness of the path. The default value of 0.5 usually works well.
- MaxAlignmentVelocityDistance: The further the agent drifts from the spline, the more it attempts to move back towards the spline. This is the distance at which the entire DesiredVelocity is devoted to returning to the spline.
- CurvatureOfMaxSlowdown: The agent's DesiredVelocity will slow down on parts of the spline with higher curvature. This is the curvature value at which the slowdown will be the greatest. By turning on DebugPath, you can see the curvature values scaled from green (no curvature) to red (this curvature value) to determine the right value for you.
- MaxCurvatureSlowdown: The multiplier on desired velocity to use when the spline curvature is at Curvature↩ OfMaxSlowdown. At lower curvature values, the velocity multiplier scales from 1 to this value.

1.8.4.2 SplineNavAgent Usage Usage of SplineNavAgent is exactly the same as NavAgent - set the Destination and use the DesiredVelocity.

1.8.5 Avoidance

Whereas NavAgent and SplineNavAgent are used to plan paths between two points, avoidance is used to steer clear of obstacles that might obstruct that path. These obstacles might be moving objects or even other navigating agents, so they can't be baked into the NavVolume. Instead, you should use the obstacle avoidance system.

1.8.5.1 AvoidanceManager Setup In order to use avoidance, there must be an AvoidanceManager present in your scene. To create an AvoidanceManager, use the menu item Tools > Infohazard > Create > Avoidance Manager.

1.8.5.2 AvoidanceManager Parameters Configuration AvoidanceManager's parameters are documented in the `API Docs`. You may want to configure these parameters to achieve optimal performance and accuracy:

- TimeHorizon: How many seconds into the future to look for obstacles to avoid.
- MaxObstaclesConsidered: The most obstacles within range an agent can consider each frame.

1.8.5.3 AvoidanceAgent Setup If you want an object to avoid obstacles, you should add an AvoidanceAgent script. This script needs to know the maximum speed the agent can travel at, the agent's size, and several other properties.

1.8.5.4 AvoidanceAgent Usage To use an AvoidanceAgent from your code, you must supply an input velocity (the velocity the agent wants to move in) and read its AvoidanceVelocity (the velocity it should move in to avoid obstacles). The input velocity is provided via a delegate, allowing you to connected it to whatever source you want.

1.8.5.5 AvoidanceAgent With NavAgent When you want to use avoidance with a NavAgent or SplineNav↩ Agent, the usage is simpler. The NavAgent should have a button to create an AvoidanceAgent on the object and instantly connect it to the NavAgent. The NavAgent will handle the input and output velocity for the AvoidanceAgent, so you can simply read the NavAgent's DesiredVelocity which now includes avoidance.

1.8.5.6 AvoidanceAgent Parameters Configuration AvoidanceAgent's parameters are documented in the `API Docs`. You may want to configure these parameters:

- Radius: How big the agent is.
- MaxSpeed: How fast the agent can move. You may also want to set this dynamically. If you are using a NavAgent, this value can be accessed through the NavAgent's AvoidanceMaxSpeed property.
- AvoidanceWeight: How much effort this agent makes to avoid other agents, relative to their own Avoidance↩ Weight. If all agents make equal effor to avoid each other, just leave this at 1.
- AvoidancePadding: Extra room to leave between this agent and any obstacle it is avoiding.
- AvoidedTags: Tags of obstacles to avoid. By default this is set to all obstacles.

1.8.5.7 AvoidanceObstacle Setup While AvoidanceAgents will automatically avoid each other, you may wish to add additional obstacles that are not agents. This can be done using AvoidanceObstacleBase or any of its subclasses (which determine the current velocity using different methods). Just add one of these scripts to an object and the obstacle is good to go:

- AvoidanceObstacleBase: Does not calculate a velocity, so it is assumed to be stationary. You can inherit from it to add your own velocity calculations.
- SimpleAvoidanceObstacle: Calculates its velocity by measuring delta position / delta time every frame.
- RigidbodyAvoidanceObstacle: Uses the current velocity of an attached Rigidbody component.

1.8.5.8 AvoidanceObstacle Parameters Configuration AvoidanceObstacleBase's parameters are documented in the `API Docs`. Because agents are already obstacles, they share some parameters. The main parameters you'll want to configure are:

- Radius: How big the agent is.
- MaxSpeed: How fast the agent can move. You may also want to set this dynamically.

1.8.6 Moving Volumes and Floating Origin

You may need to occasionally move NavVolumes, either individually or all together. When this occurs, the native data for the volumes needs to be updated. For moving individual volumes, you can enable `AutoDetectMovement` on the volumes and the data updating will be handled automatically. For moving all volumes at once such as in a floating origin system, it will be more performant to use `NavVolume.UpdateAllTransforms()` after they are moved.

1.8.6.1 Moving Volume Limitations In both cases, any pathfinding operations currently underway will automatically be restarted. Any agents that already have paths, however, will need to be manually stopped by calling `Stop()` and the destination re-supplied in shifted space.

Moving volumes does update the external link positions, so two volumes with links to one another that move together will have the links updated correctly. However, if one volume moves relative to another volume that it shares links with, the link positions will stay relative to the originating volume. Links cannot currently be re-baked at runtime.

While individual moving volumes are supported, note that when a volume is moved, any calculating paths will be canceled. So if a volume is moving every frame, the paths will be continuously canceled before they can complete. You can get past this by putting the NavPathfinder in `Main Thread End Of Frame` mode, but that will likely lead to stutters.

2 Changelog

All notable changes to this project will be documented in this file.

The format is based on `Keep a Changelog`, and this project adheres to `Semantic Versioning`.

2.1 [1.1.6] - 2023-6-21

2.1.1 Fixed

- Fixed a case where SplinePath data was not disposed.

2.2 [1.1.5] - 2023-3-10

2.2.1 Fixed

- Better prefab support for baked volumes.
  - Can now bake a volume in prefab mode.
  - Volumes that are prefab instances update their ID but do not unset their data reference.
  - Volumes baked in a prefab are stored in the prefab folder.

2.3 [1.1.4] - 2023-2-24

2.3.1 Added

- Support for moving volumes and floating origin, including a new demo scene.
  - External link positions are now stored in local space, but previously generated world-space links will still function.
  - Individual volumes can auto-detect movement, or you can call NavVolume.UpdateAllTransforms for floating origin.

2.4 [1.1.3] - 2023-1-19

2.4.1 Added

- New baking option EnableMultiQuery which fixes bake issues when VoxelSize is larger than MaxAgentRadius.
  - EnableMultiQuery is enabled by default and should be kept on unless you have a good reason to disable it.
  - Added a new visualization option that will show you what the query looks like while baking.

2.4.2 Fixed

- Fixed a mesh simplification issue that led to "Did not find vertex to clip."
- Fixed a region concavity issue that led to infinite bake time.

2.5 [1.1.2] - 2023-1-18

2.5.1 Fixed

- Fixed errors due to volume native data not being correctly initialized when "Reload Scene" is disabled in the editor settings.
- Fixed errors due to volumes being loaded or unloaded while a path is in progress.

2.6 [1.1.1] - 2022-12-14

2.6.1 Fixed

- Fixed a NullReferenceException that occurred when baking multiple volumes that didn't have their data created yet.

2.7 [1.1.0] - 2022-11-22

2.7.1 Added

- New avoidance system.
  - Avoidance can be used either on its own or with a NavAgent or SplineNavAgent.
  - Any object can be an obstacle that is avoided by agents.
  - Agents avoid obstacles as well as each other.
  - Uses the ORCA algorithm, adapted from the RVO2-3D library from University of North Carolina, which is licensed under Apache 2.0.
- Added an option to help prevent spline paths from entering blocked regions by raycasting the tangents.
- Added a button to bake all active NavVolumes at once.
- SplineNavAgents can now get un-stuck by raycasting to see if they are stopped on blocking triangles.

2.7.2 Changed

- Volume identifier now depends on the scene GUID, so a save-as can no longer cause volumes in two scenes to have the same ID.
  - This causes all volume IDs to change. Upon opening a scene with existing volumes, a prompt to migrate the IDs will appear.
- SplinePath is now a struct and uses native data, no longer requiring managed memory allocations. This also means it's Burst-compatible.
- Improved performance of spline projecting by using Burst.

2.7.3 Fixed

- Fixed several cases where projecting on a spline path would be incorrect.

2.8 [1.0.0] - 2022-11-08

2.8.1 Added

- Initial release, all files and documentation added.

3 Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

4 Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

5 Namespace Documentation

5.1 Infohazard Namespace Reference

5.2 Infohazard.HyperNav Namespace Reference

Classes

- class Avoidance

  Static container that keeps track of all avoidance agents and obstacles.
- class AvoidanceAgent

  Base implementation of IAvoidanceAgent that should work in most scenarios.
- class AvoidanceManager

  Handles calculating the avoidance velocities for all IAvoidanceAgents.
- class AvoidanceObstacleBase

  Base class for obstacles that are not expected to perform avoidance, but are avoided by agents.
- struct Edge

  Represents the indices of an edge (two connected vertices) in an indexed mesh.
- interface IAvoidanceAgent

  Interface for objects that both can be avoided and themselves avoid other obstacles using the avoidance system.
- interface IAvoidanceObstacle

Interface for objects that can be avoided using the avoidance system.

- class NavAgent

  A script that can be used to calculate paths by any entity that needs to use HyperNav for navigation.
- struct NavDataInternalPointers

  References to the NativeArrays allocated for a NativeNavVolumeData.
- class NavExternalLinkData

  A connection from one region to another region in another volume.
- struct NavHit

  Structure that is used to report the nearest point on a NavVolume to a query.
- class NavInternalLinkData

  A connection from one region to another region in the same volume.
- class NavPath

  A completed, valid path.
- class NavPathfinder

  A script used to calculate HyperNav paths.
- struct NavRegionBoundPlane

  A plane forming one of the boundaries of a region.
- class NavRegionData

  The serialized data representing a single region in a NavVolume.
- class NavUtil

  Contains utility methods for working with HyperNav navigation.
- class NavVolume

  A volume of space in which HyperNav pathfinding can occur.
- class NavVolumeData

  The baked data of a NavVolume, saved as an asset.
- struct NavWaypoint

  A waypoint in a completed path.
- class RigidbodyAvoidanceObstacle

  An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity from a Rigidbody.
- class SimpleAvoidanceObstacle

  An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity by measuring its position/time delta.
- class SplineNavAgent

  A script that can be used to calculate smooth paths by any entity that needs to use HyperNav for navigation.
- struct SplinePath

  A spline specialized for path following, created with a NavPath.
- struct SplinePoint

  Represents a point on a spline and the segment that starts with it.
- struct Triangle

  Represents the indices of a triangle (three vertices by a face) in an indexed mesh.

Enumerations

- enum AvoidanceManagerUpdateMode

  The update modes in which AvoidanceManager can operate.
- enum NavPathfindingMode

  The modes in which pathfinding can be executed.
- enum NavWaypointType

  Types of waypoint in a completed path.
- enum NavVolumeVisualizationMode

  The various modes available to generate a preview mesh in the editor for visualization.

Functions

- delegate void HyperNavPathCallback (long id, NavPath path)
  A callback that receives a path when it is complete.

5.2.1 Enumeration Type Documentation

5.2.1.1 AvoidanceManagerUpdateMode `enum Infohazard.HyperNav.AvoidanceManagerUpdateMode`

The update modes in which AvoidanceManager can operate.

5.2.1.2 NavPathfindingMode `enum Infohazard.HyperNav.NavPathfindingMode`

The modes in which pathfinding can be executed.

5.2.1.3 NavVolumeVisualizationMode `enum Infohazard.HyperNav.NavVolumeVisualizationMode`

The various modes available to generate a preview mesh in the editor for visualization.

5.2.1.4 NavWaypointType `enum Infohazard.HyperNav.NavWaypointType`

Types of waypoint in a completed path.

5.2.2 Function Documentation

5.2.2.1 HyperNavPathCallback()

```
delegate void Infohazard.HyperNav.HyperNavPathCallback (
        long id,
        NavPath path )
```

A callback that receives a path when it is complete.

5.3 Infohazard.HyperNav.Editor Namespace Reference

Classes

- struct Fast3DArray

  A data structure equivalent to a three-dimensional int array (int[,,]), but more efficient.
- class MarchingCubesCavityTables

  Tables for determining which cubes and combinations of cubes from the Marching Cubes algorithm will produce concave results.
- class MarchingCubesTables

  Tables used in the Marching Cubes algorithm.
- struct MultiRegionMeshInfo

  A struct used to store mesh data of an in-progress volume mesh.
- class NavEditorUtility

  Utility functions used internally, but you can use them too, I mean I'm not your boss.
- struct NavVolumeBakeProgress

  Represents current bake state of a volume, including progress fraction and current operation display name.
- class NavVolumeBakingUtil

  Contains all the code needed to generate the data for a NavVolume.
- class NavVolumeEditor

  Custom editor for Infohazard.HyperNav.NavVolume.
- class NavVolumeExternalLinkUtil

  Utilities for generating the external links of NavVolumes.
- class ThreadSafeIncrementor

  A value that can be incremented, decremented, or added to in a thread safe way.

5.4 Infohazard.HyperNav.Jobs Namespace Reference

Classes

- struct AvoidanceJob

  Job that calculates the IAvoidanceAgent.AvoidanceVelocity of all IAvoidanceAgents.
- struct NativeAvoidanceObstacleData

  Represents one obstacle (which may be an agent) in the avoidance system.
- struct NativeBounds

  A native-friendly of a bounding box.
- struct NativeHeap

  An implementation of a Min Heap that can be used with jobs and Burst.
- class NativeMathUtility

  Provides math operations that are compatible with Burst.
- struct NativeNavExternalLinkData

  The native-friendly data representing a connection from one region to another region in another volume.
- struct NativeNavHit

  A native-friendly representation of a navigation query result.
- struct NativeNavInternalLinkData

  The native-friendly data representing a connection from one region to another region in the same volume.
- struct NativeNavRegionData

  The native-friendly data representing a single region in a NavVolume.
- struct NativeNavVolumeData

  The baked data of a NavVolume, converted to a form compatible with Burst.

- struct NativeNavWaypoint

  A structure used by the navigation job to return the waypoints of a path.
- struct NativePlane

  A plane constructed using native math types.
- struct NativeRay

  A ray constructed using native math types.
- struct NativeRaycastElement

  A single raycast in a NavMultiRaycastJob.
- struct NavMultiRaycastJob

  Job that performs multiple raycasts in one or more NavVolumes in parallel.
- struct NavPathJob

  Burst-compatible job used to find a HyperNav path.
- struct NavRaycastJob

  Job that performs a single raycast in a NavVolume.
- struct PendingPathNode

  A discovered node in a pending path, which serves as a key into the dictionary of per-node discovered info.
- struct SplineProjectJob

  Job used to find the parameter along a spline that is nearest to the given point.
- struct UnsafeArrayPtr

  This is a simple wrapper for unmanaged memory which bypasses Unity's safety checks. This allows arrays to be nested in other arrays (or in structs contained in arrays). Note that you must keep a reference to the original Native↩ Array, or Unity will detect a memory leak.
- struct VisitedNodeInfo

  The information that has been discovered about a node during pathfinding, which is stored in a table.

Enumerations

- enum NavPathState

  The state of a pathfinding request.

5.4.1 Enumeration Type Documentation

5.4.1.1 NavPathState `enum Infohazard.HyperNav.Jobs.NavPathState`

The state of a pathfinding request.

6 Class Documentation

6.1 Infohazard.HyperNav.Avoidance Class Reference

Static container that keeps track of all avoidance agents and obstacles.

Properties

- static List< IAvoidanceObstacle > AllObstacles = new List<IAvoidanceObstacle>() [get]

  All active obstacles (including agents).
- static List< IAvoidanceAgent > AllAgents = new List<IAvoidanceAgent>() [get]

  All active agents.

6.1.1 Detailed Description

Static container that keeps track of all avoidance agents and obstacles.

6.1.2 Property Documentation

6.1.2.1 AllAgents `List<IAvoidanceAgent> Infohazard.HyperNav.Avoidance.AllAgents = new List<IAvoidanceAgent>( [static], [get]`

All active agents.

6.1.2.2 AllObstacles `List<IAvoidanceObstacle> Infohazard.HyperNav.Avoidance.AllObstacles = new List<IAvoidanceObstacle>() [static], [get]`

All active obstacles (including agents).

The documentation for this class was generated from the following file:

- Runtime/Avoidance/Avoidance.cs

6.2 Infohazard.HyperNav.AvoidanceAgent Class Reference

Base implementation of IAvoidanceAgent that should work in most scenarios.

Public Member Functions

- void UpdateAvoidanceVelocity (Vector3 newAvoidance)

  Called by the system to update AvoidanceVelocity.

Parameters

newAvoidance	*New avoidance velocity*

Protected Member Functions

- override void OnEnable ()

  Resets desired velocity and adds self to list of all obstacles.
- override void OnDisable ()

  Removes self from list of all obstacles.

Properties

- virtual float AvoidanceWeight `[get, set]`

  How much effort the agent will take to avoid obstacles and other agents.
- virtual float AvoidancePadding `[get, set]`

  How much extra space to leave when avoiding obstacles.
- virtual bool DebugAvoidance `[get, set]`

  Whether to draw debugging information in the scene view.
- TagMask AvoidedTags `[get, set]`

  Tags of objects that the agent will try to avoid.
- Func< Vector3 > InputVelocityFunc `[get, set]`

  Function used to calculate InputVelocity.
- override Vector3 InputVelocity `[get]`

  The object's desired (or actual) velocity.
- virtual bool IsActive `[get, set]`

  Whether the agent should actively avoid obstacles. If false, will still behave as an obstacle.
- virtual Vector3 AvoidanceVelocity `[get]`

  The velocity the agent should have in order to avoid collisions with obstacles and other agents.
- virtual Vector3 NormalizedAvoidanceVelocity `[get]`

  Avoidance velocity divided by max speed, so it is in [0, 1] range.

Events

- Action< Vector3 > AvoidanceUpdated

  Invoked when avoidance is updated.

Private Attributes

- float _avoidanceWeight = 1

  (Serialized) How much effort the agent will take to avoid obstacles and other agents.
- float _avoidancePadding = 0.1f

  (Serialized) How much extra space to leave when avoiding obstacles.
- bool _debugAvoidance = false

  (Serialized) Whether to draw debugging information in the scene view.
- TagMask _avoidedTags = ∼0

  (Serialized) Tags of objects that the agent will try to avoid.

6.2.1 Detailed Description

Base implementation of IAvoidanceAgent that should work in most scenarios.

It gets its desired velocity from a delegate, so you can point it to whatever system you need.

6.2.2 Member Function Documentation

6.2.2.1 OnDisable() `override void Infohazard.HyperNav.AvoidanceAgent.OnDisable ( ) [protected], [virtual]`

Removes self from list of all obstacles.

Reimplemented from Infohazard.HyperNav.AvoidanceObstacleBase.

6.2.2.2 OnEnable() `override void Infohazard.HyperNav.AvoidanceAgent.OnEnable ( ) [protected], [virtual]`

Resets desired velocity and adds self to list of all obstacles.

Reimplemented from Infohazard.HyperNav.AvoidanceObstacleBase.

6.2.2.3 UpdateAvoidanceVelocity()

```
void Infohazard.HyperNav.AvoidanceAgent.UpdateAvoidance↩
Velocity (
             Vector3 newAvoidance )
```

Called by the system to update AvoidanceVelocity.

Parameters

newAvoidance	New avoidance velocity

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.3 Member Data Documentation

6.2.3.1 _avoidancePadding `float Infohazard.HyperNav.AvoidanceAgent._avoidancePadding = 0.1f [private]`

(Serialized) How much extra space to leave when avoiding obstacles.

6.2.3.2 _avoidanceWeight `float Infohazard.HyperNav.AvoidanceAgent._avoidanceWeight = 1 [private]`

(Serialized) How much effort the agent will take to avoid obstacles and other agents.

6.2.3.3 _avoidedTags `TagMask Infohazard.HyperNav.AvoidanceAgent._avoidedTags = ∼0 [private]`

(Serialized) Tags of objects that the agent will try to avoid.

6.2.3.4 _debugAvoidance `bool Infohazard.HyperNav.AvoidanceAgent._debugAvoidance = false [private]`

(Serialized) Whether to draw debugging information in the scene view.

6.2.4 Property Documentation

6.2.4.1 AvoidancePadding `virtual float Infohazard.HyperNav.AvoidanceAgent.AvoidancePadding [get], [set], [add]`

How much extra space to leave when avoiding obstacles.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.2 AvoidanceVelocity `virtual Vector3 Infohazard.HyperNav.AvoidanceAgent.AvoidanceVelocity [get]`

The velocity the agent should have in order to avoid collisions with obstacles and other agents.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.3 AvoidanceWeight `virtual float Infohazard.HyperNav.AvoidanceAgent.AvoidanceWeight [get], [set]`

How much effort the agent will take to avoid obstacles and other agents.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.4 AvoidedTags `TagMask Infohazard.HyperNav.AvoidanceAgent.AvoidedTags [get], [set]`

Tags of objects that the agent will try to avoid.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.5 DebugAvoidance `virtual bool Infohazard.HyperNav.AvoidanceAgent.DebugAvoidance [get], [set]`

Whether to draw debugging information in the scene view.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.6 InputVelocity `override Vector3 Infohazard.HyperNav.AvoidanceAgent.InputVelocity [get]`

The object's desired (or actual) velocity.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.2.4.7 InputVelocityFunc `Func<Vector3> Infohazard.HyperNav.AvoidanceAgent.InputVelocityFunc [get], [set]`

Function used to calculate InputVelocity.

6.2.4.8 IsActive `virtual bool Infohazard.HyperNav.AvoidanceAgent.IsActive [get], [set]`

Whether the agent should actively avoid obstacles. If false, will still behave as an obstacle.

Implements Infohazard.HyperNav.IAvoidanceAgent.

6.2.4.9 NormalizedAvoidanceVelocity `virtual Vector3 Infohazard.HyperNav.AvoidanceAgent.Normalized←AvoidanceVelocity [get]`

Avoidance velocity divided by max speed, so it is in [0, 1] range.

6.2.5 Event Documentation

6.2.5.1 AvoidanceUpdated `Action<Vector3> Infohazard.HyperNav.AvoidanceAgent.AvoidanceUpdated`

Invoked when avoidance is updated.

The documentation for this class was generated from the following file:

- Runtime/Avoidance/AvoidanceAgent.cs

6.3 Infohazard.HyperNav.Jobs.AvoidanceJob Struct Reference

Job that calculates the IAvoidanceAgent.AvoidanceVelocity of all IAvoidanceAgents.

Public Member Functions

- void Execute (int index)

  Run on a single agent index.
- float3 OrcaAvoidance (int agentIndex, ref NativeArray< NativePlane > agentTempPlanes, ref NativeArray< NativePlane > agentTempProjPlanes)

  Perform avoidance calculation for an agent using the ORCA (optimal reciprocal collision avoidance) algorithm.

Public Attributes

- NativeArray< int > AgentIndices

  Indices in the Obstacles array that are agents that need updating.
- NativeArray< NativeAvoidanceObstacleData > Obstacles

  All obstacles in the world that agents must consider.
- int ObstacleCount

  Number of valid obstacles in the Obstacles array.
- int MaxObstaclesConsidered

  The maximum number of obstacles each agent will consider.
- float DeltaTime

  How much time has passed since the last avoidance update.
- float TimeHorizon

  How far in the future to look when considering avoidance.
- NativeArray< float3 > AvoidanceVelocities

  The calculated avoidance velocity for each agent in AgentIndices.
- NativeArray< NativePlane > TempPlanes

  Used to store the obstacle planes for performing linear programming.
- NativeArray< NativePlane > TempProjPlanes

  Used to store a subset of obstacle planes for linear programming in four dimensions.

6.3.1 Detailed Description

Job that calculates the IAvoidanceAgent.AvoidanceVelocity of all IAvoidanceAgents.

6.3.2 Member Function Documentation

6.3.2.1 Execute()

```
void Infohazard.HyperNav.Jobs.AvoidanceJob.Execute (
        int index )
```

Run on a single agent index.

Parameters

index	The index in the AgentIndices array.

6.3.2.2 OrcaAvoidance() `float3 Infohazard.HyperNav.Jobs.AvoidanceJob.OrcaAvoidance (`

```
int agentIndex,
ref NativeArray< NativePlane > agentTempPlanes,
ref NativeArray< NativePlane > agentTempProjPlanes )
```

Perform avoidance calculation for an agent using the ORCA (optimal reciprocal collision avoidance) algorithm.

Adapted from the RVO2-3D library from University of North Carolina, which is licensed under Apache 2.0.

Parameters

agentIndex	Index of the agent in the Obstacles array.
agentTempPlanes	Array to store temp planes for linear programming.
agentTempProjPlanes	Array to store a subset of temp planes for 4D linear programming.

Returns

The calculated best velocity for the agent.

6.3.3 Member Data Documentation

6.3.3.1 AgentIndices `NativeArray<int> Infohazard.HyperNav.Jobs.AvoidanceJob.AgentIndices`

Indices in the Obstacles array that are agents that need updating.

6.3.3.2 AvoidanceVelocities `NativeArray<float3> Infohazard.HyperNav.Jobs.AvoidanceJob.Avoidance←Velocities`

The calculated avoidance velocity for each agent in AgentIndices.

6.3.3.3 DeltaTime `float Infohazard.HyperNav.Jobs.AvoidanceJob.DeltaTime`

How much time has passed since the last avoidance update.

6.3.3.4 MaxObstaclesConsidered `int Infohazard.HyperNav.Jobs.AvoidanceJob.MaxObstaclesConsidered`

The maximum number of obstacles each agent will consider.

6.3.3.5 ObstacleCount `int Infohazard.HyperNav.Jobs.AvoidanceJob.ObstacleCount`

Number of valid obstacles in the Obstacles array.

6.3.3.6 Obstacles `NativeArray<NativeAvoidanceObstacleData> Infohazard.HyperNav.Jobs.AvoidanceJob.Obstacles`

All obstacles in the world that agents must consider.

6.3.3.7 TempPlanes `NativeArray<NativePlane> Infohazard.HyperNav.Jobs.AvoidanceJob.TempPlanes`

Used to store the obstacle planes for performing linear programming.

6.3.3.8 TempProjPlanes `NativeArray<NativePlane> Infohazard.HyperNav.Jobs.AvoidanceJob.TempProjPlanes`

Used to store a subset of obstacle planes for linear programming in four dimensions.

6.3.3.9 TimeHorizon `float Infohazard.HyperNav.Jobs.AvoidanceJob.TimeHorizon`

How far in the future to look when considering avoidance.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/AvoidanceJob.cs

6.4 Infohazard.HyperNav.AvoidanceManager Class Reference

Handles calculating the avoidance velocities for all IAvoidanceAgents.

Public Member Functions

- virtual void UpdateAvoidance (float deltaTime)

  Update the avoidance of all agents. Can be called manually if UpdateMode is set to manual.

Protected Member Functions

- virtual void OnEnable ()

  Schedule coroutine when the manager is enabled.
- void OnDisable ()

  Dispose data structures when disabled. Coroutine will stop automatically.

Properties

- AvoidanceManagerUpdateMode UpdateMode `[get, set]`

  When to update the avoidance velocities of agents.
- float TimeHorizon `[get, set]`

  How far in the future to look when avoiding collisions.
- int MaxObstaclesConsidered `[get, set]`

  The maximum number of obstacles that can be considered by each agent for avoidance.
- float DataGrowRatio `[get, set]`

  How much to grow data structures by when they are not large enough.
- bool UseJob `[get, set]`

  Whether to use the C# Job System/Burst Compiler or to just run updates on the main thread.

Private Attributes

- AvoidanceManagerUpdateMode _updateMode

  (Serialized) When to update the avoidance velocities of agents.
- float _timeHorizon = 5

  (Serialized) How far in the future to look when avoiding collisions.
- int _maxObstaclesConsidered = 10

  (Serialized) The maximum number of obstacles that can be considered by each agent for avoidance.
- float _dataGrowRatio = 1.2f

  (Serialized) How much to grow data structures by when they are not large enough.
- bool _useJob = true

  (Serialized) Whether to use the C# Job System/Burst Compiler or to just run updates on the main thread.

6.4.1 Detailed Description

Handles calculating the avoidance velocities for all IAvoidanceAgents.

There should only be one AvoidanceManager active at a time, as it will handle all agents together. This is done using the C# Job System and Burst compiler to calculate avoidance very quickly. If it is still not fast enough, try changing the TimeHorizon and/or MaxObstaclesConsidered values.

6.4.2 Member Function Documentation

6.4.2.1 OnDisable() `void Infohazard.HyperNav.AvoidanceManager.OnDisable ( ) [protected]`

Dispose data structures when disabled. Coroutine will stop automatically.

6.4.2.2 OnEnable() `virtual void Infohazard.HyperNav.AvoidanceManager.OnEnable ( ) [protected], [virtual]`

Schedule coroutine when the manager is enabled.

6.4.2.3 UpdateAvoidance() `virtual void Infohazard.HyperNav.AvoidanceManager.UpdateAvoidance ( float deltaTime ) [virtual]`

Update the avoidance of all agents. Can be called manually if UpdateMode is set to manual.

Parameters

deltaTime	Time delta since last call.

6.4.3 Member Data Documentation

6.4.3.1 _dataGrowRatio `float Infohazard.HyperNav.AvoidanceManager._dataGrowRatio = 1.2f [private]`

(Serialized) How much to grow data structures by when they are not large enough.

Higher values reduce the number of allocations if agent count is steadily growing, but may also lead to wasted memory.

6.4.3.2 _maxObstaclesConsidered `int Infohazard.HyperNav.AvoidanceManager._maxObstaclesConsidered = 10 [private]`

(Serialized) The maximum number of obstacles that can be considered by each agent for avoidance.

This value caps the number of avoidance calculations per agent. Generally it should be equal to the max number of obstacles you expect to be within TimeHorizon ∗ IAvoidanceObstacle.MaxSpeed of an agent.

6.4.3.3 _timeHorizon `float Infohazard.HyperNav.AvoidanceManager._timeHorizon = 5 [private]`

(Serialized) How far in the future to look when avoiding collisions.

A lower value reduces the number of calculations per agent, with the drawback of being able to plan less far ahead.

6.4.3.4 _updateMode `AvoidanceManagerUpdateMode Infohazard.HyperNav.AvoidanceManager._updateMode [private]`

(Serialized) When to update the avoidance velocities of agents.

6.4.3.5 _useJob `bool Infohazard.HyperNav.AvoidanceManager._useJob = true [private]`

(Serialized) Whether to use the C# Job System/Burst Compiler or to just run updates on the main thread.

6.4.4 Property Documentation

6.4.4.1 DataGrowRatio `float Infohazard.HyperNav.AvoidanceManager.DataGrowRatio [get], [set]`

How much to grow data structures by when they are not large enough.

Higher values reduce the number of allocations if agent count is steadily growing, but may also lead to wasted memory.

6.4.4.2 MaxObstaclesConsidered `int Infohazard.HyperNav.AvoidanceManager.MaxObstaclesConsidered [get], [set]`

The maximum number of obstacles that can be considered by each agent for avoidance.

This value caps the number of avoidance calculations per agent. Generally it should be equal to the max number of obstacles you expect to be within TimeHorizon $*$ IAvoidanceObstacle.MaxSpeed of an agent.

6.4.4.3 TimeHorizon `float Infohazard.HyperNav.AvoidanceManager.TimeHorizon [get], [set]`

How far in the future to look when avoiding collisions.

A lower value reduces the number of calculations per agent, with the drawback of being able to plan less far ahead.

6.4.4.4 UpdateMode `AvoidanceManagerUpdateMode Infohazard.HyperNav.AvoidanceManager.UpdateMode [get], [set]`

When to update the avoidance velocities of agents.

6.4.4.5 UseJob `bool Infohazard.HyperNav.AvoidanceManager.UseJob [get], [set]`

Whether to use the C# Job System/Burst Compiler or to just run updates on the main thread.

The documentation for this class was generated from the following file:

- Runtime/Avoidance/AvoidanceManager.cs

6.5 Infohazard.HyperNav.AvoidanceObstacleBase Class Reference

Base class for obstacles that are not expected to perform avoidance, but are avoided by agents.

Protected Member Functions

- virtual void OnEnable ()

    Resets desired velocity and adds self to list of all obstacles.
- virtual void OnDisable ()

    Removes self from list of all obstacles.

Properties

- virtual float MaxSpeed `[get, set]`

    Maximum speed the object can travel at.
- virtual float Radius `[get, set]`

    Radius of the object from its position.
- virtual Vector3 Position `[get]`

    World-space position of the object.
- virtual Vector3 InputVelocity `[get]`

    The object's desired (or actual) velocity.
- TagMask TagMask `[get, private set]`

    Tag of the object for matching agents' IAvoidanceAgent.AvoidedTags.

Private Attributes

- float _radius = 0.5f

    (Serialized) Radius of the obstacle.
- float _maxSpeed = 0

    (Serialized) Maximum speed the obstacle can travel at.

6.5.1 Detailed Description

Base class for obstacles that are not expected to perform avoidance, but are avoided by agents.

Objects that do perform avoidance (which are obstacles as well) should inherit from AvoidanceAgent instead, which is a derived class of AvoidanceObstacleBase.

6.5.2 Member Function Documentation

6.5.2.1 OnDisable() `virtual void Infohazard.HyperNav.AvoidanceObstacleBase.OnDisable ( ) [protected], [virtual]`

Removes self from list of all obstacles.

Reimplemented in Infohazard.HyperNav.AvoidanceAgent.

6.5.2.2 OnEnable() `virtual void Infohazard.HyperNav.AvoidanceObstacleBase.OnEnable ( ) [protected], [virtual]`

Resets desired velocity and adds self to list of all obstacles.

Reimplemented in Infohazard.HyperNav.AvoidanceAgent, and Infohazard.HyperNav.SimpleAvoidanceObstacle.

6.5.3 Member Data Documentation

6.5.3.1 _maxSpeed `float Infohazard.HyperNav.AvoidanceObstacleBase._maxSpeed = 0 [private]`

(Serialized) Maximum speed the obstacle can travel at.

6.5.3.2 _radius `float Infohazard.HyperNav.AvoidanceObstacleBase._radius = 0.5f [private]`

(Serialized) Radius of the obstacle.

6.5.4 Property Documentation

6.5.4.1 InputVelocity `virtual Vector3 Infohazard.HyperNav.AvoidanceObstacleBase.InputVelocity [get]`

The object's desired (or actual) velocity.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.5.4.2 MaxSpeed `virtual float Infohazard.HyperNav.AvoidanceObstacleBase.MaxSpeed [get], [set]`

Maximum speed the object can travel at.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.5.4.3 Position `virtual Vector3 Infohazard.HyperNav.AvoidanceObstacleBase.Position [get]`

World-space position of the object.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.5.4.4 Radius `virtual float Infohazard.HyperNav.AvoidanceObstacleBase.Radius [get], [set]`

Radius of the object from its position.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.5.4.5 TagMask `TagMask Infohazard.HyperNav.AvoidanceObstacleBase.TagMask [get], [private set]`

Tag of the object for matching agents' IAvoidanceAgent.AvoidedTags.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

The documentation for this class was generated from the following file:

- Runtime/Avoidance/AvoidanceObstacleBase.cs

6.6 Infohazard.HyperNav.Edge Struct Reference

Represents the indices of an edge (two connected vertices) in an indexed mesh.

Public Member Functions

- Edge (int vertex1, int vertex2)

  Construct a new Edge with the given indices.
- override bool Equals (object obj)

  Compare to another object.
- bool Equals (Edge other)

  Compare to another Edge.
- override int GetHashCode ()

  Get integer for use with hash table.

Properties

- int Vertex1 `[get]`

  First vertex index, which is the lower of the two.
- int Vertex2 `[get]`

  Second vertex index, which is the higher of the two.

Private Attributes

- int _minVertex

  (Serialized) First vertex index, which is the lower of the two.
- int _maxVertex

  (Serialized) Second vertex index, which is the higher of the two.

6.6.1 Detailed Description

Represents the indices of an edge (two connected vertices) in an indexed mesh.

The same Edge will be created regardless of the order in which indices are fed to the constructor.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Edge()

```
Infohazard.HyperNav.Edge.Edge (
    int vertex1,
    int vertex2 )
```

Construct a new Edge with the given indices.

The order of the indices doesn't matter; the same Edge is constructed either way. The indices cannot be the same.

Parameters

vertex1	First vertex index.
vertex2	Second vertex index.

6.6.3 Member Function Documentation

6.6.3.1 Equals() [1/2]

```
bool Infohazard.HyperNav.Edge.Equals (
    Edge other )
```

Compare to another Edge.

Parameters

other	Edge to compare to.

Returns

Whether the two edges are equal.

6.6.3.2 Equals() [2/2]

```
override bool Infohazard.HyperNav.Edge.Equals (
    object obj )
```

Compare to another object.

Parameters

obj	Object to compare to.

Returns

Whether the two objects are equal.

6.6.3.3 GetHashCode() `override int Infohazard.HyperNav.Edge.GetHashCode ( )`

Get integer for use with hash table.

Returns

Integer hash code.

6.6.4 Member Data Documentation

6.6.4.1 _maxVertex `int Infohazard.HyperNav.Edge._maxVertex [private]`

(Serialized) Second vertex index, which is the higher of the two.

6.6.4.2 _minVertex `int Infohazard.HyperNav.Edge._minVertex [private]`

(Serialized) First vertex index, which is the lower of the two.

6.6.5 Property Documentation

6.6.5.1 Vertex1 `int Infohazard.HyperNav.Edge.Vertex1 [get]`

First vertex index, which is the lower of the two.

6.6.5.2 Vertex2 `int Infohazard.HyperNav.Edge.Vertex2 [get]`

Second vertex index, which is the higher of the two.

The documentation for this struct was generated from the following file:

- Runtime/Utility/Edge.cs

6.7 Infohazard.HyperNav.Editor.Fast3DArray Struct Reference

A data structure equivalent to a three-dimensional int array (int[„]), but more efficient.

Public Member Functions

- Fast3DArray (int sizeX, int sizeY, int sizeZ)

  Construct a new Fast3DArray with the given dimensions.
- bool IsOneOf (int x, int y, int z, int option1, int option2)

  Return true if the element at [x, y, z] is either option1 or option2.

Public Attributes

- readonly int SizeX

  Size of first dimension.
- readonly int SizeY

  Size of second dimension.
- readonly int SizeZ

  Size of third dimension.

Properties

- int this[int x, int y, int z] `[get, set]`

  Get or set the value at given coordinates.

6.7.1 Detailed Description

A data structure equivalent to a three-dimensional int array (int[„]), but more efficient.

Unlike an int[„], this type does not perform bounds checking on each dimension. The performance is equivalent to using a single-dimension array and doing the index math yourself.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Fast3DArray()

```
Infohazard.HyperNav.Editor.Fast3DArray.Fast3DArray (
        int sizeX,
        int sizeY,
        int sizeZ )
```

Construct a new Fast3DArray with the given dimensions.

Parameters

sizeX	Size of first dimension.
sizeY	Size of second dimension.
sizeZ	Size of third dimension.

6.7.3 Member Function Documentation

6.7.3.1 IsOneOf()

```
bool Infohazard.HyperNav.Editor.Fast3DArray.IsOneOf (
            int x,
            int y,
            int z,
            int option1,
            int option2 )
```

Return true if the element at [x, y, z] is either option1 or option2.

Parameters

x	First coordinate.
y	Second coordinate.
z	Third coordinate.
option1	First option to check equality.
option2	Second option to check equality.

Returns

If the value at the given coordinates is equal to either option1 or option2.

6.7.4 Member Data Documentation

6.7.4.1 SizeX

`readonly int Infohazard.HyperNav.Editor.Fast3DArray.SizeX`

Size of first dimension.

6.7.4.2 SizeY

`readonly int Infohazard.HyperNav.Editor.Fast3DArray.SizeY`

Size of second dimension.

6.7.4.3 SizeZ

`readonly int Infohazard.HyperNav.Editor.Fast3DArray.SizeZ`

Size of third dimension.

6.7.5 Property Documentation

6.7.5.1 this[int x, int y, int z]

`int Infohazard.HyperNav.Editor.Fast3DArray.this[int x, int y, int z] [get], [set]`

Get or set the value at given coordinates.

Parameters

x	First coordinate.
y	Second coordinate.
z	Third coordinate.

The documentation for this struct was generated from the following file:

- Editor/Fast3DArray.cs

6.8 Infohazard.HyperNav.IAvoidanceAgent Interface Reference

Interface for objects that both can be avoided and themselves avoid other obstacles using the avoidance system.

Public Member Functions

- void UpdateAvoidanceVelocity (Vector3 newAvoidance)

  Called by the system to update AvoidanceVelocity.

Properties

- float AvoidanceWeight `[get]`

  How much effort the agent will take to avoid obstacles and other agents.
- float AvoidancePadding `[get]`

  How much extra space to leave when avoiding obstacles.
- Vector3 AvoidanceVelocity `[get]`

  The velocity the agent should have in order to avoid collisions with obstacles and other agents.
- bool IsActive `[get]`

  Whether the agent should actively avoid obstacles. If false, will still behave as an obstacle.
- bool DebugAvoidance `[get]`

  Whether to draw debugging information in the scene view.
- TagMask AvoidedTags `[get]`

  Tags of objects that the agent will try to avoid.

6.8.1 Detailed Description

Interface for objects that both can be avoided and themselves avoid other obstacles using the avoidance system.

6.8.2 Member Function Documentation

6.8.2.1 UpdateAvoidanceVelocity()

```
void Infohazard.HyperNav.IAvoidanceAgent.UpdateAvoidance↵
Velocity (
            Vector3 newAvoidance )
```

Called by the system to update AvoidanceVelocity.

Parameters

newAvoidance	New avoidance velocity

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3 Property Documentation

6.8.3.1 AvoidancePadding `float Infohazard.HyperNav.IAvoidanceAgent.AvoidancePadding [get]`

How much extra space to leave when avoiding obstacles.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3.2 AvoidanceVelocity `Vector3 Infohazard.HyperNav.IAvoidanceAgent.AvoidanceVelocity [get]`

The velocity the agent should have in order to avoid collisions with obstacles and other agents.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3.3 AvoidanceWeight `float Infohazard.HyperNav.IAvoidanceAgent.AvoidanceWeight [get]`

How much effort the agent will take to avoid obstacles and other agents.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3.4 AvoidedTags `TagMask Infohazard.HyperNav.IAvoidanceAgent.AvoidedTags [get]`

Tags of objects that the agent will try to avoid.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3.5 DebugAvoidance `bool Infohazard.HyperNav.IAvoidanceAgent.DebugAvoidance [get]`

Whether to draw debugging information in the scene view.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

6.8.3.6 IsActive `bool Infohazard.HyperNav.IAvoidanceAgent.IsActive [get]`

Whether the agent should actively avoid obstacles. If false, will still behave as an obstacle.

Implemented in Infohazard.HyperNav.AvoidanceAgent.

The documentation for this interface was generated from the following file:

- Runtime/Avoidance/IAvoidanceAgent.cs

6.9 Infohazard.HyperNav.IAvoidanceObstacle Interface Reference

Interface for objects that can be avoided using the avoidance system.

Properties

- Vector3 Position `[get]`

  World-space position of the object.
- Vector3 InputVelocity `[get]`

  The object's desired (or actual) velocity.
- float MaxSpeed `[get]`

  Maximum speed the object can travel at.
- float Radius `[get]`

  Radius of the object from its position.
- TagMask TagMask `[get]`

  Tag of the object for matching agents' IAvoidanceAgent.AvoidedTags.

6.9.1 Detailed Description

Interface for objects that can be avoided using the avoidance system.

6.9.2 Property Documentation

6.9.2.1 InputVelocity `Vector3 Infohazard.HyperNav.IAvoidanceObstacle.InputVelocity [get]`

The object's desired (or actual) velocity.

Implemented in Infohazard.HyperNav.AvoidanceAgent, Infohazard.HyperNav.AvoidanceObstacleBase, Infohazard.HyperNav.Rigidboc and Infohazard.HyperNav.SimpleAvoidanceObstacle.

6.9.2.2 MaxSpeed `float Infohazard.HyperNav.IAvoidanceObstacle.MaxSpeed [get]`

Maximum speed the object can travel at.

Implemented in Infohazard.HyperNav.AvoidanceObstacleBase.

6.9.2.3 Position `Vector3 Infohazard.HyperNav.IAvoidanceObstacle.Position [get]`

World-space position of the object.

Implemented in Infohazard.HyperNav.AvoidanceObstacleBase.

6.9.2.4 Radius `float Infohazard.HyperNav.IAvoidanceObstacle.Radius [get]`

Radius of the object from its position.

Implemented in Infohazard.HyperNav.AvoidanceObstacleBase.

6.9.2.5 TagMask `TagMask Infohazard.HyperNav.IAvoidanceObstacle.TagMask [get]`

Tag of the object for matching agents' IAvoidanceAgent.AvoidedTags.

Implemented in Infohazard.HyperNav.AvoidanceObstacleBase.

The documentation for this interface was generated from the following file:

- Runtime/Avoidance/IAvoidanceObstacle.cs

6.10 Infohazard.HyperNav.Editor.MarchingCubesCavityTables Class Reference

Tables for determining which cubes and combinations of cubes from the Marching Cubes algorithm will produce concave results.

Static Public Attributes

- static readonly bool[ ] CubesWithInternalCavities

  This table maps a cube index (the same index used in MarchingCubesTables.TriTable to a boolean, which indicates whether that cube contains internal concavities.
- static readonly int[ ][ ][ ] CubeConcaveNeighbors

  This table maps a cube index to another table, which maps a direction index to a list of other cube indices, which, when they are adjacent in that direction, will create a concavity.

6.10.1 Detailed Description

Tables for determining which cubes and combinations of cubes from the Marching Cubes algorithm will produce concave results.

This file is auto-generated but should never change.

6.10.2 Member Data Documentation

6.10.2.1 CubeConcaveNeighbors `readonly int [ ][ ][ ] Infohazard.HyperNav.Editor.MarchingCubes↩ CavityTables.CubeConcaveNeighbors [static]`

This table maps a cube index to another table, which maps a direction index to a list of other cube indices, which, when they are adjacent in that direction, will create a concavity.

index1 = cube index
index2 = direction index (use directions array)
index3 = other cube index that is concave when in that direction

6.10.2.2 CubesWithInternalCavities `readonly bool [ ] Infohazard.HyperNav.Editor.MarchingCubes↩ CavityTables.CubesWithInternalCavities [static]`

This table maps a cube index (the same index used in MarchingCubesTables.TriTable to a boolean, which indicates whether that cube contains internal concavities.

If a cube contains internal cavities, it will always be concave no matter what cubes it is adjacent to.

The documentation for this class was generated from the following file:

- Editor/MarchingCubesCavityTables.cs

6.11 Infohazard.HyperNav.Editor.MarchingCubesTables Class Reference

Tables used in the Marching Cubes algorithm.

Static Public Attributes

- static readonly byte[ ][ ] TriTable

  This table maps one of 256 8-bit cube IDs to a list of triangles.
- static readonly byte[,] EdgeToVertexIndices

  This table maps one of 12 edge indices to the indices of the two vertices that it connects.
- static readonly Vector3Int[ ] Vertices

  This table maps one of 8 vertex indices to a local position in a cube.
- static readonly Vector3Int[ ] PositiveDirections

  Easy way to loop through positive directions in the order X, Y, Z.
- static readonly byte[ ] AcrossCenterMidpoints

  A list of the edge indices that cross the center of the cube.
- static readonly byte[ ][ ] VerticesOnSideAPerDirection

  This table maps the index of a direction in PositiveDirections to a list of vertices on the positive side of that direction.
- static readonly byte[ ][ ] VerticesOnSideBPerDirection

  This table maps the index of a direction in PositiveDirections to a list of vertices on the negative side of that direction.

6.11.1 Detailed Description

Tables used in the Marching Cubes algorithm.

6.11.2 Member Data Documentation

6.11.2.1 AcrossCenterMidpoints `readonly byte [] Infohazard.HyperNav.Editor.MarchingCubes↩ Tables.AcrossCenterMidpoints [static]`

Initial value:

```
= {
            6,
            7,
            4,
            5,

            2,
            3,
            0,
            1,

            10,
            11,
            8,
            9,
        }
```

A list of the edge indices that cross the center of the cube.

6.11.2.2 EdgeToVertexIndices `readonly byte [,] Infohazard.HyperNav.Editor.MarchingCubes↩ Tables.EdgeToVertexIndices [static]`

Initial value:

```
= {
            {0, 1},
            {1, 2},
            {2, 3},
            {3, 0},

            {4, 5},
            {5, 6},
            {6, 7},
            {7, 4},

            {0, 4},
            {1, 5},
            {2, 6},
            {3, 7},
        }
```

This table maps one of 12 edge indices to the indices of the two vertices that it connects.

6.11.2.3 PositiveDirections `readonly Vector3Int [] Infohazard.HyperNav.Editor.MarchingCubes↩ Tables.PositiveDirections [static]`

Initial value:

```
= {
            Vector3Int.right, Vector3Int.up, Vector3Int.forward,
        }
```

Easy way to loop through positive directions in the order X, Y, Z.

6.11.2.4 TriTable `readonly byte [ ][ ] Infohazard.HyperNav.Editor.MarchingCubesTables.TriTable [static]`

This table maps one of 256 8-bit cube IDs to a list of triangles.

Each bit of the cube ID represents one of 8 corners, which is either on or off. The order of these corners is that of the Vertices array. The output byte[] is a list of edge indices (each three edge indices makes a triangle). Each edge index is an index in EdgeToVertexIndices, giving two vertices to take the midpoint of in order to get one point of a triangle.

6.11.2.5 Vertices `readonly Vector3Int [ ] Infohazard.HyperNav.Editor.MarchingCubesTables.Vertices [static]`

Initial value:

```
= {
            new Vector3Int(0, 0, 0),
            new Vector3Int(1, 0, 0),
            new Vector3Int(1, 0, 1),
            new Vector3Int(0, 0, 1),

            new Vector3Int(0, 1, 0),
            new Vector3Int(1, 1, 0),
            new Vector3Int(1, 1, 1),
            new Vector3Int(0, 1, 1),
        }
```

This table maps one of 8 vertex indices to a local position in a cube.

6.11.2.6 VerticesOnSideAPerDirection `readonly byte [ ][ ] Infohazard.HyperNav.Editor.MarchingCubesTables.VerticesOnSideAPerDirection [static]`

Initial value:

```
= {
            new byte[] { 0, 3, 7, 4 },
            new byte[] { 0, 1, 2, 3 },
            new byte[] { 0, 1, 5, 4 },
        }
```

This table maps the index of a direction in PositiveDirections to a list of vertices on the positive side of that direction.

6.11.2.7 VerticesOnSideBPerDirection `readonly byte [ ][ ] Infohazard.HyperNav.Editor.MarchingCubesTables.VerticesOnSideBPerDirection [static]`

Initial value:

```
= {
            new byte[] { 1, 2, 6, 5 },
            new byte[] { 4, 5, 6, 7 },
            new byte[] { 2, 3, 7, 6 },
        }
```

This table maps the index of a direction in PositiveDirections to a list of vertices on the negative side of that direction.

The documentation for this class was generated from the following file:

- Editor/MarchingCubesTables.cs

6.12 Infohazard.HyperNav.Editor.MultiRegionMeshInfo Struct Reference

A struct used to store mesh data of an in-progress volume mesh.

Static Public Member Functions

- static MultiRegionMeshInfo CreateEmptyInfo ()

  Create a new empty MultiRegionMeshInfo with all data structures allocated.

Properties

- List< Vector3 > Vertices `[get, private set]`

  All vertices of the mesh.
- List< List< int > > VertexConnections `[get, private set]`

  For each vertex index, which other vertex indices it is connected to via edges.
- List< List< int > > VertexRegionMembership `[get, private set]`

  For each vertex index, which regions it is a part of.
- List< List< int > > RegionTriangleLists `[get, private set]`

  For each region, the indices of all the triangles in that region.
- Dictionary< Triangle, Dictionary< int, int > > TriangleIndicesPerRegion `[get, private set]`

  For each triangle, for each region, what index that triangle's vertices start in that region.

6.12.1 Detailed Description

A struct used to store mesh data of an in-progress volume mesh.

Contains cached info about connections to make mesh operations simpler.

6.12.2 Member Function Documentation

6.12.2.1 CreateEmptyInfo() `static MultiRegionMeshInfo Infohazard.HyperNav.Editor.MultiRegionMeshInfo.CreateEmptyInfo ( ) [static]`

Create a new empty MultiRegionMeshInfo with all data structures allocated.

Returns

The created MultiRegionMeshInfo.

6.12.3 Property Documentation

6.12.3.1 RegionTriangleLists `List<List<int> > Infohazard.HyperNav.Editor.MultiRegionMeshInfo.RegionTriangleLists [get], [private set]`

For each region, the indices of all the triangles in that region.

6.12.3.2 TriangleIndicesPerRegion `Dictionary<Triangle, Dictionary<int, int> > Infohazard.HyperNav.Editor.MultiRegionMeshInfo.TriangleIndicesPerRegion [get], [private set]`

For each triangle, for each region, what index that triangle's vertices start in that region.

6.12.3.3 VertexConnections `List<List<int> > Infohazard.HyperNav.Editor.MultiRegionMeshInfo.VertexConnections [get], [private set]`

For each vertex index, which other vertex indices it is connected to via edges.

6.12.3.4 VertexRegionMembership `List<List<int> > Infohazard.HyperNav.Editor.MultiRegionMeshInfo.VertexRegionMembership [get], [private set]`

For each vertex index, which regions it is a part of.

6.12.3.5 Vertices `List<Vector3> Infohazard.HyperNav.Editor.MultiRegionMeshInfo.Vertices [get], [private set]`

All vertices of the mesh.

The documentation for this struct was generated from the following file:

- Editor/NavVolumeBakingUtil.cs

6.13 Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData Struct Reference

Represents one obstacle (which may be an agent) in the avoidance system.

Public Attributes

- float3 Position

  Position of the obstacle.
- float3 InputVelocity

  Velocity of the obstacle (or desired velocity if it is an agent).
- float Radius

  Radius of the obstacle.
- float Padding

  If an agent, extra padding to give when avoiding obstacles.
- float Avoidance

  If an agent, its contribution weight to avoidance. If not an agent, zero.
- float Speed

  The maximum speed that this obstacle can move at.
- long TagMask

  The tag mask of this obstacle.
- long AvoidedTags

  If an agent, the tag masks this agent will avoid.
- bool Debug

  If an agent, whether to draw debug lines when calculating avoidance.

6.13.1 Detailed Description

Represents one obstacle (which may be an agent) in the avoidance system.

6.13.2 Member Data Documentation

6.13.2.1 Avoidance `float Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Avoidance`

If an agent, its contribution weight to avoidance. If not an agent, zero.

6.13.2.2 AvoidedTags `long Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.AvoidedTags`

If an agent, the tag masks this agent will avoid.

6.13.2.3 Debug `bool Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Debug`

If an agent, whether to draw debug lines when calculating avoidance.

6.13.2.4 InputVelocity `float3 Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.InputVelocity`

Velocity of the obstacle (or desired velocity if it is an agent).

6.13.2.5 Padding `float Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Padding`

If an agent, extra padding to give when avoiding obstacles.

6.13.2.6 Position `float3 Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Position`

Position of the obstacle.

6.13.2.7 Radius `float Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Radius`

Radius of the obstacle.

6.13.2.8 Speed `float Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.Speed`

The maximum speed that this obstacle can move at.

6.13.2.9 TagMask `long Infohazard.HyperNav.Jobs.NativeAvoidanceObstacleData.TagMask`

The tag mask of this obstacle.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeAvoidanceData.cs

6.14 Infohazard.HyperNav.Jobs.NativeBounds Struct Reference

A native-friendly of a bounding box.

Public Member Functions

- NativeBounds (float4 center, float4 extents)

  Initialize a new NativeBounds with the given data.

Public Attributes

- readonly float4 Center

  Center of the bounds.
- readonly float4 Extents

  Extents of the bounds (half of its size).

6.14.1 Detailed Description

A native-friendly of a bounding box.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 NativeBounds()

```
Infohazard.HyperNav.Jobs.NativeBounds.NativeBounds (
        float4 center,
        float4 extents )
```

Initialize a new NativeBounds with the given data.

Parameters

center	Center of the bounds.
extents	Extents of the bounds (half of its size).

6.14.3 Member Data Documentation

6.14.3.1 Center

```
readonly float4 Infohazard.HyperNav.Jobs.NativeBounds.Center
```

Center of the bounds.

6.14.3.2 Extents

```
readonly float4 Infohazard.HyperNav.Jobs.NativeBounds.Extents
```

Extents of the bounds (half of its size).

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.15 Infohazard.HyperNav.Jobs.NativeHeap< T > Struct Template Reference

An implementation of a Min Heap that can be used with jobs and Burst.

Public Member Functions

- NativeHeap (int initialCapacity, Allocator allocator)

  Create a new NativeHeap.
- void Dispose ()

  Free the memory used by the heap.
- void Clear ()

  Remove all items from the heap.
- void Add (T item, float priority)

  Add an item to the heap with the given priority.
- void Update (T item, float newPriority, bool replace=false, T replaceWith=default)

  Change the priority of an item in the heap, and optionally replace it with a new item.
- bool TryPeek (out T value)

  Get the element at the top of the heap without removing it.
- bool TryRemove (out T value)

  Remove the element at the top of the heap and return it.

Properties

- int Count `[get]`

  Current number of items in the heap.
- bool IsCreated `[get]`

  Whether the heap has been allocated.

6.15.1 Detailed Description

An implementation of a Min Heap that can be used with jobs and Burst.

The memory of the heap is allocated as a NativeList{T}.

Template Parameters

T	Element type of the heap.

Type Constraints

T* : *unmanaged

***T* : *IEquatable*<*T*>**

6.15.2 Constructor & Destructor Documentation

6.15.2.1 NativeHeap()

```
Infohazard.HyperNav.Jobs.NativeHeap< T >.NativeHeap (
        int initialCapacity,
        Allocator allocator )
```

Create a new NativeHeap.

Parameters

initialCapacity	The initial capacity to allocate.
allocator	The allocator to use.

6.15.3 Member Function Documentation

6.15.3.1 Add()

```
void Infohazard.HyperNav.Jobs.NativeHeap< T >.Add (
        T item,
        float priority )
```

Add an item to the heap with the given priority.

Parameters

item	Item to add.
priority	The item's priority.

6.15.3.2 Clear()

```
void Infohazard.HyperNav.Jobs.NativeHeap< T >.Clear ( )
```

Remove all items from the heap.

6.15.3.3 Dispose()

```
void Infohazard.HyperNav.Jobs.NativeHeap< T >.Dispose ( )
```

Free the memory used by the heap.

6.15.3.4 TryPeek()

```
bool Infohazard.HyperNav.Jobs.NativeHeap< T >.TryPeek (
        out T value )
```

Get the element at the top of the heap without removing it.

Parameters

value	The value at the top of the heap.

Returns

Whether the heap had an item to get.

6.15.3.5 TryRemove() `bool Infohazard.HyperNav.Jobs.NativeHeap< T >.TryRemove ( out T value )`

```
bool Infohazard.HyperNav.Jobs.NativeHeap< T >.TryRemove (
        out T value )
```

Remove the element at the top of the heap and return it.

Parameters

value	The value that was at the top of the heap.

Returns

Whether the heap had an item to remove.

6.15.3.6 Update()

```
void Infohazard.HyperNav.Jobs.NativeHeap< T >.Update (
        T item,
        float newPriority,
        bool replace = false,
        T replaceWith = default )
```

Change the priority of an item in the heap, and optionally replace it with a new item.

Parameters

item	The item to change.
newPriority	The new priority to use.
replace	Whether to replace the item as well.
replaceWith	What item to replace it with.

Exceptions

ArgumentException	If item is not contained in the heap.

6.15.4 Property Documentation

6.15.4.1 Count

```
int Infohazard.HyperNav.Jobs.NativeHeap< T >.Count  [get]
```

Current number of items in the heap.

6.15.4.2 IsCreated `bool Infohazard.HyperNav.Jobs.NativeHeap< T >.IsCreated [get]`

Whether the heap has been allocated.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeHeap.cs

6.16 Infohazard.HyperNav.Jobs.NativeMathUtility Class Reference

Provides math operations that are compatible with Burst.

Static Public Member Functions

- static float4 ProjectOnPlane (in float4 vector, in float3 normal)

  Project a vector onto a the plane defined by a normal.
- static bool GetNearestPointOnSegment (in float4 v1, in float4 v2, in float4 point, out float4 pointOnSegment)

  Find the point on a bounded line segment where it is nearest to a position, and return whether that point is in the segment's bounds.
- static bool GetNearestPointOnTriangle (in float4 v1, in float4 v2, in float4 v3, in float4 point, out float4 point←OnTriangle)

  Find the point on a triangle where it is nearest to a position, and return whether that point is in the triangle's bounds.
- static bool IsPointInsideBound (in float4 v1, in float4 v2, in float3 normal, in float4 point)

  Returns true if a given point is on the inner side (defined by a given normal) of a segment.
- static bool DoesSegmentIntersectTriangle (in float4 v1, in float4 v2, in float4 v3, in float4 s1, in float4 s2, out float t)

  Raycast a line segment against a triangle, and return whether they intersect.
- static unsafe bool NavRaycast (float4 start, float4 end, bool earlyReturn, in NativeNavVolumeData volume, out float t)

  Cast a ray against the blocking triangles of the volume, and return the nearest hit.
- static float4 GetPerpendicularVector (float4 vector)

  Returns an arbitrary vector that is perpendicular to the given vector.

6.16.1 Detailed Description

Provides math operations that are compatible with Burst.

Managed side versions are available in the Infohazard.Core library under MathUtility.

6.16.2 Member Function Documentation

6.16.2.1 DoesSegmentIntersectTriangle() `static bool Infohazard.HyperNav.Jobs.NativeMathUtility.←`

```
DoesSegmentIntersectTriangle (
            in float4 v1,
            in float4 v2,
            in float4 v3,
            in float4 s1,
            in float4 s2,
            out float t ) [static]
```

Raycast a line segment against a triangle, and return whether they intersect.

Parameters

v1	The first triangle point.
v2	The second triangle point.
v3	The third triangle point.
s1	The start of the segment.
s2	The end of the segment.
t	The point along the input segment where it intersects the triangle, or -1.

Returns

Whether the segment intersects the triangle.

6.16.2.2 GetNearestPointOnSegment()

```
static bool Infohazard.HyperNav.Jobs.NativeMathUtility.↩
GetNearestPointOnSegment (
            in float4 v1,
            in float4 v2,
            in float4 point,
            out float4 pointOnSegment )  [static]
```

Find the point on a bounded line segment where it is nearest to a position, and return whether that point is in the segment's bounds.

Does not return points on the ends of the segment. If the nearest point on the segment's line is outside the segment, will fail and not return a valid point.

Parameters

v1	The start of the segment.
v2	The end of the segment.
point	The point to search for.
pointOnSegment	The point on the segment closest to the input point.

Returns

Whether the nearest point is within the segment's bounds.

6.16.2.3 GetNearestPointOnTriangle()

```
static bool Infohazard.HyperNav.Jobs.NativeMathUtility.↩
GetNearestPointOnTriangle (
            in float4 v1,
            in float4 v2,
            in float4 v3,
            in float4 point,
            out float4 pointOnTriangle )  [static]
```

Find the point on a triangle where it is nearest to a position, and return whether that point is in the triangle's bounds.

Does not return points on the edge of the triangle. If the nearest point on the triangle's plane is outside the triangle, will fail and not return a valid point.

Parameters

v1	The first triangle point.
v2	The second triangle point.
v3	The third triangle point.
point	The point to search for.
pointOnTriangle	The point on the triangle closest to the input point.

Returns

Whether the nearest point is within the triangle's bounds.

6.16.2.4 GetPerpendicularVector()

```
static float4 Infohazard.HyperNav.Jobs.NativeMathUtility.Get←↩
PerpendicularVector (
            float4 vector )  [static]
```

Returns an arbitrary vector that is perpendicular to the given vector.

Parameters

vector	Input vector.

Returns

A perpendicular vector.

6.16.2.5 IsPointInsideBound()

```
static bool Infohazard.HyperNav.Jobs.NativeMathUtility.IsPoint←↩
InsideBound (
            in float4 v1,
            in float4 v2,
            in float3 normal,
            in float4 point )  [static]
```

Returns true if a given point is on the inner side (defined by a given normal) of a segment.

Parameters

v1	The start of the segment.
v2	The end of the segment.
normal	The normal, defining which side is inside.
point	The point to search for.

Returns

Whether the point is on the inner side.

6.16.2.6 NavRaycast()

```
static unsafe bool Infohazard.HyperNav.Jobs.NativeMathUtility.NavRaycast
(
            float4 start,
            float4 end,
            bool earlyReturn,
            in NativeNavVolumeData volume,
            out float t )  [static]
```

Cast a ray against the blocking triangles of the volume, and return the nearest hit.

Parameters

start	The position (in world space) to start the query at.
end	The position (in world space) to end the query at.
earlyReturn	If true, will return true as soon as any triangle is hit, not necessarily giving you the closest hit point.
volume	The volume in which to raycast.
t	If the query hits a triangle, the ratio between start and end at which the hit occurred.

Returns

Whether a triangle was hit.

6.16.2.7 ProjectOnPlane()

```
static float4 Infohazard.HyperNav.Jobs.NativeMathUtility.ProjectOn↩
Plane (
            in float4 vector,
            in float3 normal )  [static]
```

Project a vector onto a the plane defined by a normal.

Parameters

vector	The vector to project.
normal	The normal of the plane.

Returns

The projected vector.

The documentation for this class was generated from the following file:

- Runtime/Jobs/NativeMathUtility.cs

6.17 Infohazard.HyperNav.Jobs.NativeNavExternalLinkData Struct Reference

The native-friendly data representing a connection from one region to another region in another volume.

Public Member Functions

- NativeNavExternalLinkData (long toVolume, int toRegion, float4 fromPosition, float4 toPosition)

  Initialize a new NativeNavExternalLinkData with the given data.

Public Attributes

- readonly long ToVolume

  The ID of the connected volume.
- readonly int ToRegion

  The ID of the connected region.
- readonly float4 FromPosition

  The position at which the connection originates (world space).
- readonly float4 ToPosition

  The position at which the connection ends (world space).
- readonly float InternalCost

  The distance from FromPosition to ToPosition.

6.17.1 Detailed Description

The native-friendly data representing a connection from one region to another region in another volume.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 NativeNavExternalLinkData()

```
Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.Native←
NavExternalLinkData (
              long toVolume,
              int toRegion,
              float4 fromPosition,
              float4 toPosition )
```

Initialize a new NativeNavExternalLinkData with the given data.

Parameters

toVolume	The ID of the connected volume.
toRegion	The ID of the connected region.
fromPosition	The position at which the connection originates.
toPosition	The position at which the connection ends.

6.17.3 Member Data Documentation

6.17.3.1 FromPosition `readonly float4 Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.FromPosition`

The position at which the connection originates (world space).

6.17.3.2 InternalCost `readonly float Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.InternalCost`

The distance from FromPosition to ToPosition.

6.17.3.3 ToPosition `readonly float4 Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.ToPosition`

The position at which the connection ends (world space).

6.17.3.4 ToRegion `readonly int Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.ToRegion`

The ID of the connected region.

6.17.3.5 ToVolume `readonly long Infohazard.HyperNav.Jobs.NativeNavExternalLinkData.ToVolume`

The ID of the connected volume.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.18 Infohazard.HyperNav.Jobs.NativeNavHit Struct Reference

A native-friendly representation of a navigation query result.

Public Member Functions

- NativeNavHit (long volume, int region, bool isOnEdge, float4 position)

  Initialize a new NativeNavHit with the given data.

Public Attributes

- readonly long Volume

  ID of the volume that was hit.
- readonly int Region

  ID of the region that was hit.
- readonly bool IsOnEdge

  Whether the result point was on the edge of the region.
- readonly float4 Position

  Position of the hit.

6.18.1 Detailed Description

A native-friendly representation of a navigation query result.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 NativeNavHit()

```
Infohazard.HyperNav.Jobs.NativeNavHit.NativeNavHit (
        long volume,
        int region,
        bool isOnEdge,
        float4 position )
```

Initialize a new NativeNavHit with the given data.

Parameters

volume	ID of the volume that was hit.
region	ID of the region that was hit.
isOnEdge	Whether the result point was on the edge of the region.
position	Position of the hit.

6.18.3 Member Data Documentation

6.18.3.1 IsOnEdge

```
readonly bool Infohazard.HyperNav.Jobs.NativeNavHit.IsOnEdge
```

Whether the result point was on the edge of the region.

6.18.3.2 Position

```
readonly float4 Infohazard.HyperNav.Jobs.NativeNavHit.Position
```

Position of the hit.

6.18.3.3 Region `readonly int Infohazard.HyperNav.Jobs.NativeNavHit.Region`

ID of the region that was hit.

6.18.3.4 Volume `readonly long Infohazard.HyperNav.Jobs.NativeNavHit.Volume`

ID of the volume that was hit.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.19 Infohazard.HyperNav.Jobs.NativeNavInternalLinkData Struct Reference

The native-friendly data representing a connection from one region to another region in the same volume.

Public Member Functions

- NativeNavInternalLinkData (int toRegion, int verticesStart, int verticesCount, int edgesStart, int edgesCount, int trianglesStart, int trianglesCount)

  Initialize a new NativeNavInternalLinkData with the given data.

Public Attributes

- readonly int ToRegion

  The ID of the connected region.
- readonly int VerticesStart

  The index of the link's first vertex in the volume's NativeNavVolumeData.LinkVertices list.
- readonly int VerticesCount

  The number of link vertices.
- readonly int EdgesStart

  The index of the link's first edge in the volume's NativeNavVolumeData.LinkEdges list.
- readonly int EdgesCount

  The number of link edges.
- readonly int TrianglesStart

  The index of the link's first triangle in the volume's NativeNavVolumeData.LinkTriangles list.
- readonly int TrianglesCount

  The number of link triangles.

6.19.1 Detailed Description

The native-friendly data representing a connection from one region to another region in the same volume.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 NativeNavInternalLinkData() `Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.NativeNavInternalLinkData`

```
Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.Native↩
NavInternalLinkData (
            int toRegion,
            int verticesStart,
            int verticesCount,
            int edgesStart,
            int edgesCount,
            int trianglesStart,
            int trianglesCount )
```

Initialize a new NativeNavInternalLinkData with the given data.

Parameters

toRegion	The ID of the connected region.
verticesStart	The index of the link's first vertex.
verticesCount	The number of link vertices.
edgesStart	The index of the link's first edge.
edgesCount	The number of link edges.
trianglesStart	The index of the link's first triangle.
trianglesCount	The number of link triangles.

6.19.3 Member Data Documentation

6.19.3.1 EdgesCount `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.Edges↩ Count`

The number of link edges.

6.19.3.2 EdgesStart `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.EdgesStart`

The index of the link's first edge in the volume's NativeNavVolumeData.LinkEdges list.

6.19.3.3 ToRegion `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.ToRegion`

The ID of the connected region.

6.19.3.4 TrianglesCount `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.↩ TrianglesCount`

The number of link triangles.

6.19.3.5 TrianglesStart `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.TrianglesStart`

The index of the link's first triangle in the volume's NativeNavVolumeData.LinkTriangles list.

6.19.3.6 VerticesCount `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.VerticesCount`

The number of link vertices.

6.19.3.7 VerticesStart `readonly int Infohazard.HyperNav.Jobs.NativeNavInternalLinkData.VerticesStart`

The index of the link's first vertex in the volume's NativeNavVolumeData.LinkVertices list.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.20 Infohazard.HyperNav.Jobs.NativeNavRegionData Struct Reference

The native-friendly data representing a single region in a NavVolume.

Public Member Functions

- NativeNavRegionData (int id, NativeBounds bounds, int internalLinkStart, int internalLinkCount, int externalLinkStart, int externalLinkCount)

  Initialize a new NativeNavRegionData with the given data.

Public Attributes

- readonly int ID

  The ID of the region.
- readonly NativeBounds Bounds

  The bounds of the region in local space of the volume.
- readonly int InternalLinkStart

  The index of the region's first internal link in the volume's NativeNavVolumeData.InternalLinks list.
- readonly int InternalLinkCount

  The number of internal links.
- readonly int ExternalLinkStart

  The index of the region's first external link in the volume's NativeNavVolumeData.ExternalLinks list.
- readonly int ExternalLinkCount

  The number of external links.

6.20.1 Detailed Description

The native-friendly data representing a single region in a NavVolume.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 NativeNavRegionData()

```
Infohazard.HyperNav.Jobs.NativeNavRegionData.NativeNavRegion↩
Data (
            int id,
            NativeBounds bounds,
            int internalLinkStart,
            int internalLinkCount,
            int externalLinkStart,
            int externalLinkCount )
```

Initialize a new NativeNavRegionData with the given data.

Parameters

id	The ID of the region.
bounds	The bounds of the region in local space of the volume.
internalLinkStart	The index of the region's first internal link.
internalLinkCount	The number of internal links.
externalLinkStart	The index of the region's first external link.
externalLinkCount	The number of external links.

6.20.3 Member Data Documentation

6.20.3.1 Bounds

```
readonly NativeBounds Infohazard.HyperNav.Jobs.NativeNavRegionData.Bounds
```

The bounds of the region in local space of the volume.

6.20.3.2 ExternalLinkCount

```
readonly int Infohazard.HyperNav.Jobs.NativeNavRegionData.External↩
LinkCount
```

The number of external links.

6.20.3.3 ExternalLinkStart `readonly int Infohazard.HyperNav.Jobs.NativeNavRegionData.ExternalLinkStart`

The index of the region's first external link in the volume's NativeNavVolumeData.ExternalLinks list.

6.20.3.4 ID `readonly int Infohazard.HyperNav.Jobs.NativeNavRegionData.ID`

The ID of the region.

6.20.3.5 InternalLinkCount `readonly int Infohazard.HyperNav.Jobs.NativeNavRegionData.InternalLinkCount`

The number of internal links.

6.20.3.6 InternalLinkStart `readonly int Infohazard.HyperNav.Jobs.NativeNavRegionData.InternalLinkStart`

The index of the region's first internal link in the volume's NativeNavVolumeData.InternalLinks list.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.21 Infohazard.HyperNav.Jobs.NativeNavVolumeData Struct Reference

The baked data of a NavVolume, converted to a form compatible with Burst.

Public Member Functions

- NativeNavVolumeData (long id, float4x4 transform, float4x4 inverseTransform, NativeBounds bounds, UnsafeArrayPtr< float4 > vertices, UnsafeArrayPtr< NativeNavRegionData > regions, UnsafeArrayPtr< int > blockingTriangleIndices, UnsafeArrayPtr< NativeNavInternalLinkData > internalLinks, UnsafeArrayPtr< NativeNavExternalLinkData > externalLinks, UnsafeArrayPtr< int > linkVertices, UnsafeArrayPtr< int2 > linkEdges, UnsafeArrayPtr< int3 > linkTriangles)

  Initialize a new NativeNavVolumeData with the given data.

Public Attributes

- readonly long ID

  ID of the volume.
- readonly float4x4 Transform

  Transform matrix of the volume.
- readonly float4x4 InverseTransform

  Inverse transform matrix of the volume.
- readonly NativeBounds Bounds

  Bounds of the volume in local space.
- readonly UnsafeArrayPtr< float4 > Vertices

  The vertex positions of all of the volume's regions, in local space.
- readonly UnsafeArrayPtr< NativeNavRegionData > Regions

  The regions of the volume.
- readonly UnsafeArrayPtr< int > BlockingTriangleIndices

  The vertex indices of triangles that define impassible space in the volume.
- readonly UnsafeArrayPtr< NativeNavInternalLinkData > InternalLinks

  The internal links of all of the volume's regions.
- readonly UnsafeArrayPtr< NativeNavExternalLinkData > ExternalLinks

  The external links of all of the volume's regions.
- readonly UnsafeArrayPtr< int > LinkVertices

  The shared vertices of all of the volume's internal links.
- readonly UnsafeArrayPtr< int2 > LinkEdges

  The shared edges of all of the volume's internal links.
- readonly UnsafeArrayPtr< int3 > LinkTriangles

  The shared triangles of all of the volume's internal links.

6.21.1 Detailed Description

The baked data of a NavVolume, converted to a form compatible with Burst.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 NativeNavVolumeData() `Infohazard.HyperNav.Jobs.NativeNavVolumeData.NativeNavVolume←Data (`

```
            long id,
            float4x4 transform,
            float4x4 inverseTransform,
            NativeBounds bounds,
            UnsafeArrayPtr< float4 > vertices,
            UnsafeArrayPtr< NativeNavRegionData > regions,
            UnsafeArrayPtr< int > blockingTriangleIndices,
            UnsafeArrayPtr< NativeNavInternalLinkData > internalLinks,
            UnsafeArrayPtr< NativeNavExternalLinkData > externalLinks,
            UnsafeArrayPtr< int > linkVertices,
            UnsafeArrayPtr< int2 > linkEdges,
            UnsafeArrayPtr< int3 > linkTriangles )
```

Initialize a new NativeNavVolumeData with the given data.

Parameters

id	ID of the volume.
transform	Transform matrix of the volume.
inverseTransform	Inverse transform matrix of the volume.
bounds	Bounds of the volume in local space.
vertices	The vertex positions of all of the volume's regions.
regions	The regions of the volume.
blockingTriangleIndices	The indices of triangles that define impassible space in the volume.
internalLinks	The internal links of all of the volume's regions.
externalLinks	The external links of all of the volume's regions.
linkVertices	The shared vertices of all of the volume's internal links.
linkEdges	The shared edges of all of the volume's internal links.
linkTriangles	The shared triangles of all of the volume's internal links.

6.21.3 Member Data Documentation

6.21.3.1 BlockingTriangleIndices `readonly UnsafeArrayPtr<int> Infohazard.HyperNav.Jobs.NativeNavVolumeData.BlockingTriangleIndices`

The vertex indices of triangles that define impassible space in the volume.

6.21.3.2 Bounds `readonly NativeBounds Infohazard.HyperNav.Jobs.NativeNavVolumeData.Bounds`

Bounds of the volume in local space.

6.21.3.3 ExternalLinks `readonly UnsafeArrayPtr<NativeNavExternalLinkData> Infohazard.HyperNav.Jobs.NativeNavVolumeData.ExternalLinks`

The external links of all of the volume's regions.

6.21.3.4 ID `readonly long Infohazard.HyperNav.Jobs.NativeNavVolumeData.ID`

ID of the volume.

6.21.3.5 InternalLinks `readonly UnsafeArrayPtr<NativeNavInternalLinkData> Infohazard.Hyper↩ Nav.Jobs.NativeNavVolumeData.InternalLinks`

The internal links of all of the volume's regions.

6.21.3.6 InverseTransform `readonly float4x4 Infohazard.HyperNav.Jobs.NativeNavVolumeData.↩ InverseTransform`

Inverse transform matrix of the volume.

6.21.3.7 LinkEdges `readonly UnsafeArrayPtr<int2> Infohazard.HyperNav.Jobs.NativeNavVolume↩ Data.LinkEdges`

The shared edges of all of the volume's internal links.

6.21.3.8 LinkTriangles `readonly UnsafeArrayPtr<int3> Infohazard.HyperNav.Jobs.NativeNav↩ VolumeData.LinkTriangles`

The shared triangles of all of the volume's internal links.

6.21.3.9 LinkVertices `readonly UnsafeArrayPtr<int> Infohazard.HyperNav.Jobs.NativeNavVolume↩ Data.LinkVertices`

The shared vertices of all of the volume's internal links.

6.21.3.10 Regions `readonly UnsafeArrayPtr<NativeNavRegionData> Infohazard.HyperNav.Jobs.↩ NativeNavVolumeData.Regions`

The regions of the volume.

6.21.3.11 Transform `readonly float4x4 Infohazard.HyperNav.Jobs.NativeNavVolumeData.Transform`

Transform matrix of the volume.

6.21.3.12 Vertices `readonly UnsafeArrayPtr<float4> Infohazard.HyperNav.Jobs.NativeNavVolumeData.Vertices`

The vertex positions of all of the volume's regions, in local space.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.22 Infohazard.HyperNav.Jobs.NativeNavWaypoint Struct Reference

A structure used by the navigation job to return the waypoints of a path.

Public Member Functions

- NativeNavWaypoint (float4 position, NavWaypointType type, long volumeID)

  Initialize a new NativeNavWaypoint with the given data.

Public Attributes

- readonly float4 Position

  Position of the waypoint in world space.
- readonly NavWaypointType Type

  Type of the waypoint in relation to the containing volume.
- readonly long VolumeID

  Identifier of the NavVolume that contains this waypoint, or -1.

6.22.1 Detailed Description

A structure used by the navigation job to return the waypoints of a path.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 NativeNavWaypoint()

```
Infohazard.HyperNav.Jobs.NativeNavWaypoint.NativeNavWaypoint (
        float4 position,
        NavWaypointType type,
        long volumeID )
```

Initialize a new NativeNavWaypoint with the given data.

Parameters

position	Position of the waypoint in world space.
type	Type of the waypoint in relation to the containing volume.
volumeID	Identifier of the NavVolume that contains this waypoint, or -1.

6.22.3 Member Data Documentation

6.22.3.1 Position `readonly float4 Infohazard.HyperNav.Jobs.NativeNavWaypoint.Position`

Position of the waypoint in world space.

6.22.3.2 Type `readonly NavWaypointType Infohazard.HyperNav.Jobs.NativeNavWaypoint.Type`

Type of the waypoint in relation to the containing volume.

6.22.3.3 VolumeID `readonly long Infohazard.HyperNav.Jobs.NativeNavWaypoint.VolumeID`

Identifier of the NavVolume that contains this waypoint, or -1.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeNavData.cs

6.23 Infohazard.HyperNav.Jobs.NativePlane Struct Reference

A plane constructed using native math types.

Public Member Functions

- NativePlane (float4 normal, float4 point)

  Construct a new NativePlane, calculating the distance based on any point in the plane.

Public Attributes

- float4 Normal

  Normal of the plane, which should be normalized.
- float Distance

  Distance from the origin to the nearest point on the plane.

Properties

- float4 Center `[get]`

  Nearest point to the origin on the plane.

6.23.1 Detailed Description

A plane constructed using native math types.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 NativePlane()

```
Infohazard.HyperNav.Jobs.NativePlane.NativePlane (
        float4 normal,
        float4 point )
```

Construct a new NativePlane, calculating the distance based on any point in the plane.

Parameters

normal	Normal of the plane, which should be normalized.
point	Any point on the plane.

6.23.3 Member Data Documentation

6.23.3.1 Distance `float Infohazard.HyperNav.Jobs.NativePlane.Distance`

Distance from the origin to the nearest point on the plane.

6.23.3.2 Normal `float4 Infohazard.HyperNav.Jobs.NativePlane.Normal`

Normal of the plane, which should be normalized.

6.23.4 Property Documentation

6.23.4.1 Center `float4 Infohazard.HyperNav.Jobs.NativePlane.Center [get]`

Nearest point to the origin on the plane.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeAvoidanceData.cs

6.24 Infohazard.HyperNav.Jobs.NativeRay Struct Reference

A ray constructed using native math types.

Public Attributes

- float4 Origin

  Origin of the ray.
- float4 Direction

  Direction of the ray, which should be normalized.

6.24.1 Detailed Description

A ray constructed using native math types.

6.24.2 Member Data Documentation

6.24.2.1 Direction `float4 Infohazard.HyperNav.Jobs.NativeRay.Direction`

Direction of the ray, which should be normalized.

6.24.2.2 Origin `float4 Infohazard.HyperNav.Jobs.NativeRay.Origin`

Origin of the ray.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NativeAvoidanceData.cs

6.25 Infohazard.HyperNav.Jobs.NativeRaycastElement Struct Reference

A single raycast in a NavMultiRaycastJob.

Public Attributes

- long VolumeID

  Volume to raycast in.
- float OutDistance

  Where the hit distance (or -1 if no hit) of the raycast is written.
- float4 Start

  Start point of the segment.
- float4 End

  End point of the segment.

6.25.1 Detailed Description

A single raycast in a NavMultiRaycastJob.

6.25.2 Member Data Documentation

6.25.2.1 End `float4 Infohazard.HyperNav.Jobs.NativeRaycastElement.End`

End point of the segment.

6.25.2.2 OutDistance `float Infohazard.HyperNav.Jobs.NativeRaycastElement.OutDistance`

Where the hit distance (or -1 if no hit) of the raycast is written.

6.25.2.3 Start `float4 Infohazard.HyperNav.Jobs.NativeRaycastElement.Start`

Start point of the segment.

6.25.2.4 VolumeID `long Infohazard.HyperNav.Jobs.NativeRaycastElement.VolumeID`

Volume to raycast in.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavRaycastJob.cs

6.26 Infohazard.HyperNav.NavAgent Class Reference

A script that can be used to calculate paths by any entity that needs to use HyperNav for navigation.

Classes

- class PropNames

  This is used to refer to the names of private fields in this class from a custom Editor.

Public Member Functions

- virtual void Stop (bool abortPaths)

  Stop following the current path, and optionally cancel all path requests.
- virtual void UpdatePath ()

  Request a new path from the current position to the desired destination.

Protected Member Functions

- virtual void Awake ()

  Sets the AvoidanceAgent.Infohazard.HyperNav.AvoidanceAgent.InputVelocityFunc.
- virtual void OnEnable ()

  Resets MeasuredVelocity and sets Arrived to true.
- virtual void OnDisable ()

  Stops all pathfinding and cancels path requests.
- virtual void Update ()

  Updates measured velocity and current index in path.
- virtual void OnDrawGizmos ()

  Draws the current path as a sequence of debug lines if DebugPath is true.
- virtual Vector3 CalculateDesiredNavigationVelocity ()

  Calculate the velocity the agent wants to move in, in the range [0, 1].
- virtual void UpdateMeasuredVelocity ()

  Update the value of MeasuredVelocity, which is used to determine StoppingDistance.
- virtual void UpdatePathIndex ()

  Update the current path index, which is used to determine NextWaypoint.
- virtual void OnPathReady (long id, NavPath path)

  Callback that is received when a pathfinding request completes, which should start moving along that path.

Properties

- float Acceptance `[get, set]`

  How close the agent must get to a destination before it is considered to have arrived.
- float AccelerationEstimate `[get, set]`

  (Serialized) This should be set to the maximum acceleration of your agent.
- float SampleRadius `[get, set]`

  The radius to search when finding the nearest NavVolume.
- float DesiredSpeedRatio `[get, set]`

  The desired fraction of the maximum speed to travel at.
- bool DebugPath `[get, set]`

  Whether to draw a debug line in the scene view showing the agent's current path.
- bool KeepPathWhileCalculating `[get, set]`

  Whether to keep following the current path while waiting for a new path to finish calculating.
- AvoidanceAgent AvoidanceAgent `[get, set]`

  AvoidanceAgent that this agent uses for avoidance (can be null).
- bool ControlAvoidanceIsActive `[get, set]`

  If true, the Infohazard.HyperNav.AvoidanceAgent.IsActive state of the AvoidanceAgent is set based on whether there is a current valid path.
- bool IsPathPending `[get]`

  Whether a path is currently in the process of being calculated for this agent.
- virtual float StoppingDistance `[get]`

  The distance that it will take the agent to come to a stop from its current velocity, determined using the AccelerationEstimate.
- Vector3 NextWaypoint `[get]`

  The current path waypoint that the agent is trying to move towards.
- bool Arrived `[get, private set]`

  True if the agent has no active or pending path.
- Vector3 Destination `[get, set]`

  Get or set the agent's destination (the position it is trying to get to).

- Vector3 MeasuredVelocity `[get, protected set]`

  Velocity of the agent measured as delta position / delta time over the last frame, which is used to determine stopping distance.
- NavPath CurrentPath `[get, set]`

  The current path that the agent is following.
- float AvoidanceMaxSpeed `[get, set]`

  Maximum speed possible by this agent when avoiding obstacles.

Events

- Action PathReady

  Invoked when the agent finds a path to the destination.
- Action PathFailed

  Invoked when the agent fails to find a path to the destination.

Private Attributes

- float _acceptance = 1

  (Serialized) How close the agent must get to a destination before it is considered to have arrived.
- float _accelerationEstimate = 0

  (Serialized) This should be set to the maximum acceleration of your agent (can be set dynamically as well).
- float _sampleRadius = 2

  (Serialized) The radius to search when finding the nearest NavVolume.
- float _desiredSpeedRatio = 1

  (Serialized) The desired fraction of the maximum speed to travel at.
- bool _debugPath = true

  (Serialized) Whether to draw a debug line in the scene view showing the agent's current path.
- bool _keepPathWhileCalculating = true

  (Serialized) Whether to keep following the current path while waiting for a new path to finish calculating.
- AvoidanceAgent _avoidanceAgent

  (Serialized) AvoidanceAgent that this agent uses for avoidance (can be null).
- bool _controlAvoidanceIsActive = true

  (Serialized) If true, the Infohazard.HyperNav.AvoidanceAgent.IsActive state of the AvoidanceAgent is set based on whether there is a current valid path.

6.26.1 Detailed Description

A script that can be used to calculate paths by any entity that needs to use HyperNav for navigation.

While a NavAgent is not necessary to use HyperNav, it makes pathfinding easier. The NavAgent does not impose any restrictions on how movement occurs, nor does it actually perform any movement. It simply provides a desired movement velocity, which other scripts on the object are responsible for using however they need.

The agent can have one active path (the path it is currently following), but can have multiple pending paths (paths in the process of being calculated by a NavPathfinder).

If you desire smoother movement then what the NavAgent provides, see SplineNavAgent.

6.26.2 Member Function Documentation

6.26.2.1 Awake() `virtual void Infohazard.HyperNav.NavAgent.Awake ( ) [protected], [virtual]`

Sets the AvoidanceAgent.Infohazard.HyperNav.AvoidanceAgent.InputVelocityFunc.

6.26.2.2 CalculateDesiredNavigationVelocity() `virtual Vector3 Infohazard.HyperNav.NavAgent.↩ CalculateDesiredNavigationVelocity ( ) [protected], [virtual]`

Calculate the velocity the agent wants to move in, in the range [0, 1].

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.3 OnDisable() `virtual void Infohazard.HyperNav.NavAgent.OnDisable ( ) [protected], [virtual]`

Stops all pathfinding and cancels path requests.

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.4 OnDrawGizmos() `virtual void Infohazard.HyperNav.NavAgent.OnDrawGizmos ( ) [protected], [virtual]`

Draws the current path as a sequence of debug lines if DebugPath is true.

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.5 OnEnable() `virtual void Infohazard.HyperNav.NavAgent.OnEnable ( ) [protected], [virtual]`

Resets MeasuredVelocity and sets Arrived to true.

6.26.2.6 OnPathReady()

```
virtual void Infohazard.HyperNav.NavAgent.OnPathReady (
            long id,
            NavPath path ) [protected], [virtual]
```

Callback that is received when a pathfinding request completes, which should start moving along that path.

Parameters

id	The id of the path request.
path	The completed path, which is null if no path was found.

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.7 Stop() `virtual void Infohazard.HyperNav.NavAgent.Stop ( bool abortPaths ) [virtual]`

Stop following the current path, and optionally cancel all path requests.

Parameters

abortPaths	Whether to cancel pending path requests.

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.8 Update() `virtual void Infohazard.HyperNav.NavAgent.Update ( ) [protected], [virtual]`

Updates measured velocity and current index in path.

Reimplemented in Infohazard.HyperNav.SplineNavAgent.

6.26.2.9 UpdateMeasuredVelocity() `virtual void Infohazard.HyperNav.NavAgent.UpdateMeasured↩ Velocity ( ) [protected], [virtual]`

Update the value of MeasuredVelocity, which is used to determine StoppingDistance.

6.26.2.10 UpdatePath() `virtual void Infohazard.HyperNav.NavAgent.UpdatePath ( ) [virtual]`

Request a new path from the current position to the desired destination.

It is usually not necessary to call this yourself, as it is called when setting Destination. However, if the agent gets stuck or pushed off course, you may wish to use this to get a new path.

6.26.2.11 UpdatePathIndex() `virtual void Infohazard.HyperNav.NavAgent.UpdatePathIndex ( ) [protected], [virtual]`

Update the current path index, which is used to determine NextWaypoint.

6.26.3 Member Data Documentation

6.26.3.1 _accelerationEstimate `float Infohazard.HyperNav.NavAgent._accelerationEstimate = 0 [private]`

(Serialized) This should be set to the maximum acceleration of your agent (can be set dynamically as well).

6.26.3.2 _acceptance `float Infohazard.HyperNav.NavAgent._acceptance = 1 [private]`

(Serialized) How close the agent must get to a destination before it is considered to have arrived.

6.26.3.3 _avoidanceAgent `AvoidanceAgent Infohazard.HyperNav.NavAgent._avoidanceAgent [private]`

(Serialized) AvoidanceAgent that this agent uses for avoidance (can be null).

6.26.3.4 _controlAvoidanceIsActive `bool Infohazard.HyperNav.NavAgent._controlAvoidanceIsActive = true [private]`

(Serialized) If true, the Infohazard.HyperNav.AvoidanceAgent.IsActive state of the AvoidanceAgent is set based on whether there is a current valid path.

6.26.3.5 _debugPath `bool Infohazard.HyperNav.NavAgent._debugPath = true [private]`

(Serialized) Whether to draw a debug line in the scene view showing the agent's current path.

6.26.3.6 _desiredSpeedRatio `float Infohazard.HyperNav.NavAgent._desiredSpeedRatio = 1 [private]`

(Serialized) The desired fraction of the maximum speed to travel at.

6.26.3.7 _keepPathWhileCalculating `bool Infohazard.HyperNav.NavAgent._keepPathWhileCalculating = true [private]`

(Serialized) Whether to keep following the current path while waiting for a new path to finish calculating.

6.26.3.8 _sampleRadius `float Infohazard.HyperNav.NavAgent._sampleRadius = 2 [private]`

(Serialized) The radius to search when finding the nearest NavVolume.

6.26.4 Property Documentation

6.26.4.1 AccelerationEstimate `float Infohazard.HyperNav.NavAgent.AccelerationEstimate [get], [set]`

(Serialized) This should be set to the maximum acceleration of your agent.

This is used to determine when the agent needs to start slowing down when approaching its destination.

6.26.4.2 Acceptance `float Infohazard.HyperNav.NavAgent.Acceptance [get], [set]`

How close the agent must get to a destination before it is considered to have arrived.

Note that setting acceptance too low may prevent the agent from ever stopping, but setting it to high can make the agent stop too far from the destination.

6.26.4.3 Arrived `bool Infohazard.HyperNav.NavAgent.Arrived [get], [private set]`

True if the agent has no active or pending path.

6.26.4.4 AvoidanceAgent `AvoidanceAgent Infohazard.HyperNav.NavAgent.AvoidanceAgent [get], [set]`

AvoidanceAgent that this agent uses for avoidance (can be null).

6.26.4.5 AvoidanceMaxSpeed `float Infohazard.HyperNav.NavAgent.AvoidanceMaxSpeed [get], [set]`

Maximum speed possible by this agent when avoiding obstacles.

6.26.4.6 ControlAvoidanceIsActive `bool Infohazard.HyperNav.NavAgent.ControlAvoidanceIsActive [get], [set]`

If true, the Infohazard.HyperNav.AvoidanceAgent.IsActive state of the AvoidanceAgent is set based on whether there is a current valid path.

6.26.4.7 CurrentPath `NavPath Infohazard.HyperNav.NavAgent.CurrentPath [get], [set]`

The current path that the agent is following.

6.26.4.8 DebugPath `bool Infohazard.HyperNav.NavAgent.DebugPath [get], [set]`

Whether to draw a debug line in the scene view showing the agent's current path.

6.26.4.9 DesiredSpeedRatio `float Infohazard.HyperNav.NavAgent.DesiredSpeedRatio [get], [set]`

The desired fraction of the maximum speed to travel at.

6.26.4.10 Destination `Vector3 Infohazard.HyperNav.NavAgent.Destination [get], [set]`

Get or set the agent's destination (the position it is trying to get to).

If set within the _acceptance radius of the current position, will abort all movement.

6.26.4.11 IsPathPending `bool Infohazard.HyperNav.NavAgent.IsPathPending [get]`

Whether a path is currently in the process of being calculated for this agent.

6.26.4.12 KeepPathWhileCalculating `bool Infohazard.HyperNav.NavAgent.KeepPathWhileCalculating [get], [set]`

Whether to keep following the current path while waiting for a new path to finish calculating.

If true, there can be two pending paths at the same time - the most and least recently requested ones. This ensures that even when the agent is receiving pathfinding requests faster than they can be calculated, they will still finish and the agent will not be deadlocked and unable to ever complete a path.

6.26.4.13 MeasuredVelocity `Vector3 Infohazard.HyperNav.NavAgent.MeasuredVelocity [get], [protected set]`

Velocity of the agent measured as delta position / delta time over the last frame, which is used to determine stopping distance.

This value is calculated in UpdateMeasuredVelocity. You can override that method to implement your own logic for calculating velocity.

6.26.4.14 NextWaypoint `Vector3 Infohazard.HyperNav.NavAgent.NextWaypoint [get]`

The current path waypoint that the agent is trying to move towards.

If there is no active path, will return the agent's current position.

6.26.4.15 SampleRadius `float Infohazard.HyperNav.NavAgent.SampleRadius [get], [set]`

The radius to search when finding the nearest NavVolume.

6.26.4.16 StoppingDistance `virtual float Infohazard.HyperNav.NavAgent.StoppingDistance [get]`

The distance that it will take the agent to come to a stop from its current velocity, determined using the AccelerationEstimate.

6.26.5 Event Documentation

6.26.5.1 PathFailed `Action Infohazard.HyperNav.NavAgent.PathFailed`

Invoked when the agent fails to find a path to the destination.

6.26.5.2 PathReady `Action Infohazard.HyperNav.NavAgent.PathReady`

Invoked when the agent finds a path to the destination.

The documentation for this class was generated from the following file:

- Runtime/NavAgent.cs

6.27 Infohazard.HyperNav.NavDataInternalPointers Struct Reference

References to the NativeArrays allocated for a NativeNavVolumeData.

Public Member Functions

- void Dispose ()

  Dispose and nullify all of the native array references.

6.27.1 Detailed Description

References to the NativeArrays allocated for a NativeNavVolumeData.

In the NativeNavVolumeData itself, these arrays are kept as pointers, which cannot be used to deallocate the arrays under Unity's safe memory system. In order to play nicely with that system, the original references must be kept and disposed.

6.27.2 Member Function Documentation

6.27.2.1 Dispose() `void Infohazard.HyperNav.NavDataInternalPointers.Dispose ( )`

Dispose and nullify all of the native array references.

The documentation for this struct was generated from the following file:

- Runtime/NavVolumeData.cs

6.28 Infohazard.HyperNav.Editor.NavEditorUtility Class Reference

Utility functions used internally, but you can use them too, I mean I'm not your boss.

Static Public Member Functions

- static void ExportPreviewMesh (Mesh mesh)

  Export a mesh as an OBJ file.
- static void CreateVolume ()

  Menu item to create a new NavVolume.
- static void CreatePathfinder ()

  Menu item to create a new NavPathfinder.
- static void CreateAvoidanceManager ()

  Menu item to create a new AvoidanceManager.

6.28.1 Detailed Description

Utility functions used internally, but you can use them too, I mean I'm not your boss.

6.28.2 Member Function Documentation

6.28.2.1 CreateAvoidanceManager() `static void Infohazard.HyperNav.Editor.NavEditorUtility.CreateAvoidanceManager ( ) [static]`

Menu item to create a new AvoidanceManager.

6.28.2.2 CreatePathfinder() `static void Infohazard.HyperNav.Editor.NavEditorUtility.CreatePathfinder ( ) [static]`

Menu item to create a new NavPathfinder.

6.28.2.3 CreateVolume() `static void Infohazard.HyperNav.Editor.NavEditorUtility.CreateVolume ( ) [static]`

Menu item to create a new NavVolume.

6.28.2.4 ExportPreviewMesh() `static void Infohazard.HyperNav.Editor.NavEditorUtility.ExportPreviewMesh ( Mesh mesh ) [static]`

Export a mesh as an OBJ file.

This is the most basic export possible, and should not be used for actual art. It is only used to more closely inspect a preview mesh in a modeling application. The material names will be included in the OBJ, but the MTL file is not created. Normals are also not included.

Parameters

mesh	The mesh to export.

The documentation for this class was generated from the following file:

- Editor/NavEditorUtility.cs

6.29 Infohazard.HyperNav.NavExternalLinkData Class Reference

A connection from one region to another region in another volume.

Public Member Functions

- void ToInternalData (Transform volumeTransform, bool volumeLocalSpace, out NativeNavExternalLinkData data)

  Convert to a native representation, transforming points to world space if necessary.

Static Public Member Functions

- static NavExternalLinkData Create (long connectedVolumeID, int connectedRegionID, Vector3 fromPosition, Vector3 toPosition)

  Create a new NavExternalLinkData with the given properties.

Properties

- long ConnectedVolumeID `[get, set]`

  The NavVolume.InstanceID of the connected volume.
- int ConnectedRegionID `[get]`

  The ID of the connected region.
- Vector3 FromPosition `[get]`

  The position at which the connection originates (local space).
- Vector3 ToPosition `[get]`

  The position at which the connection ends (local space).

Private Attributes

- long _connectedVolumeID

  (Serialized) The NavVolume.InstanceID of the connected volume.
- int _connectedRegionID

  The ID of the connected region.
- Vector3 _fromPosition

  The position at which the connection originates (local space).
- Vector3 _toPosition

  The position at which the connection ends (local space).

6.29.1 Detailed Description

A connection from one region to another region in another volume.

6.29.2 Member Function Documentation

6.29.2.1 Create()

```
static NavExternalLinkData Infohazard.HyperNav.NavExternalLinkData.Create (
        long connectedVolumeID,
        int connectedRegionID,
        Vector3 fromPosition,
        Vector3 toPosition )  [static]
```

Create a new NavExternalLinkData with the given properties.

Parameters

connectedVolumeID	ID of the connected volume.
connectedRegionID	ID of the connected region.
fromPosition	Position at which the connection originates.
toPosition	Position at which the connection ends.

Returns

The created NavExternalLinkData.

6.29.2.2 ToInternalData()

```
void Infohazard.HyperNav.NavExternalLinkData.ToInternalData (
            Transform volumeTransform,
            bool volumeLocalSpace,
            out NativeNavExternalLinkData data )
```

Convert to a native representation, transforming points to world space if necessary.

Parameters

volumeTransform	Transform of the NavVolume.
volumeLocalSpace	Whether link is stored in local space.
data	The created native data.

6.29.3 Member Data Documentation

6.29.3.1 _connectedRegionID

```
int Infohazard.HyperNav.NavExternalLinkData._connectedRegionID
[private]
```

The ID of the connected region.

6.29.3.2 _connectedVolumeID

```
long Infohazard.HyperNav.NavExternalLinkData._connectedVolumeID
[private]
```

(Serialized) The NavVolume.InstanceID of the connected volume.

6.29.3.3 _fromPosition

```
Vector3 Infohazard.HyperNav.NavExternalLinkData._fromPosition [private]
```

The position at which the connection originates (local space).

6.29.3.4 _toPosition

```
Vector3 Infohazard.HyperNav.NavExternalLinkData._toPosition [private]
```

The position at which the connection ends (local space).

6.29.4 Property Documentation

6.29.4.1 ConnectedRegionID `int Infohazard.HyperNav.NavExternalLinkData.ConnectedRegionID [get]`

The ID of the connected region.

6.29.4.2 ConnectedVolumeID `long Infohazard.HyperNav.NavExternalLinkData.ConnectedVolumeID [get], [set]`

The NavVolume.InstanceID of the connected volume.

6.29.4.3 FromPosition `Vector3 Infohazard.HyperNav.NavExternalLinkData.FromPosition [get]`

The position at which the connection originates (local space).

6.29.4.4 ToPosition `Vector3 Infohazard.HyperNav.NavExternalLinkData.ToPosition [get]`

The position at which the connection ends (local space).

The documentation for this class was generated from the following file:

- Runtime/NavVolumeData.cs

6.30 Infohazard.HyperNav.NavHit Struct Reference

Structure that is used to report the nearest point on a NavVolume to a query.

Properties

- NavVolume Volume `[get, set]`

  The NavVolume that was hit.
- int Region `[get, set]`

  The region index within the hit Volume.
- bool IsOnEdge `[get, set]`

  If true, query point was outside the region and thus this result is the nearest point. If false, query point was inside the region and the hit was at that exact position.
- Vector3 Position `[get, set]`

  The position of the query result point.
- Vector3 Normal `[get, set]`

  Currently not used and always Vector3.zero.
- bool IsValid `[get]`

  Whether a valid Volume and Region were hit.

6.30.1 Detailed Description

Structure that is used to report the nearest point on a NavVolume to a query.

6.30.2 Property Documentation

6.30.2.1 IsOnEdge `bool Infohazard.HyperNav.NavHit.IsOnEdge [get], [set]`

If true, query point was outside the region and thus this result is the nearest point. If false, query point was inside the region and the hit was at that exact position.

6.30.2.2 IsValid `bool Infohazard.HyperNav.NavHit.IsValid [get]`

Whether a valid Volume and Region were hit.

6.30.2.3 Normal `Vector3 Infohazard.HyperNav.NavHit.Normal [get], [set]`

Currently not used and always Vector3.zero.

6.30.2.4 Position `Vector3 Infohazard.HyperNav.NavHit.Position [get], [set]`

The position of the query result point.

6.30.2.5 Region `int Infohazard.HyperNav.NavHit.Region [get], [set]`

The region index within the hit Volume.

6.30.2.6 Volume `NavVolume Infohazard.HyperNav.NavHit.Volume [get], [set]`

The NavVolume that was hit.

The documentation for this struct was generated from the following file:

- Runtime/NavHit.cs

6.31 Infohazard.HyperNav.NavInternalLinkData Class Reference

A connection from one region to another region in the same volume.

Static Public Member Functions

- static NavInternalLinkData Create (int connectedRegionID, int[ ] vertices, Edge[ ] edges, Triangle[ ] triangles)

  Create a new NavInternalLinkData with the given properties.

Properties

- int ConnectedRegionID `[get]`

  The ID of the connected region.
- IReadOnlyList< int > Vertices `[get]`

  The indices of vertices that the two regions share.
- IReadOnlyList< Edge > Edges `[get]`

  The indices of edges that the two regions share.
- IReadOnlyList< Triangle > Triangles `[get]`

  The indices of triangles that the two regions share.

Private Attributes

- int _connectedRegionID

  (Serialized) The ID of the connected region.
- int[ ] _vertices

  (Serialized) The indices of vertices that the two regions share.
- Edge[ ] _edges

  (Serialized) The indices of edges that the two regions share.
- Triangle[ ] _triangles

  (Serialized) The indices of triangles that the two regions share.

6.31.1 Detailed Description

A connection from one region to another region in the same volume.

6.31.2 Member Function Documentation

6.31.2.1 Create()

```
static NavInternalLinkData Infohazard.HyperNav.NavInternalLinkData.Create (
        int connectedRegionID,
        int[ ] vertices,
        Edge[ ] edges,
        Triangle[ ] triangles ) [static]
```

Create a new NavInternalLinkData with the given properties.

Parameters

connectedRegionID	ID of the connected region.
vertices	Shared vertices.
edges	Shared edges.
triangles	Shared triangles.

Returns

The created NavInternalLinkData.

6.31.3 Member Data Documentation

6.31.3.1 _connectedRegionID `int Infohazard.HyperNav.NavInternalLinkData._connectedRegionID [private]`

(Serialized) The ID of the connected region.

6.31.3.2 _edges `Edge [] Infohazard.HyperNav.NavInternalLinkData._edges [private]`

(Serialized) The indices of edges that the two regions share.

6.31.3.3 _triangles `Triangle [] Infohazard.HyperNav.NavInternalLinkData._triangles [private]`

(Serialized) The indices of triangles that the two regions share.

6.31.3.4 _vertices `int [] Infohazard.HyperNav.NavInternalLinkData._vertices [private]`

(Serialized) The indices of vertices that the two regions share.

6.31.4 Property Documentation

6.31.4.1 ConnectedRegionID `int Infohazard.HyperNav.NavInternalLinkData.ConnectedRegionID [get]`

The ID of the connected region.

6.31.4.2 Edges `IReadOnlyList<Edge> Infohazard.HyperNav.NavInternalLinkData.Edges [get]`

The indices of edges that the two regions share.

6.31.4.3 Triangles `IReadOnlyList<Triangle> Infohazard.HyperNav.NavInternalLinkData.Triangles [get]`

The indices of triangles that the two regions share.

6.31.4.4 Vertices `IReadOnlyList<int> Infohazard.HyperNav.NavInternalLinkData.Vertices [get]`

The indices of vertices that the two regions share.

The documentation for this class was generated from the following file:

- Runtime/NavVolumeData.cs

6.32 Infohazard.HyperNav.Jobs.NavMultiRaycastJob Struct Reference

Job that performs multiple raycasts in one or more NavVolumes in parallel.

Public Member Functions

- void Execute (int index)

    Execute the job for the raycast at index.

Public Attributes

- NativeParallelHashMap< long, NativeNavVolumeData > Volumes

    All loaded volume data.
- NativeArray< NativeRaycastElement > Raycasts

    The raycasts to perform (results are stored in each element's NativeRaycastElement.OutDistance.

6.32.1 Detailed Description

Job that performs multiple raycasts in one or more NavVolumes in parallel.

6.32.2 Member Function Documentation

6.32.2.1 Execute() `void Infohazard.HyperNav.Jobs.NavMultiRaycastJob.Execute ( int index )`

Execute the job for the raycast at index.

Parameters

index	Index of the raycast to execute.

6.32.3 Member Data Documentation

6.32.3.1 Raycasts `NativeArray<NativeRaycastElement> Infohazard.HyperNav.Jobs.NavMultiRaycastJob.Raycasts`

The raycasts to perform (results are stored in each element's NativeRaycastElement.OutDistance.

6.32.3.2 Volumes `NativeParallelHashMap<long, NativeNavVolumeData> Infohazard.HyperNav.Jobs.NavMultiRaycastJob.Volumes`

All loaded volume data.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavRaycastJob.cs

6.33 Infohazard.HyperNav.NavPath Class Reference

A completed, valid path.

Public Member Functions

- void Dispose ()

  Dispose the path, returning it to an object pool.

Properties

- long ID = -1 `[get, set]`

  ID of the path.
- bool HasBeenDisposed = true `[get, set]`

  Whether the path has been disposed.
- Vector3 StartPos `[get, set]`

  The position where the path originates from.
- Vector3 EndPos `[get, set]`

  The destination of the path.
- NavHit StartHit `[get, set]`

  The navigation query result at the start of the path.
- NavHit EndHit `[get, set]`

  The navigation query result at the end of the path.
- NavPathfinder Pathfinder `[get, set]`

  The NavPathfinder that was used to calculate the path.
- List< NavWaypoint > InternalWaypoints = new List<NavWaypoint>() `[get]`

  Mutable list of waypoints of the path.
- IReadOnlyList< NavWaypoint > Waypoints `[get]`

  List of waypoints of the path.

6.33.1 Detailed Description

A completed, valid path.

6.33.2 Member Function Documentation

6.33.2.1 Dispose() `void Infohazard.HyperNav.NavPath.Dispose ( )`

Dispose the path, returning it to an object pool.

6.33.3 Property Documentation

6.33.3.1 EndHit `NavHit Infohazard.HyperNav.NavPath.EndHit [get], [set]`

The navigation query result at the end of the path.

6.33.3.2 EndPos `Vector3 Infohazard.HyperNav.NavPath.EndPos [get], [set]`

The destination of the path.

6.33.3.3 HasBeenDisposed `bool Infohazard.HyperNav.NavPath.HasBeenDisposed = true [get], [set]`

Whether the path has been disposed.

6.33.3.4 ID `long Infohazard.HyperNav.NavPath.ID = -1 [get], [set]`

ID of the path.

6.33.3.5 InternalWaypoints `List<NavWaypoint> Infohazard.HyperNav.NavPath.InternalWaypoints = new List<NavWaypoint>() [get], [package]`

Mutable list of waypoints of the path.

6.33.3.6 Pathfinder `NavPathfinder Infohazard.HyperNav.NavPath.Pathfinder [get], [set], [package]`

The NavPathfinder that was used to calculate the path.

6.33.3.7 StartHit `NavHit Infohazard.HyperNav.NavPath.StartHit [get], [set]`

The navigation query result at the start of the path.

6.33.3.8 StartPos `Vector3 Infohazard.HyperNav.NavPath.StartPos [get], [set]`

The position where the path originates from.

6.33.3.9 Waypoints `IReadOnlyList<NavWaypoint> Infohazard.HyperNav.NavPath.Waypoints [get]`

List of waypoints of the path.

The documentation for this class was generated from the following file:

- Runtime/NavPathfinder.cs

6.34 Infohazard.HyperNav.NavPathfinder Class Reference

A script used to calculate HyperNav paths.

Classes

- class PropNames

  This is used to refer to the names of private fields in this class from a custom Editor.

Public Member Functions

- long FindPath (Vector3 start, Vector3 end, HyperNavPathCallback receiver, float sampleRadius=0)

  Find a path between two positions, and invoke the receiver when it is completed.
- long FindPath (NavHit startHit, NavHit endHit, Vector3 startPos, Vector3 endPos, HyperNavPathCallback receiver)

  Find a path between two already-calculated nav query results, and invoke the receiver when it is completed.
- void CancelPath (long id, bool logError=true)

  Cancel a pending path with the given ID.

Protected Member Functions

- virtual void OnEnable ()

  If IsMainInstance is true, set MainInstance or log an error if it is already set.
- virtual void OnDisable ()

  Dispose the pools of pending paths and completed paths, and all memory allocated for pathfinding jobs.
- virtual void Update ()

  If mode is JobThread, check job completion. If mode is MainThreadAsynchronous, perform pathfinding work.

Package Functions

- void DisposePath (NavPath path)

  Called by NavPath.Dispose.

Properties

- bool IsMainInstance `[get, set]`

  Whether to set NavPathfinder.Instance to this instance.
- NavPathfindingMode PathfindingMode `[get, set]`

  The mode to use for calculating paths.
- int MaxExecutingRequests `[get, set]`

  (MainThreadAsynchronous Mode ONLY) If greater than zero, limit on the number of requests actively being worked on.
- int MaxPathOpsPerFrame `[get, set]`

  (MainThreadAsynchronous Mode ONLY) Maximum total number of pathfinding steps that can be performed per frame by this instance.
- int MaxConcurrentJobs `[get, set]`

  (JobThread Mode ONLY) Maximum number of pathfinding jobs that can be actively running at once.
- int MaxCompletionFrames `[get, set]`

  (JobThread Mode ONLY) Maximum number of frames a job can take before the main thread must wait for it.
- static NavPathfinder MainInstance `[get, private set]`

  The main instance, which should be used in most situations.

Private Member Functions

- void LateUpdate ()

  Move paths from the pending queue and start executing them.

Private Attributes

- bool _isMainInstance = true

  (Serialized) Whether to set NavPathfinder.Instance to this instance.
- NavPathfindingMode _pathfindingMode = NavPathfindingMode.JobThread

  (Serialized) The mode to use for calculating paths.
- int _maxExecutingRequests = 0

  (Serialized) If greater than zero, limit on the number of requests actively being worked on.
- int _maxPathOpsPerFrame = 100

  (Serialized) Maximum total number of pathfinding steps that can be performed per frame by this instance.
- int _maxConcurrentJobs = 1

  (Serialized) Maximum number of pathfinding jobs that can be actively running at once.
- int _maxCompletionFrames = 3

  (Serialized) Maximum number of frames a job can take before the main thread must wait for it.

6.34.1 Detailed Description

A script used to calculate HyperNav paths.

Can be used as a singleton, or you can have more than one if needed.

6.34.2 Member Function Documentation

6.34.2.1 CancelPath()

```
void Infohazard.HyperNav.NavPathfinder.CancelPath (
        long id,
        bool logError = true )
```

Cancel a pending path with the given ID.

If the mode is set to JobThread and the requested path is already executing, the actual work thread cannot be cancelled. However, this will still remove the receiver, so no matter what that will not be called for the path.

Parameters

id	The path ID to cancel.
logError	Whether to log an error if the path is not running.

6.34.2.2 DisposePath()

```
void Infohazard.HyperNav.NavPathfinder.DisposePath (
        NavPath path )   [package]
```

Called by NavPath.Dispose.

Parameters

path	The path to dispose.

6.34.2.3 FindPath() [1/2]

```
long Infohazard.HyperNav.NavPathfinder.FindPath (
        NavHit startHit,
        NavHit endHit,
        Vector3 startPos,
        Vector3 endPos,
        HyperNavPathCallback receiver )
```

Find a path between two already-calculated nav query results, and invoke the receiver when it is completed.

If pathfinding cannot occur, for example because there are no volumes, this method will return -1 and the receiver will not be invoked. If no path can be found, the receiver will be invoked with a null Path argument.

Parameters

startHit	Query result for the start of the path.
endHit	Query result for the end of the path.
startPos	Start position for the path.
endPos	Destination for the path.
receiver	Callback to receive the path when it has been calculated.

Returns

The ID of the pending path, or -1 if pathfinding cannot occur.

6.34.2.4 FindPath() [2/2]

```
long Infohazard.HyperNav.NavPathfinder.FindPath (
        Vector3 start,
        Vector3 end,
        HyperNavPathCallback receiver,
        float sampleRadius = 0 )
```

Find a path between two positions, and invoke the receiver when it is completed.

If pathfinding cannot occur, for example because there are no volumes, or because a query at start or end fails, this method will return -1 and the receiver will not be invoked. If no path can be found, the receiver will be invoked with a null Path argument.

Parameters

start	Start position for the path.
end	Destination for the path.
receiver	Callback to receive the path when it has been calculated.
sampleRadius	Radius to search for volumes at the start and end locations./>

Returns

The ID of the pending path, or -1 if pathfinding cannot occur.

6.34.2.5 LateUpdate()

```
void Infohazard.HyperNav.NavPathfinder.LateUpdate ( ) [private]
```

Move paths from the pending queue and start executing them.

6.34.2.6 OnDisable()

```
virtual void Infohazard.HyperNav.NavPathfinder.OnDisable ( ) [protected], [virtual]
```

Dispose the pools of pending paths and completed paths, and all memory allocated for pathfinding jobs.

6.34.2.7 OnEnable() `virtual void Infohazard.HyperNav.NavPathfinder.OnEnable ( ) [protected], [virtual]`

If IsMainInstance is true, set MainInstance or log an error if it is already set.

6.34.2.8 Update() `virtual void Infohazard.HyperNav.NavPathfinder.Update ( ) [protected], [virtual]`

If mode is JobThread, check job completion. If mode is MainThreadAsynchronous, perform pathfinding work.

6.34.3 Member Data Documentation

6.34.3.1 _isMainInstance `bool Infohazard.HyperNav.NavPathfinder._isMainInstance = true [private]`

(Serialized) Whether to set NavPathfinder.Instance to this instance.

6.34.3.2 _maxCompletionFrames `int Infohazard.HyperNav.NavPathfinder._maxCompletionFrames = 3 [private]`

(Serialized) Maximum number of frames a job can take before the main thread must wait for it.

6.34.3.3 _maxConcurrentJobs `int Infohazard.HyperNav.NavPathfinder._maxConcurrentJobs = 1 [private]`

(Serialized) Maximum number of pathfinding jobs that can be actively running at once.

6.34.3.4 _maxExecutingRequests `int Infohazard.HyperNav.NavPathfinder._maxExecutingRequests = 0 [private]`

(Serialized) If greater than zero, limit on the number of requests actively being worked on.

6.34.3.5 _maxPathOpsPerFrame `int Infohazard.HyperNav.NavPathfinder._maxPathOpsPerFrame = 100 [private]`

(Serialized) Maximum total number of pathfinding steps that can be performed per frame by this instance.

6.34.3.6 _pathfindingMode `NavPathfindingMode Infohazard.HyperNav.NavPathfinder._pathfinding↩ Mode = NavPathfindingMode.JobThread [private]`

(Serialized) The mode to use for calculating paths.

6.34.4 Property Documentation

6.34.4.1 IsMainInstance `bool Infohazard.HyperNav.NavPathfinder.IsMainInstance [get], [set]`

Whether to set NavPathfinder.Instance to this instance.

This cannot be set while the game is running.

6.34.4.2 MainInstance `NavPathfinder Infohazard.HyperNav.NavPathfinder.MainInstance [static], [get], [private set]`

The main instance, which should be used in most situations.

If you need more than one NavPathfinder, you can use direct references to other instances with IsMainInstance set to false.

6.34.4.3 MaxCompletionFrames `int Infohazard.HyperNav.NavPathfinder.MaxCompletionFrames [get], [set]`

(JobThread Mode ONLY) Maximum number of frames a job can take before the main thread must wait for it.

Unity imposes a limit of 3 frames for faster TempJob allocations, so increasing this beyond 3 will slightly decrease memory performance. If a job takes longer than this and is forced to block the main thread, a warning will be logged showing you how long it blocked the main thread for. This cannot be set while any paths are executing.

6.34.4.4 MaxConcurrentJobs `int Infohazard.HyperNav.NavPathfinder.MaxConcurrentJobs [get], [set]`

(JobThread Mode ONLY) Maximum number of pathfinding jobs that can be actively running at once.

Requests beyond this number are queued. You should keep this number fairly low, as each job has the potential to take up a CPU thread.

6.34.4.5 MaxExecutingRequests `int Infohazard.HyperNav.NavPathfinder.MaxExecutingRequests [get], [set]`

(MainThreadAsynchronous Mode ONLY) If greater than zero, limit on the number of requests actively being worked on.

6.34.4.6 MaxPathOpsPerFrame `int Infohazard.HyperNav.NavPathfinder.MaxPathOpsPerFrame [get], [set]`

(MainThreadAsynchronous Mode ONLY) Maximum total number of pathfinding steps that can be performed per frame by this instance.

This limit is shared by all executing paths in this instance.

6.34.4.7 PathfindingMode `NavPathfindingMode Infohazard.HyperNav.NavPathfinder.PathfindingMode [get], [set]`

The mode to use for calculating paths.

This cannot be set while the game is running.

The documentation for this class was generated from the following file:

- Runtime/NavPathfinder.cs

6.35 Infohazard.HyperNav.Jobs.NavPathJob Struct Reference

Burst-compatible job used to find a HyperNav path.

Public Member Functions

- void Execute ()

  Execute the pathfinding operation all the way through.
- void UpdatePath (int operationsLimit, out int operationsUsed, out NavPathState state)

  Execute pathfinding algorithm up to the given number of steps.
- void Initialize ()

  Initialize the pathfinding algorithm.

Public Attributes

- NativeParallelHashMap< long, NativeNavVolumeData > Volumes

  Map containing all loaded NavVolumes, keyed by their instance ID>
- float4 StartPosition

  Position where the path starts (world space).
- NativeNavHit StartHit

  Nav query result where the path starts.
- NativeNavHit EndHit

  Nav query result where the path ends.
- NativeList< NativeNavWaypoint > OutPathWaypoints

  Used to return the result path (as a list of waypoints) back to managed code.
- NativeParallelHashMap< PendingPathNode, VisitedNodeInfo > NodeTable

  Map containing all discovered nodes in the current pathfinding operation.
- NativeList< PendingPathNode > Waypoints

  Internal list for holding the in-progress path waypoints.
- NativeHeap< PendingPathNode > Frontier

  Internal heap for holding queue of nodes to visit.

6.35.1 Detailed Description

Burst-compatible job used to find a HyperNav path.

The methods in this class can be used both as a job and called directly from managed code. This enables pathfinding to operate in different modes without duplicating any of this code.

6.35.2 Member Function Documentation

6.35.2.1 Execute() `void Infohazard.HyperNav.Jobs.NavPathJob.Execute ( )`

Execute the pathfinding operation all the way through.

6.35.2.2 Initialize() `void Infohazard.HyperNav.Jobs.NavPathJob.Initialize ( )`

Initialize the pathfinding algorithm.

Must be called before UpdatePath can be called directly from managed code.

6.35.2.3 UpdatePath()

```
void Infohazard.HyperNav.Jobs.NavPathJob.UpdatePath (
        int operationsLimit,
        out int operationsUsed,
        out NavPathState state )
```

Execute pathfinding algorithm up to the given number of steps.

When called directly from managed code, Initialize must be called first.

Parameters

operationsLimit	Maximum number of pathfinding steps.
operationsUsed	Actual number of pathfinding steps used.
state	State of the pathfinding algorithm upon return.

6.35.3 Member Data Documentation

6.35.3.1 EndHit `NativeNavHit Infohazard.HyperNav.Jobs.NavPathJob.EndHit`

Nav query result where the path ends.

6.35.3.2 Frontier `NativeHeap<PendingPathNode> Infohazard.HyperNav.Jobs.NavPathJob.Frontier`

Internal heap for holding queue of nodes to visit.

6.35.3.3 NodeTable `NativeParallelHashMap<PendingPathNode, VisitedNodeInfo> Infohazard.HyperNav.Jobs.NavPathJob.NodeTable`

Map containing all discovered nodes in the current pathfinding operation.

6.35.3.4 OutPathWaypoints `NativeList<NativeNavWaypoint> Infohazard.HyperNav.Jobs.NavPathJob.OutPathWaypoints`

Used to return the result path (as a list of waypoints) back to managed code.

6.35.3.5 StartHit `NativeNavHit Infohazard.HyperNav.Jobs.NavPathJob.StartHit`

Nav query result where the path starts.

6.35.3.6 StartPosition `float4 Infohazard.HyperNav.Jobs.NavPathJob.StartPosition`

Position where the path starts (world space).

6.35.3.7 Volumes `NativeParallelHashMap<long, NativeNavVolumeData> Infohazard.HyperNav.Jobs.NavPathJob.Volumes`

Map containing all loaded NavVolumes, keyed by their instance ID>

6.35.3.8 Waypoints `NativeList<PendingPathNode> Infohazard.HyperNav.Jobs.NavPathJob.Waypoints`

Internal list for holding the in-progress path waypoints.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavPathJob.cs

6.36 Infohazard.HyperNav.Jobs.NavRaycastJob Struct Reference

Job that performs a single raycast in a NavVolume.

Public Member Functions

- void Execute ()

  Execute the raycast.

Public Attributes

- NativeNavVolumeData Volume

  Volume to raycast in.
- float4 Start

  Start point of the segment.
- float4 End

  End point of the segment.
- NativeArray< float > OutDistance

  A single-element array where the hit distance (or -1 if no hit) is written.

6.36.1 Detailed Description

Job that performs a single raycast in a NavVolume.

6.36.2 Member Function Documentation

6.36.2.1 Execute() `void Infohazard.HyperNav.Jobs.NavRaycastJob.Execute ( )`

Execute the raycast.

6.36.3 Member Data Documentation

6.36.3.1 End `float4 Infohazard.HyperNav.Jobs.NavRaycastJob.End`

End point of the segment.

6.36.3.2 OutDistance `NativeArray<float> Infohazard.HyperNav.Jobs.NavRaycastJob.OutDistance`

A single-element array where the hit distance (or -1 if no hit) is written.

6.36.3.3 Start `float4 Infohazard.HyperNav.Jobs.NavRaycastJob.Start`

Start point of the segment.

6.36.3.4 Volume `NativeNavVolumeData Infohazard.HyperNav.Jobs.NavRaycastJob.Volume`

Volume to raycast in.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavRaycastJob.cs

6.37 Infohazard.HyperNav.NavRegionBoundPlane Struct Reference

A plane forming one of the boundaries of a region.

Static Public Member Functions

- static NavRegionBoundPlane Create (Vector3 normal, int intersectVertex)

  Create a new NavRegionBoundPlane with the given properties.

Properties

- Vector3 Normal `[get]`

  Normal of the plane.
- int IntersectVertex `[get]`

  Index of a vertex in the volume that the plane intersects.

Private Attributes

- Vector3 _normal

  (Serialized) Normal of the plane.
- int _intersectVertex

  (Serialized) Index of a vertex in the volume that the plane intersects.

6.37.1 Detailed Description

A plane forming one of the boundaries of a region.

6.37.2 Member Function Documentation

6.37.2.1 Create()

```
static NavRegionBoundPlane Infohazard.HyperNav.NavRegionBoundPlane.Create (
        Vector3 normal,
        int intersectVertex ) [static]
```

Create a new NavRegionBoundPlane with the given properties.

Parameters

normal	Normal of the plane.
intersectVertex	Index of a vertex that the plane intersects.

Returns

The created NavRegionBoundPlane.

6.37.3 Member Data Documentation

6.37.3.1 _intersectVertex `int Infohazard.HyperNav.NavRegionBoundPlane._intersectVertex [private]`

(Serialized) Index of a vertex in the volume that the plane intersects.

6.37.3.2 _normal `Vector3 Infohazard.HyperNav.NavRegionBoundPlane._normal [private]`

(Serialized) Normal of the plane.

6.37.4 Property Documentation

6.37.4.1 IntersectVertex `int Infohazard.HyperNav.NavRegionBoundPlane.IntersectVertex [get]`

Index of a vertex in the volume that the plane intersects.

6.37.4.2 Normal `Vector3 Infohazard.HyperNav.NavRegionBoundPlane.Normal [get]`

Normal of the plane.

The documentation for this struct was generated from the following file:

- Runtime/NavVolumeData.cs

6.38 Infohazard.HyperNav.NavRegionData Class Reference

The serialized data representing a single region in a NavVolume.

Public Member Functions

- void SetExternalLinks (NavExternalLinkData[ ] externalConnections)

  Update the ExternalLinks of the region.

Static Public Member Functions

- static NavRegionData Create (int id, int[ ] indices, Bounds bounds, NavInternalLinkData[ ] internalLinks, NavRegionBoundPlane[ ] boundPlanes)

  Construct a new NavRegionData with the given values.

Properties

- int ID `[get]`

  The ID of the region.
- Bounds Bounds `[get]`

  The bounds of the region in local space of the volume.
- IReadOnlyList< int > Indices `[get]`

  The indices of the region's triangle vertices in the volume's vertices array.
- IReadOnlyList< NavInternalLinkData > InternalLinks `[get]`

  The links between this region and other regions in the same volume.
- IReadOnlyList< NavExternalLinkData > ExternalLinks `[get]`

  The links between this region and regions in other volumes.
- IReadOnlyList< NavRegionBoundPlane > BoundPlanes `[get]`

  The planes that form the boundaries of this region, to check if a point is inside or not.

Private Attributes

- int _id

  (Serialized) The ID of the region.
- Bounds _bounds

  (Serialized) The bounds of the region in local space of the volume.
- int[ ] _indices

  (Serialized) The indices of the region's triangle vertices in the volume's vertices array.
- NavInternalLinkData[ ] _internalLinks

  (Serialized) The links between this region and other regions in the same volume.
- NavExternalLinkData[ ] _externalLinks

  (Serialized) The links between this region and regions in other volumes.
- NavRegionBoundPlane[ ] _boundPlanes

  (Serialized) The planes that form the boundaries of this region, to check if a point is inside or not.

6.38.1 Detailed Description

The serialized data representing a single region in a NavVolume.

6.38.2 Member Function Documentation

6.38.2.1 Create()

```
static NavRegionData Infohazard.HyperNav.NavRegionData.Create (
        int id,
        int[] indices,
        Bounds bounds,
        NavInternalLinkData[] internalLinks,
        NavRegionBoundPlane[] boundPlanes )  [static]
```

Construct a new NavRegionData with the given values.

No value for ExternalLinks is provided here because that must be calculated later.

Parameters

id	ID of the region.
indices	Indices of the region triangles.
bounds	Bounds of the region.
internalLinks	Links to other regions in same volume.
boundPlanes	Planes that form the boundaries of the region.

Returns

The created NavRegionData.

6.38.2.2 SetExternalLinks()

```
void Infohazard.HyperNav.NavRegionData.SetExternalLinks (
        NavExternalLinkData[] externalConnections )
```

Update the ExternalLinks of the region.

Parameters

externalConnections	The list of external links to use.

6.38.3 Member Data Documentation

6.38.3.1 _boundPlanes

```
NavRegionBoundPlane [] Infohazard.HyperNav.NavRegionData._boundPlanes [private]
```

(Serialized) The planes that form the boundaries of this region, to check if a point is inside or not.

6.38.3.2 _bounds

```
Bounds Infohazard.HyperNav.NavRegionData._bounds [private]
```

(Serialized) The bounds of the region in local space of the volume.

6.38.3.3 _externalLinks `NavExternalLinkData [ ] Infohazard.HyperNav.NavRegionData._externalLinks [private]`

(Serialized) The links between this region and regions in other volumes.

6.38.3.4 _id `int Infohazard.HyperNav.NavRegionData._id [private]`

(Serialized) The ID of the region.

6.38.3.5 _indices `int [ ] Infohazard.HyperNav.NavRegionData._indices [private]`

(Serialized) The indices of the region's triangle vertices in the volume's vertices array.

6.38.3.6 _internalLinks `NavInternalLinkData [ ] Infohazard.HyperNav.NavRegionData._internalLinks [private]`

(Serialized) The links between this region and other regions in the same volume.

6.38.4 Property Documentation

6.38.4.1 BoundPlanes `IReadOnlyList<NavRegionBoundPlane> Infohazard.HyperNav.NavRegionData.↩ BoundPlanes [get]`

The planes that form the boundaries of this region, to check if a point is inside or not.

6.38.4.2 Bounds `Bounds Infohazard.HyperNav.NavRegionData.Bounds [get]`

The bounds of the region in local space of the volume.

6.38.4.3 ExternalLinks `IReadOnlyList<NavExternalLinkData> Infohazard.HyperNav.NavRegionData.↩ ExternalLinks [get]`

The links between this region and regions in other volumes.

6.38.4.4 ID `int Infohazard.HyperNav.NavRegionData.ID [get]`

The ID of the region.

6.38.4.5 Indices `IReadOnlyList<int> Infohazard.HyperNav.NavRegionData.Indices [get]`

The indices of the region's triangle vertices in the volume's vertices array.

6.38.4.6 InternalLinks `IReadOnlyList<NavInternalLinkData> Infohazard.HyperNav.NavRegionData.↩ InternalLinks [get]`

The links between this region and other regions in the same volume.

The documentation for this class was generated from the following file:

- Runtime/NavVolumeData.cs

6.39 Infohazard.HyperNav.NavUtil Class Reference

Contains utility methods for working with HyperNav navigation.

Static Public Member Functions

- static bool SamplePosition (Vector3 position, out NavHit hit, float maxDistance)

  Perform a query to find the nearest point on any NavVolume to the given point.

6.39.1 Detailed Description

Contains utility methods for working with HyperNav navigation.

6.39.2 Member Function Documentation

6.39.2.1 SamplePosition()

```
static bool Infohazard.HyperNav.NavUtil.SamplePosition (
            Vector3 position,
            out NavHit hit,
            float maxDistance ) [static]
```

Perform a query to find the nearest point on any NavVolume to the given point.

Parameters

position	The point at which to search.
hit	The resulting hit, containing the nearest point on a volume.
maxDistance	The radius in which to search (a larger value is more expensive).

Returns

Whether a hit on any volume could be found in the given radius.

The documentation for this class was generated from the following file:

- Runtime/NavUtil.cs

6.40 Infohazard.HyperNav.NavVolume Class Reference

A volume of space in which HyperNav pathfinding can occur.

Classes

- class PropNames

  This is used to refer to the names of private fields in this class from a custom Editor.

Public Member Functions

- virtual bool SamplePosition (Vector3 position, out NavHit hit, float maxDistance)

  Perform a query to find the nearest point on this volume to the given point.
- bool Raycast (Vector3 start, Vector3 end, out float hit)

  Cast a ray against the blocking triangles of the volume, and return the nearest hit.
- void UpdateTransform ()

  Update the native data of this NavVolume.

Static Public Member Functions

- static void UpdateAllTransforms ()

  Update the native data on all loaded NavVolumes.

Static Public Attributes

- static NativeParallelHashMap< long, NativeNavVolumeData > VolumeData

  Data for all loaded volumes in the format used by jobs.

Protected Member Functions

- virtual void OnEnable ()

  Register this volume in the Volumes dictionary and perform initialization.
- virtual void OnDisable ()

  Remove this volume from the Volumes dictionary.
- virtual void OnDestroy ()

  Dispose native-side data for this volume.
- virtual void Update ()

  Update UniqueID in editor, and check movement.

Properties

- Bounds Bounds `[get, set]`

  The boundaries of the volume.
- NavVolumeData Data `[get, set]`

  The baked data for the volume.
- long InstanceID `[get]`

  The unique ID for this volume to identify it in pathfinding jobs and serialized data.
- bool AutoDetectMovement `[get, set]`

  Whether to automatically update native data if the volume moves.
- float VoxelSize `[get, set]`

  The voxel size of this volume, which determines the precision but also baking cost.
- float MaxAgentRadius `[get, set]`

  The maximum size of agents using this volume.
- bool EnableMultiQuery `[get, set]`

  Whether to enable multiple physics queries per voxel to get a more accurate result.
- float MaxExternalLinkDistance `[get, set]`

  The maximum distance that external links can extend outside of this volume.
- LayerMask BlockingLayers `[get, set]`

  Which layers are considered impassible for pathfinding.
- bool StaticOnly `[get, set]`

  Whether only static objects should be included in the baked data.
- bool UseStartLocations `[get, set]`

  Whether only regions connected to certain locations are considered valid.
- IReadOnlyList< Vector3 > StartLocations `[get, set]`

  If _useStartLocations is true, which start locations to use.
- bool UseMultithreading `[get, set]`

  Whether to use multiple threads when baking the volume.
- NavVolumeVisualizationMode VisualizationMode `[get, set]`

  Stage at which to visualize the volume bake process in the scene view.
- bool VisualizeNeighbors `[get, set]`

  Whether to show the connections of a selected region in the scene view.
- int VisualizeNeighborsRegion `[get, set]`

  If _visualizeNeighbors is true, which region to visualize in the scene view.
- bool ShowVertexNumbers `[get, set]`

  Whether to show the vertex numbers of the preview mesh in the scene view (for debugging).
- float ShowVertexNumbersRange `[get, set]`

  Max distance from the camera at which vertex numbers will be shown.
- bool VisualizeVoxelQueries `[get, set]`

  Whether to visualize the queries that are performed for a voxel when baking.
- static int VolumeChangingCount `[get, private set]`

  Number of places that are modifying volume data.
- static IReadOnlyDictionary< long, NavVolume > Volumes `[get]`

  All currently loaded volumes.

Events

- static Action VolumeDataChanging

  Event that is invoked immediately before active volume data changes.
- static Action VolumeDataChanged

  Event that is invoked immediately after active volume data changes.

Private Attributes

- Bounds _bounds = new Bounds(Vector3.zero, Vector3.one)

  (Serialized) The boundaries of the volume.
- NavVolumeData _data

  (Serialized) The baked data for the volume.
- long _instanceID

  (Serialized) The unique ID for this volume to identify it in pathfinding jobs and serialized data.
- bool _autoDetectMovement = false

  (Serialized) Whether to automatically update native data if the volume moves.
- LayerMask _blockingLayers = 1

  (Serialized) Which layers are considered impassible for pathfinding.
- bool _staticOnly = true

  (Serialized) Whether only static objects should be included in the baked data.
- float _maxAgentRadius = 1

  (Serialized) The maximum size of agents using this volume.
- bool _enableMultiQuery = true

  (Serialized) Whether to enable multiple physics queries per voxel to get a more accurate result.
- float _maxExternalLinkDistance = 1

  (Serialized) The maximum distance that external links can extend outside of this volume.
- float _voxelSize = 1

  (Serialized) The voxel size of this volume, which determines the precision but also baking cost.
- bool _useStartLocations = false

  (Serialized) Whether only regions connected to certain locations are considered valid.
- Vector3[ ] _startLocations

  (Serialized) If _useStartLocations is true, which start locations to use.
- bool _useMultithreading = true

  (Serialized) Whether to use multiple threads when baking the volume.
- NavVolumeVisualizationMode _visualizationMode = NavVolumeVisualizationMode.Final

  (Serialized) Stage at which to visualize the volume bake process in the scene view.
- bool _visualizeNeighbors

  (Serialized) Whether to show the connections of a selected region in the scene view.
- int _visualizeNeighborsRegion

  (Serialized) If _visualizeNeighbors is true, which region to visualize in the scene view.
- bool _showVertexNumbers

  (Serialized) Whether to show the vertex numbers of the preview mesh in the scene view (for debugging).
- float _showVertexNumbersRange = 2

  (Serialized) Max distance from the camera at which vertex numbers will be shown.
- bool _visualizeVoxelQueries

  (Serialized) Whether to visualize the queries that are performed for a voxel when baking.

6.40.1 Detailed Description

A volume of space in which HyperNav pathfinding can occur.

Each NavVolume is divided into convex regions that form pathfinding nodes. A volume's regions can have connections to each other, and to regions of other volumes. The information in a NavVolume must be baked in the editor - it cannot be calculated at runtime (for now).

6.40.2 Member Function Documentation

6.40.2.1 OnDestroy() `virtual void Infohazard.HyperNav.NavVolume.OnDestroy ( ) [protected], [virtual]`

Dispose native-side data for this volume.

6.40.2.2 OnDisable() `virtual void Infohazard.HyperNav.NavVolume.OnDisable ( ) [protected], [virtual]`

Remove this volume from the Volumes dictionary.

6.40.2.3 OnEnable() `virtual void Infohazard.HyperNav.NavVolume.OnEnable ( ) [protected], [virtual]`

Register this volume in the Volumes dictionary and perform initialization.

6.40.2.4 Raycast()

```
bool Infohazard.HyperNav.NavVolume.Raycast (
        Vector3 start,
        Vector3 end,
        out float hit )
```

Cast a ray against the blocking triangles of the volume, and return the nearest hit.

Parameters

start	The position (in world space) to start the query at.
end	The position (in world space) to end the query at.
hit	If the query hits a triangle, the ratio between start and end at which the hit occurred.

Returns

Whether a triangle was hit.

6.40.2.5 SamplePosition()

```
virtual bool Infohazard.HyperNav.NavVolume.SamplePosition (
            Vector3 position,
            out NavHit hit,
            float maxDistance ) [virtual]
```

Perform a query to find the nearest point on this volume to the given point.

Parameters

position	The point at which to search.
hit	The resulting hit, containing the nearest point on this volume.
maxDistance	The radius in which to search (a larger value is more expensive).

Returns

Whether a hit on this volume could be found in the given radius.

6.40.2.6 Update()

```
virtual void Infohazard.HyperNav.NavVolume.Update ( ) [protected], [virtual]
```

Update UniqueID in editor, and check movement.

6.40.2.7 UpdateAllTransforms()

```
static void Infohazard.HyperNav.NavVolume.UpdateAllTransforms (
) [static]
```

Update the native data on all loaded NavVolumes.

Use this after moving all volumes when AutoDetectMovement is disabled.

6.40.2.8 UpdateTransform()

```
void Infohazard.HyperNav.NavVolume.UpdateTransform ( )
```

Update the native data of this NavVolume.

This is called automatically if AutoDetectMovement is enabled.

6.40.3 Member Data Documentation

6.40.3.1 _autoDetectMovement `bool Infohazard.HyperNav.NavVolume._autoDetectMovement = false [private]`

(Serialized) Whether to automatically update native data if the volume moves.

6.40.3.2 _blockingLayers `LayerMask Infohazard.HyperNav.NavVolume._blockingLayers = 1 [private]`

(Serialized) Which layers are considered impassible for pathfinding.

6.40.3.3 _bounds `Bounds Infohazard.HyperNav.NavVolume._bounds = new Bounds(Vector3.zero, Vector3.one) [private]`

(Serialized) The boundaries of the volume.

6.40.3.4 _data `NavVolumeData Infohazard.HyperNav.NavVolume._data [private]`

(Serialized) The baked data for the volume.

6.40.3.5 _enableMultiQuery `bool Infohazard.HyperNav.NavVolume._enableMultiQuery = true [private]`

(Serialized) Whether to enable multiple physics queries per voxel to get a more accurate result.

6.40.3.6 _instanceID `long Infohazard.HyperNav.NavVolume._instanceID [private]`

(Serialized) The unique ID for this volume to identify it in pathfinding jobs and serialized data.

6.40.3.7 _maxAgentRadius `float Infohazard.HyperNav.NavVolume._maxAgentRadius = 1 [private]`

(Serialized) The maximum size of agents using this volume.

6.40.3.8 _maxExternalLinkDistance `float Infohazard.HyperNav.NavVolume._maxExternalLinkDistance = 1 [private]`

(Serialized) The maximum distance that external links can extend outside of this volume.

6.40.3.9 _showVertexNumbers `bool Infohazard.HyperNav.NavVolume._showVertexNumbers [private]`

(Serialized) Whether to show the vertex numbers of the preview mesh in the scene view (for debugging).

6.40.3.10 _showVertexNumbersRange `float Infohazard.HyperNav.NavVolume._showVertexNumbersRange = 2 [private]`

(Serialized) Max distance from the camera at which vertex numbers will be shown.

6.40.3.11 _startLocations `Vector3 [] Infohazard.HyperNav.NavVolume._startLocations [private]`

(Serialized) If _useStartLocations is true, which start locations to use.

6.40.3.12 _staticOnly `bool Infohazard.HyperNav.NavVolume._staticOnly = true [private]`

(Serialized) Whether only static objects should be included in the baked data.

6.40.3.13 _useMultithreading `bool Infohazard.HyperNav.NavVolume._useMultithreading = true [private]`

(Serialized) Whether to use multiple threads when baking the volume.

6.40.3.14 _useStartLocations `bool Infohazard.HyperNav.NavVolume._useStartLocations = false [private]`

(Serialized) Whether only regions connected to certain locations are considered valid.

6.40.3.15 _visualizationMode `NavVolumeVisualizationMode Infohazard.HyperNav.NavVolume._visualizationMode = NavVolumeVisualizationMode.Final [private]`

(Serialized) Stage at which to visualize the volume bake process in the scene view.

6.40.3.16 _visualizeNeighbors `bool Infohazard.HyperNav.NavVolume._visualizeNeighbors [private]`

(Serialized) Whether to show the connections of a selected region in the scene view.

6.40.3.17 _visualizeNeighborsRegion `int Infohazard.HyperNav.NavVolume._visualizeNeighborsRegion [private]`

(Serialized) If _visualizeNeighbors is true, which region to visualize in the scene view.

6.40.3.18 _visualizeVoxelQueries `bool Infohazard.HyperNav.NavVolume._visualizeVoxelQueries [private]`

(Serialized) Whether to visualize the queries that are performed for a voxel when baking.

6.40.3.19 _voxelSize `float Infohazard.HyperNav.NavVolume._voxelSize = 1 [private]`

(Serialized) The voxel size of this volume, which determines the precision but also baking cost.

6.40.3.20 VolumeData `NativeParallelHashMap<long, NativeNavVolumeData> Infohazard.HyperNav.↩ NavVolume.VolumeData [static]`

Data for all loaded volumes in the format used by jobs.

6.40.4 Property Documentation

6.40.4.1 AutoDetectMovement `bool Infohazard.HyperNav.NavVolume.AutoDetectMovement [get], [set]`

Whether to automatically update native data if the volume moves.

Note that if this is true and the volume moves every frame, pathfinding will never be able to occur.

6.40.4.2 BlockingLayers `LayerMask Infohazard.HyperNav.NavVolume.BlockingLayers [get], [set]`

Which layers are considered impassible for pathfinding.

6.40.4.3 Bounds `Bounds Infohazard.HyperNav.NavVolume.Bounds [get], [set]`

The boundaries of the volume.

This cannot be set while the game is running.

6.40.4.4 Data `NavVolumeData Infohazard.HyperNav.NavVolume.Data [get], [set]`

The baked data for the volume.

This cannot be set while the game is running.

6.40.4.5 EnableMultiQuery `bool Infohazard.HyperNav.NavVolume.EnableMultiQuery [get], [set]`

Whether to enable multiple physics queries per voxel to get a more accurate result.

6.40.4.6 InstanceID `long Infohazard.HyperNav.NavVolume.InstanceID [get]`

The unique ID for this volume to identify it in pathfinding jobs and serialized data.

6.40.4.7 MaxAgentRadius `float Infohazard.HyperNav.NavVolume.MaxAgentRadius [get], [set]`

The maximum size of agents using this volume.

6.40.4.8 MaxExternalLinkDistance `float Infohazard.HyperNav.NavVolume.MaxExternalLinkDistance [get], [set]`

The maximum distance that external links can extend outside of this volume.

6.40.4.9 ShowVertexNumbers `bool Infohazard.HyperNav.NavVolume.ShowVertexNumbers [get], [set]`

Whether to show the vertex numbers of the preview mesh in the scene view (for debugging).

6.40.4.10 ShowVertexNumbersRange `float Infohazard.HyperNav.NavVolume.ShowVertexNumbersRange [get], [set]`

Max distance from the camera at which vertex numbers will be shown.

6.40.4.11 StartLocations `IReadOnlyList<Vector3> Infohazard.HyperNav.NavVolume.StartLocations [get], [set]`

If _useStartLocations is true, which start locations to use.

6.40.4.12 StaticOnly `bool Infohazard.HyperNav.NavVolume.StaticOnly [get], [set]`

Whether only static objects should be included in the baked data.

6.40.4.13 UseMultithreading `bool Infohazard.HyperNav.NavVolume.UseMultithreading [get], [set]`

Whether to use multiple threads when baking the volume.

This should only be turned off for debugging.

6.40.4.14 UseStartLocations `bool Infohazard.HyperNav.NavVolume.UseStartLocations [get], [set]`

Whether only regions connected to certain locations are considered valid.

This can be used to exclude certain regions from a volume, such as regions that are outside reachable area.

6.40.4.15 VisualizationMode `NavVolumeVisualizationMode Infohazard.HyperNav.NavVolume.VisualizationMode [get], [set]`

Stage at which to visualize the volume bake process in the scene view.

6.40.4.16 VisualizeNeighbors `bool Infohazard.HyperNav.NavVolume.VisualizeNeighbors [get], [set]`

Whether to show the connections of a selected region in the scene view.

6.40.4.17 VisualizeNeighborsRegion `int Infohazard.HyperNav.NavVolume.VisualizeNeighborsRegion [get], [set]`

If _visualizeNeighbors is true, which region to visualize in the scene view.

6.40.4.18 VisualizeVoxelQueries `bool Infohazard.HyperNav.NavVolume.VisualizeVoxelQueries [get], [set]`

Whether to visualize the queries that are performed for a voxel when baking.

6.40.4.19 VolumeChangingCount `int Infohazard.HyperNav.NavVolume.VolumeChangingCount [static], [get], [private set]`

Number of places that are modifying volume data.

6.40.4.20 Volumes `IReadOnlyDictionary<long, NavVolume> Infohazard.HyperNav.NavVolume.Volumes [static], [get]`

All currently loaded volumes.

6.40.4.21 VoxelSize `float Infohazard.HyperNav.NavVolume.VoxelSize [get], [set]`

The voxel size of this volume, which determines the precision but also baking cost.

6.40.5 Event Documentation

6.40.5.1 VolumeDataChanged `Action Infohazard.HyperNav.NavVolume.VolumeDataChanged [static]`

Event that is invoked immediately after active volume data changes.

6.40.5.2 VolumeDataChanging `Action Infohazard.HyperNav.NavVolume.VolumeDataChanging [static]`

Event that is invoked immediately before active volume data changes.

The documentation for this class was generated from the following file:

- Runtime/NavVolume.cs

6.41 Infohazard.HyperNav.Editor.NavVolumeBakeProgress Struct Reference

Represents current bake state of a volume, including progress fraction and current operation display name.

6.41.1 Detailed Description

Represents current bake state of a volume, including progress fraction and current operation display name.

The documentation for this struct was generated from the following file:

- Editor/NavVolumeBakingUtil.cs

6.42 Infohazard.HyperNav.Editor.NavVolumeBakingUtil Class Reference

Contains all the code needed to generate the data for a NavVolume.

Static Public Member Functions

- static void GetOrCreateData (NavVolume volume)

  Get the NavVolumeData for a given volume, or create and save the object if it doesn't exist yet.
- static void BakeData (NavVolume volume)

  Bake the NavVolumeData for a given volume.
- static void CancelBake (NavVolume volume)

  Cancel an actively baking volume and clear out its data.
- static void ClearData (NavVolume volume)

  Clear out the baked data of a volume (does not destroy or un-assign the actual data object).
- static void BuildTriangulationPreviewMesh (NavVolume volume, IReadOnlyList< Vector3 > vertices, IReadOnlyList< IReadOnlyList< int > > triLists)

  Build a preview mesh for the given region based on the given list of vertices and lists of triangles.

Static Public Attributes

- static readonly Dictionary< NavVolume, NavVolumeBakeProgress > BakeProgress

  The bake progress for each volume currently being baked.
- static readonly Dictionary< NavVolume, EditorCoroutine > BakeCoroutines

  The coroutine for each volume currently being baked.

Events

- static Action< NavVolume > BakeProgressUpdated

  Invoked when the bake progress for a NavVolume changes.

6.42.1 Detailed Description

Contains all the code needed to generate the data for a NavVolume.

6.42.2 Member Function Documentation

6.42.2.1 BakeData()

```
static void Infohazard.HyperNav.Editor.NavVolumeBakingUtil.BakeData (
    NavVolume volume ) [static]
```

Bake the NavVolumeData for a given volume.

Parameters

volume	The volume to bake.

6.42.2.2 BuildTriangulationPreviewMesh()

```
static void Infohazard.HyperNav.Editor.NavVolumeBakingUtil.BuildTriangulationPreviewMesh (
            NavVolume volume,
            IReadOnlyList< Vector3 > vertices,
            IReadOnlyList< IReadOnlyList< int > > triLists )  [static]
```

Build a preview mesh for the given region based on the given list of vertices and lists of triangles.

Parameters

volume	The volume to build for.
vertices	The list of all vertices.
triLists	The list of all triangle lists.

6.42.2.3 CancelBake() `static void Infohazard.HyperNav.Editor.NavVolumeBakingUtil.CancelBake ( NavVolume volume )  [static]`

Cancel an actively baking volume and clear out its data.

Parameters

volume	The volume being baked.

6.42.2.4 ClearData() `static void Infohazard.HyperNav.Editor.NavVolumeBakingUtil.ClearData ( NavVolume volume )  [static]`

Clear out the baked data of a volume (does not destroy or un-assign the actual data object).

Parameters

volume	

6.42.2.5 GetOrCreateData() `static void Infohazard.HyperNav.Editor.NavVolumeBakingUtil.GetOr↵CreateData ( NavVolume volume )  [static]`

Get the NavVolumeData for a given volume, or create and save the object if it doesn't exist yet.

Parameters

volume	The NavVolume component.

6.42.3 Member Data Documentation

6.42.3.1 BakeCoroutines `readonly Dictionary<NavVolume, EditorCoroutine> Infohazard.HyperNav.Editor.NavVolumeBakingUtil.BakeCoroutines [static]`

Initial value:

```
=
          new Dictionary<NavVolume, EditorCoroutine>()
```

The coroutine for each volume currently being baked.

6.42.3.2 BakeProgress `readonly Dictionary<NavVolume, NavVolumeBakeProgress> Infohazard.HyperNav.Editor.NavVolumeBakingUtil.BakeProgress [static]`

Initial value:

```
=
          new Dictionary<NavVolume, NavVolumeBakeProgress>()
```

The bake progress for each volume currently being baked.

6.42.4 Event Documentation

6.42.4.1 BakeProgressUpdated `Action<NavVolume> Infohazard.HyperNav.Editor.NavVolumeBakingUtil.BakeProgressUpdated [static]`

Invoked when the bake progress for a NavVolume changes.

The documentation for this class was generated from the following file:

- Editor/NavVolumeBakingUtil.cs

6.43 Infohazard.HyperNav.NavVolumeData Class Reference

The baked data of a NavVolume, saved as an asset.

Public Member Functions

- void Populate (Vector3[ ] vertices, NavRegionData[ ] regions, int[ ] blockingTriangleIndices)

  Populate the properties of this NavVolumeData.
- void MarkExternalLinksLocalSpace ()

  After updating external links, mark them as being in local space.
- void Clear ()

  Clear the properties of this NavVolumeData.
- void ToInternalData (NavVolume volume, out NativeNavVolumeData data, out NavDataInternalPointers pointers)

  Convert this NavVolumeData to the native format so that it can be used by jobs.

Properties

- IReadOnlyList< Vector3 > Vertices `[get]`

  The vertex positions of all of the volume's regions, in local space.
- IReadOnlyList< NavRegionData > Regions `[get]`

  The regions of the volume.
- IReadOnlyList< int > BlockingTriangleIndices `[get]`

  The vertex indices of triangles that define impassible space in the volume.
- bool ExternalLinksAreLocalSpace `[get]`

  Whether the external links are in local space (false = world space).

Private Attributes

- Vector3[ ] _vertices

  (Serialized) The vertex positions of all of the volume's regions, in local space.
- NavRegionData[ ] _regions

  (Serialized) The regions of the volume.
- int[ ] _blockingTriangleIndices

  (Serialized) The vertex indices of triangles that define impassible space in the volume.
- bool _externalLinksAreLocalSpace

  (Serialized) Whether the external links are in local space (false = world space).

6.43.1 Detailed Description

The baked data of a NavVolume, saved as an asset.

6.43.2 Member Function Documentation

6.43.2.1 Clear() `void Infohazard.HyperNav.NavVolumeData.Clear ( )`

Clear the properties of this NavVolumeData.

6.43.2.2 MarkExternalLinksLocalSpace() `void Infohazard.HyperNav.NavVolumeData.MarkExternal↩ LinksLocalSpace ( )`

After updating external links, mark them as being in local space.

6.43.2.3 Populate()

```
void Infohazard.HyperNav.NavVolumeData.Populate (
        Vector3[ ] vertices,
        NavRegionData[ ] regions,
        int[ ] blockingTriangleIndices )
```

Populate the properties of this NavVolumeData.

Parameters

vertices	Vertex positions of the volume's regions.
regions	Regions of the volume.
blockingTriangleIndices	Indices of triangles that define impassible space.

6.43.2.4 ToInternalData()

```
void Infohazard.HyperNav.NavVolumeData.ToInternalData (
        NavVolume volume,
        out NativeNavVolumeData data,
        out NavDataInternalPointers pointers )
```

Convert this NavVolumeData to the native format so that it can be used by jobs.

Parameters

volume	Volume that owns this data.
data	Created native data.
pointers	Created data structure references (must be kept in order to deallocate).

6.43.3 Member Data Documentation

6.43.3.1 _blockingTriangleIndices

```
int [] Infohazard.HyperNav.NavVolumeData._blockingTriangleIndices [private]
```

(Serialized) The vertex indices of triangles that define impassible space in the volume.

6.43.3.2 _externalLinksAreLocalSpace

```
bool Infohazard.HyperNav.NavVolumeData._externalLinksAreLocalSpace [private]
```

(Serialized) Whether the external links are in local space (false = world space).

6.43.3.3 _regions

```
NavRegionData [] Infohazard.HyperNav.NavVolumeData._regions [private]
```

(Serialized) The regions of the volume.

6.43.3.4 _vertices

```
Vector3 [] Infohazard.HyperNav.NavVolumeData._vertices [private]
```

(Serialized) The vertex positions of all of the volume's regions, in local space.

6.43.4 Property Documentation

6.43.4.1 BlockingTriangleIndices `IReadOnlyList<int> Infohazard.HyperNav.NavVolumeData.BlockingTriangleIndices [get]`

The vertex indices of triangles that define impassible space in the volume.

6.43.4.2 ExternalLinksAreLocalSpace `bool Infohazard.HyperNav.NavVolumeData.ExternalLinksAreLocalSpace [get]`

Whether the external links are in local space (false = world space).

6.43.4.3 Regions `IReadOnlyList<NavRegionData> Infohazard.HyperNav.NavVolumeData.Regions [get]`

The regions of the volume.

6.43.4.4 Vertices `IReadOnlyList<Vector3> Infohazard.HyperNav.NavVolumeData.Vertices [get]`

The vertex positions of all of the volume's regions, in local space.

The documentation for this class was generated from the following file:

- Runtime/NavVolumeData.cs

6.44 Infohazard.HyperNav.Editor.NavVolumeEditor Class Reference

Custom editor for Infohazard.HyperNav.NavVolume.

6.44.1 Detailed Description

Custom editor for Infohazard.HyperNav.NavVolume.

The documentation for this class was generated from the following file:

- Editor/NavVolumeEditor.cs

6.45 Infohazard.HyperNav.Editor.NavVolumeExternalLinkUtil Class Reference

Utilities for generating the external links of NavVolumes.

Static Public Member Functions

- static void GenerateAllExternalLinks ()

  Generate the external links for all loaded NavVolumes.
- static void GenerateExternalLinks (NavVolume volume)

  Generate the external links for a specific NavVolume.

6.45.1 Detailed Description

Utilities for generating the external links of NavVolumes.

6.45.2 Member Function Documentation

6.45.2.1 GenerateAllExternalLinks() `static void Infohazard.HyperNav.Editor.NavVolumeExternal←LinkUtil.GenerateAllExternalLinks ( ) [static]`

Generate the external links for all loaded NavVolumes.

6.45.2.2 GenerateExternalLinks() `static void Infohazard.HyperNav.Editor.NavVolumeExternalLink←Util.GenerateExternalLinks ( NavVolume volume ) [static]`

Generate the external links for a specific NavVolume.

Parameters

volume	

The documentation for this class was generated from the following file:

- Editor/NavVolumeExternalLinkUtil.cs

6.46 Infohazard.HyperNav.NavWaypoint Struct Reference

A waypoint in a completed path.

Properties

- NavWaypointType Type `[get, set]`

  The type of waypoint in relation to its containing volume.
- Vector3 Position `[get, set]`

  The position of the waypoint in world space.
- long VolumeID `[get, set]`

  Identifier of the NavVolume that contains this waypoint, or -1.

6.46.1 Detailed Description

A waypoint in a completed path.

6.46.2 Property Documentation

6.46.2.1 Position `Vector3 Infohazard.HyperNav.NavWaypoint.Position [get], [set]`

The position of the waypoint in world space.

6.46.2.2 Type `NavWaypointType Infohazard.HyperNav.NavWaypoint.Type [get], [set]`

The type of waypoint in relation to its containing volume.

6.46.2.3 VolumeID `long Infohazard.HyperNav.NavWaypoint.VolumeID [get], [set]`

Identifier of the NavVolume that contains this waypoint, or -1.

The documentation for this struct was generated from the following file:

- Runtime/NavPathfinder.cs

6.47 Infohazard.HyperNav.Jobs.PendingPathNode Struct Reference

A discovered node in a pending path, which serves as a key into the dictionary of per-node discovered info.

Public Member Functions

- bool Equals (PendingPathNode other)

  Compare to another PendingPathNode.
- override bool Equals (object obj)

  Compare to another object.
- override int GetHashCode ()

  Get integer for use with hash table.

Public Attributes

- int FromRegion

  The region from which this node originates.
- int ToRegion

  The region to which this node leads.
- long FromVolume

  The volume from which this node originates.
- long ToVolume

  The volume to which this node leads.
- float4 Position

  The position of this node.
- bool IsExternalConnection

  Whether this node is an external connection (bridges two different volumes).
- int ConnectionIndex

  Which connection in the originating region's connections array this node represents.

6.47.1 Detailed Description

A discovered node in a pending path, which serves as a key into the dictionary of per-node discovered info.

Each PendingPathNode represents a transition between two regions.

6.47.2 Member Function Documentation

6.47.2.1 Equals() `[1/2]` `override bool Infohazard.HyperNav.Jobs.PendingPathNode.Equals ( object obj )`

Compare to another object.

Parameters

obj	Object to compare to.

Returns

Whether the two objects are equal.

6.47.2.2 Equals() `[2/2]` `bool Infohazard.HyperNav.Jobs.PendingPathNode.Equals ( PendingPathNode other )`

Compare to another PendingPathNode.

Parameters

other	Node to compare to.

Returns

Whether the two nodes are equal.

6.47.2.3 GetHashCode() `override int Infohazard.HyperNav.Jobs.PendingPathNode.GetHashCode ( )`

Get integer for use with hash table.

Returns

Integer hash code.

6.47.3 Member Data Documentation

6.47.3.1 ConnectionIndex `int Infohazard.HyperNav.Jobs.PendingPathNode.ConnectionIndex`

Which connection in the originating region's connections array this node represents.

6.47.3.2 FromRegion `int Infohazard.HyperNav.Jobs.PendingPathNode.FromRegion`

The region from which this node originates.

6.47.3.3 FromVolume `long Infohazard.HyperNav.Jobs.PendingPathNode.FromVolume`

The volume from which this node originates.

6.47.3.4 IsExternalConnection `bool Infohazard.HyperNav.Jobs.PendingPathNode.IsExternalConnection`

Whether this node is an external connection (bridges two different volumes).

6.47.3.5 Position `float4 Infohazard.HyperNav.Jobs.PendingPathNode.Position`

The position of this node.

6.47.3.6 ToRegion `int Infohazard.HyperNav.Jobs.PendingPathNode.ToRegion`

The region to which this node leads.

6.47.3.7 ToVolume `long Infohazard.HyperNav.Jobs.PendingPathNode.ToVolume`

The volume to which this node leads.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavPathJob.cs

6.48 Infohazard.HyperNav.NavAgent.PropNames Class Reference

This is used to refer to the names of private fields in this class from a custom Editor.

6.48.1 Detailed Description

This is used to refer to the names of private fields in this class from a custom Editor.

The documentation for this class was generated from the following file:

- Runtime/NavAgent.cs

6.49 Infohazard.HyperNav.NavPathfinder.PropNames Class Reference

This is used to refer to the names of private fields in this class from a custom Editor.

6.49.1 Detailed Description

This is used to refer to the names of private fields in this class from a custom Editor.

The documentation for this class was generated from the following file:

- Runtime/NavPathfinder.cs

6.50 Infohazard.HyperNav.NavVolume.PropNames Class Reference

This is used to refer to the names of private fields in this class from a custom Editor.

6.50.1 Detailed Description

This is used to refer to the names of private fields in this class from a custom Editor.

The documentation for this class was generated from the following file:

- Runtime/NavVolume.cs

6.51 Infohazard.HyperNav.RigidbodyAvoidanceObstacle Class Reference

An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity from a Rigidbody.

Public Member Functions

- virtual void Reset ()

  Set _rigidbody.

Properties

- Rigidbody Rigidbody `[get, set]`

  Rigidbody to get the velocity from.
- override Vector3 InputVelocity `[get]`

  The object's desired (or actual) velocity.

Private Attributes

- Rigidbody _rigidbody

  (Serialized) Rigidbody to get the velocity from.

Additional Inherited Members

6.51.1 Detailed Description

An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity from a Rigidbody.

6.51.2 Member Function Documentation

6.51.2.1 Reset() `virtual void Infohazard.HyperNav.RigidbodyAvoidanceObstacle.Reset ( ) [virtual]`

Set _rigidbody.

6.51.3 Member Data Documentation

6.51.3.1 _rigidbody `Rigidbody Infohazard.HyperNav.RigidbodyAvoidanceObstacle._rigidbody [private]`

(Serialized) Rigidbody to get the velocity from.

6.51.4 Property Documentation

6.51.4.1 InputVelocity `override Vector3 Infohazard.HyperNav.RigidbodyAvoidanceObstacle.InputVelocity [get]`

The object's desired (or actual) velocity.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

6.51.4.2 Rigidbody `Rigidbody Infohazard.HyperNav.RigidbodyAvoidanceObstacle.Rigidbody [get], [set]`

Rigidbody to get the velocity from.

The documentation for this class was generated from the following file:

- Runtime/Avoidance/RigidbodyAvoidanceObstacle.cs

6.52 Infohazard.HyperNav.SimpleAvoidanceObstacle Class Reference

An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity by measuring its position/time delta.

Protected Member Functions

- override void OnEnable ()

  Resets desired velocity and adds self to list of all obstacles.
- virtual void LateUpdate ()

  Update computed velocity.

Properties

- override Vector3 InputVelocity `[get]`
  The object's desired (or actual) velocity.

6.52.1 Detailed Description

An IAvoidanceObstacle that gets its IAvoidanceObstacle.InputVelocity by measuring its position/time delta.

6.52.2 Member Function Documentation

6.52.2.1 LateUpdate() `virtual void Infohazard.HyperNav.SimpleAvoidanceObstacle.LateUpdate ( ) [protected], [virtual]`

Update computed velocity.

6.52.2.2 OnEnable() `override void Infohazard.HyperNav.SimpleAvoidanceObstacle.OnEnable ( ) [protected], [virtual]`

Resets desired velocity and adds self to list of all obstacles.

Reimplemented from Infohazard.HyperNav.AvoidanceObstacleBase.

6.52.3 Property Documentation

6.52.3.1 InputVelocity `override Vector3 Infohazard.HyperNav.SimpleAvoidanceObstacle.Input↩ Velocity [get]`

The object's desired (or actual) velocity.

Implements Infohazard.HyperNav.IAvoidanceObstacle.

The documentation for this class was generated from the following file:

- Runtime/Avoidance/SimpleAvoidanceObstacle.cs

6.53 Infohazard.HyperNav.SplineNavAgent Class Reference

A script that can be used to calculate smooth paths by any entity that needs to use HyperNav for navigation.

Public Member Functions

- override void Stop (bool abortPaths)
  Stop following the current path, and optionally cancel all path requests.

Parameters

abortPaths	*Whether to cancel pending path requests.*

Protected Member Functions

- override void Update ()

  Updates measured velocity and finds the nearest point on the spline.
- override void OnDrawGizmos ()

  Draws the current spline if NavAgent.DebugPath is true.
- override Vector3 CalculateDesiredNavigationVelocity ()

  Calculate the velocity the agent wants to move in, in the range [0, 1].
- override void OnPathReady (long id, NavPath path)

  Callback that is received when a pathfinding request completes, which should start moving along that path.

Parameters

id	*The id of the path request.*
path	*The completed path, which is null if no path was found.*

- override void OnDisable ()

  Stops all pathfinding and cancels path requests.

Properties

- float TangentScale `[get, set]`

  Scale to apply to spline tangents (lower values make the spline less curvy).
- bool RaycastTangents `[get, set]`

  Whether to shorten tangents by raycasting to ensure they don't penetrate blocked areas.
- int DistanceSamplesPerSegment `[get, set]`

  How many samples to take per segment of the spline when mapping the distance.
- int DebugPointCount `[get, set]`

  If NavAgent.DebugPath is enabled, how many points to use to draw the curve.
- float MaxAlignmentVelocityDistance `[get, set]`

  At what distance from the spline the agent will have all its desired velocity devoted to returning.
- float CurvatureSampleDistance `[get, set]`

  Delta-T to use when sampling curvature (should be quite small).
- float CurvatureOfMaxSlowdown `[get, set]`

  At what curvature value is the agent at its max curvature slowdown.
- float MaxCurvatureSlowdown `[get, set]`

  The multiplier on desired tangent velocity when at the max curvature value.
- bool DebugProjectOnSpline `[get, set]`

  Whether to draw debug lines when projecting on the spline.
- float BlockedDetectionDistance `[get, set]`

  Distance in front of the agent to check to see if it needs to avoid level geometry.
- float BlockedDetectionBackDistance `[get, set]`

  Distance behind the agent to check to see if it needs to avoid level geometry.
- float BlockedDetectionMinSplineDistance `[get, set]`

  How far the agent must be from the spline to check for blocking level geometry.

- SplinePath SplinePath `[get]`

  The spline that the agent is currently following.
- float CurrentSplineParameter `[get]`

  The spline parameter value the agent is nearest to on the spline.
- float CurrentSplineDistance `[get]`

  The distance along the spline the agent is nearest to.
- float MaxSplineDistance `[get]`

  The length of the agent's current spline path.

Private Attributes

- float _tangentScale = 0.5f

  (Serialized) Scale to apply to spline tangents (lower values make the spline less curvy).
- bool _raycastTangents = true

  (Serialized) Whether to shorten tangents by raycasting to ensure they don't penetrate blocked areas.
- int _distanceSamplesPerSegment = 5

  (Serialized) How many samples to take per segment of the spline when mapping the distance.
- int _debugPointCount = 50

  (Serialized) If NavAgent.DebugPath is enabled, how many points to use to draw the curve.
- float _maxAlignmentVelocityDistance = 1.5f

  (Serialized) At what distance from the spline the agent will have all its desired velocity devoted to returning.
- float _curvatureSampleDistance = 0.01f

  (Serialized) Delta-T to use when sampling curvature (should be quite small).
- float _curvatureOfMaxSlowdown = 0.5f

  (Serialized) At what curvature value is the agent at its max curvature slowdown.
- float _maxCurvatureSlowdown = 0.5f

  (Serialized) The multiplier on desired tangent velocity when at the max curvature value.
- bool _debugProjectOnSpline = false

  (Serialized) Whether to draw debug lines when projecting on the spline.
- float _blockedDetectionDistance = 0.1f

  (Serialized) Distance in front of the agent to check to see if it needs to avoid level geometry.
- float _blockedDetectionBackDistance = 0.1f

  (Serialized) Distance behind the agent to check to see if it needs to avoid level geometry.
- float _blockedDetectionMinSplineDistance = 0.2f

  (Serialized) How far the agent must be from the spline to check for blocking level geometry.

Additional Inherited Members

6.53.1 Detailed Description

A script that can be used to calculate smooth paths by any entity that needs to use HyperNav for navigation.

A SplineNavAgent works just like (and is a subclass of) NavAgent. However, a SplineNavAgent feeds the path waypoints into a spline function to get a smoother path. In order to follow the spline, the SplineNavAgent creates its NavAgent.DesiredVelocity based on two factors:

- Tangent: The direction the spline is pointing nearest the agent.
- Alignment: The direction from the agent to the nearest point on the spline.

The agent will increase the influence of the alignment velocity the further it gets from the spline, in order to prevent drifting too far off. Additionally, the agent can slow down its desired tangent velocity on high-curvature regions, in order to take the curves more slowly.

6.53.2 Member Function Documentation

6.53.2.1 CalculateDesiredNavigationVelocity() `override Vector3 Infohazard.HyperNav.SplineNav↩ Agent.CalculateDesiredNavigationVelocity ( ) [protected], [virtual]`

Calculate the velocity the agent wants to move in, in the range [0, 1].

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.2.2 OnDisable() `override void Infohazard.HyperNav.SplineNavAgent.OnDisable ( ) [protected], [virtual]`

Stops all pathfinding and cancels path requests.

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.2.3 OnDrawGizmos() `override void Infohazard.HyperNav.SplineNavAgent.OnDrawGizmos ( ) [protected], [virtual]`

Draws the current spline if NavAgent.DebugPath is true.

The spline will be drawn with DebugPointCount points.

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.2.4 OnPathReady() `override void Infohazard.HyperNav.SplineNavAgent.OnPathReady (`

```
        long id,
        NavPath path ) [protected], [virtual]
```

Callback that is received when a pathfinding request completes, which should start moving along that path.

Parameters

id	The id of the path request.
path	The completed path, which is null if no path was found.

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.2.5 Stop() `override void Infohazard.HyperNav.SplineNavAgent.Stop (`

```
        bool abortPaths ) [virtual]
```

Stop following the current path, and optionally cancel all path requests.

Parameters

abortPaths	Whether to cancel pending path requests.

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.2.6 Update() `override void Infohazard.HyperNav.SplineNavAgent.Update ( ) [protected], [virtual]`

Updates measured velocity and finds the nearest point on the spline.

Reimplemented from Infohazard.HyperNav.NavAgent.

6.53.3 Member Data Documentation

6.53.3.1 _blockedDetectionBackDistance `float Infohazard.HyperNav.SplineNavAgent._blockedDetectionBackDistance = 0.1f [private]`

(Serialized) Distance behind the agent to check to see if it needs to avoid level geometry.

6.53.3.2 _blockedDetectionDistance `float Infohazard.HyperNav.SplineNavAgent._blockedDetectionDistance = 0.1f [private]`

(Serialized) Distance in front of the agent to check to see if it needs to avoid level geometry.

6.53.3.3 _blockedDetectionMinSplineDistance `float Infohazard.HyperNav.SplineNavAgent._blockedDetectionMinSplineDistance = 0.2f [private]`

(Serialized) How far the agent must be from the spline to check for blocking level geometry.

6.53.3.4 _curvatureOfMaxSlowdown `float Infohazard.HyperNav.SplineNavAgent._curvatureOfMaxSlowdown = 0.5f [private]`

(Serialized) At what curvature value is the agent at its max curvature slowdown.

6.53.3.5 _curvatureSampleDistance

`float Infohazard.HyperNav.SplineNavAgent._curvatureSample↩ Distance = 0.01f [private]`

(Serialized) Delta-T to use when sampling curvature (should be quite small).

6.53.3.6 _debugPointCount

`int Infohazard.HyperNav.SplineNavAgent._debugPointCount = 50 [private]`

(Serialized) If NavAgent.DebugPath is enabled, how many points to use to draw the curve.

6.53.3.7 _debugProjectOnSpline

`bool Infohazard.HyperNav.SplineNavAgent._debugProjectOnSpline = false [private]`

(Serialized) Whether to draw debug lines when projecting on the spline.

6.53.3.8 _distanceSamplesPerSegment

`int Infohazard.HyperNav.SplineNavAgent._distanceSamples↩ PerSegment = 5 [private]`

(Serialized) How many samples to take per segment of the spline when mapping the distance.

6.53.3.9 _maxAlignmentVelocityDistance

`float Infohazard.HyperNav.SplineNavAgent._maxAlignment↩ VelocityDistance = 1.5f [private]`

(Serialized) At what distance from the spline the agent will have all its desired velocity devoted to returning.

6.53.3.10 _maxCurvatureSlowdown

`float Infohazard.HyperNav.SplineNavAgent._maxCurvature↩ Slowdown = 0.5f [private]`

(Serialized) The multiplier on desired tangent velocity when at the max curvature value.

6.53.3.11 _raycastTangents

`bool Infohazard.HyperNav.SplineNavAgent._raycastTangents = true [private]`

(Serialized) Whether to shorten tangents by raycasting to ensure they don't penetrate blocked areas.

6.53.3.12 _tangentScale `float Infohazard.HyperNav.SplineNavAgent._tangentScale = 0.5f [private]`

(Serialized) Scale to apply to spline tangents (lower values make the spline less curvy).

6.53.4 Property Documentation

6.53.4.1 BlockedDetectionBackDistance `float Infohazard.HyperNav.SplineNavAgent.BlockedDetectionBackDistance [get], [set]`

Distance behind the agent to check to see if it needs to avoid level geometry.

6.53.4.2 BlockedDetectionDistance `float Infohazard.HyperNav.SplineNavAgent.BlockedDetectionDistance [get], [set]`

Distance in front of the agent to check to see if it needs to avoid level geometry.

6.53.4.3 BlockedDetectionMinSplineDistance `float Infohazard.HyperNav.SplineNavAgent.BlockedDetectionMinSplineDistance [get], [set]`

How far the agent must be from the spline to check for blocking level geometry.

6.53.4.4 CurrentSplineDistance `float Infohazard.HyperNav.SplineNavAgent.CurrentSplineDistance [get]`

The distance along the spline the agent is nearest to.

6.53.4.5 CurrentSplineParameter `float Infohazard.HyperNav.SplineNavAgent.CurrentSplineParameter [get]`

The spline parameter value the agent is nearest to on the spline.

6.53.4.6 CurvatureOfMaxSlowdown `float Infohazard.HyperNav.SplineNavAgent.CurvatureOfMaxSlowdown [get], [set]`

At what curvature value is the agent at its max curvature slowdown.

6.53.4.7 CurvatureSampleDistance `float Infohazard.HyperNav.SplineNavAgent.CurvatureSample← Distance [get], [set]`

Delta-T to use when sampling curvature (should be quite small).

6.53.4.8 DebugPointCount `int Infohazard.HyperNav.SplineNavAgent.DebugPointCount [get], [set]`

If NavAgent.DebugPath is enabled, how many points to use to draw the curve.

6.53.4.9 DebugProjectOnSpline `bool Infohazard.HyperNav.SplineNavAgent.DebugProjectOnSpline [get], [set]`

Whether to draw debug lines when projecting on the spline.

6.53.4.10 DistanceSamplesPerSegment `int Infohazard.HyperNav.SplineNavAgent.DistanceSamples← PerSegment [get], [set]`

How many samples to take per segment of the spline when mapping the distance.

6.53.4.11 MaxAlignmentVelocityDistance `float Infohazard.HyperNav.SplineNavAgent.MaxAlignment← VelocityDistance [get], [set]`

At what distance from the spline the agent will have all its desired velocity devoted to returning.

6.53.4.12 MaxCurvatureSlowdown `float Infohazard.HyperNav.SplineNavAgent.MaxCurvatureSlowdown [get], [set]`

The multiplier on desired tangent velocity when at the max curvature value.

6.53.4.13 MaxSplineDistance `float Infohazard.HyperNav.SplineNavAgent.MaxSplineDistance [get]`

The length of the agent's current spline path.

6.53.4.14 RaycastTangents `bool Infohazard.HyperNav.SplineNavAgent.RaycastTangents [get], [set]`

Whether to shorten tangents by raycasting to ensure they don't penetrate blocked areas.

6.53.4.15 SplinePath `SplinePath Infohazard.HyperNav.SplineNavAgent.SplinePath [get]`

The spline that the agent is currently following.

6.53.4.16 TangentScale `float Infohazard.HyperNav.SplineNavAgent.TangentScale [get], [set]`

Scale to apply to spline tangents (lower values make the spline less curvy).

The documentation for this class was generated from the following file:

- Runtime/Spline/SplineNavAgent.cs

6.54 Infohazard.HyperNav.SplinePath Struct Reference

A spline specialized for path following, created with a NavPath.

Public Member Functions

- SplinePath (NavPath path, float tangentScale, int sampleCount, bool raycastTangents)

  Create a new SplinePath with the given path.
- void Initialize (NavPath path, float tangentScale, int sampleCount, bool raycastTangents)

  Re-initialize an existing SplinePath with the given path.
- void Dispose ()

  Dispose arrays allocated for this spline path.
- float GetDistance (float parameter)

  Get the distance along the spline for a given parameter value.
- float GetParameter (float distance)

  Get the parameter value for a given distance along the spline.
- Vector3 GetControlPosition (int index)

  Get the position of a given control point.
- Vector3 GetControlTangent (int index)

  Get the tangent of a given control point.
- NavVolume GetVolume (float parameter)

  Get the NavVolume that contains the given parameter value on the spline.
- Vector3 GetPosition (float parameter)

  Get the position at a given parameter value.
- Vector3 GetTangent (float parameter)

  Get the tangent at a given parameter value.
- Vector3 GetCurvature (float parameter, float offset=0.01f)

  Sample the curvature at a given parameter value.
- float ProjectPosition (Vector3 position, int iterations=5, bool debug=false)

  Approximate the parameter value of the position along the spline nearest to the given position.

Properties

- float Length `[get, private set]`

  Length of the spline in world units.
- int PointCount `[get, private set]`

  Number of control points on the spline.
- bool IsCreated `[get, private set]`

  Whether an actual spline has been constructed.
- NativeArray< SplinePoint > ControlPoints `[get]`

  List of all the control points of the spline.

6.54.1 Detailed Description

A spline specialized for path following, created with a NavPath.

Unlike most spline tools, the tangents in this spline are calculated automatically.

This spline implementation uses two coordinate spaces: parameter and distance. Distance ranges from zero to the length of the spline, and values are distributed (approximately) evenly. Parameter ranges from zero to one and is the actual value supplied to the spline function, but values are not distributed evenly.

6.54.2 Constructor & Destructor Documentation

6.54.2.1 SplinePath()

```
Infohazard.HyperNav.SplinePath.SplinePath (
        NavPath path,
        float tangentScale,
        int sampleCount,
        bool raycastTangents )
```

Create a new SplinePath with the given path.

Parameters

path	The input navigation path.
tangentScale	Scale to apply to spline tangents (lower values make the spline less curvy).
sampleCount	How many samples to take per segment of the spline when mapping the distance.
raycastTangents	Whether to shorten tangents by raycasting against NavVolume blocking triangles.

6.54.3 Member Function Documentation

6.54.3.1 Dispose()

```
void Infohazard.HyperNav.SplinePath.Dispose ( )
```

Dispose arrays allocated for this spline path.

6.54.3.2 GetControlPosition() `Vector3 Infohazard.HyperNav.SplinePath.GetControlPosition ( int index )`

Get the position of a given control point.

Parameters

index	Control point index.

Returns

Position of that control point, in world space.

6.54.3.3 GetControlTangent() `Vector3 Infohazard.HyperNav.SplinePath.GetControlTangent ( int index )`

Get the tangent of a given control point.

Parameters

index	Control point index.

Returns

Tangent of that control point, in world space.

6.54.3.4 GetCurvature() `Vector3 Infohazard.HyperNav.SplinePath.GetCurvature ( float parameter, float offset = 0.01f )`

Sample the curvature at a given parameter value.

Unlike GetPosition and GetTangent, this does not return an exact value.

Parameters

parameter	The parameter value in range [0, 1].
offset	Offset distance to sample derivative of tangent function.

Returns

The sampled curvature value (use magnitude to get scalar curvature).

6.54.3.5 GetDistance() `float Infohazard.HyperNav.SplinePath.GetDistance ( float parameter )`

Get the distance along the spline for a given parameter value.

Parameters

parameter	The parameter value in range [0, 1].

Returns

The distance value in range [0, Length].

6.54.3.6 GetParameter() `float Infohazard.HyperNav.SplinePath.GetParameter ( float distance )`

Get the parameter value for a given distance along the spline.

Parameters

distance	The distance value in range [0, Length].

Returns

The parameter value in range [0, 1].

6.54.3.7 GetPosition() `Vector3 Infohazard.HyperNav.SplinePath.GetPosition ( float parameter )`

Get the position at a given parameter value.

Parameters

parameter	The parameter value in range [0, 1].

Returns

Position along the spline, in world space.

6.54.3.8 GetTangent() `Vector3 Infohazard.HyperNav.SplinePath.GetTangent ( float parameter )`

Get the tangent at a given parameter value.

Parameters

parameter	The parameter value in range [0, 1].

Returns

Tangent at that position, in world space.

6.54.3.9 GetVolume()

```
NavVolume Infohazard.HyperNav.SplinePath.GetVolume (
        float parameter )
```

Get the NavVolume that contains the given parameter value on the spline.

Parameters

parameter	Input parameter value.

Returns

The containing NavVolume.

6.54.3.10 Initialize()

```
void Infohazard.HyperNav.SplinePath.Initialize (
        NavPath path,
        float tangentScale,
        int sampleCount,
        bool raycastTangents )
```

Re-initialize an existing SplinePath with the given path.

Parameters

path	The input navigation path.
tangentScale	Scale to apply to spline tangents (lower values make the spline less curvy).
sampleCount	How many samples to take per segment of the spline when mapping the distance.
raycastTangents	Whether to shorten tangents by raycasting against NavVolume blocking triangles.

6.54.3.11 ProjectPosition()

```
float Infohazard.HyperNav.SplinePath.ProjectPosition (
        Vector3 position,
        int iterations = 5,
        bool debug = false )
```

Approximate the parameter value of the position along the spline nearest to the given position.

This uses Newton's method. Increasing the iteration count increases both accuracy and cost.

Parameters

position	Position to project.
iterations	Number of Newton's method iterations.
debug	Whether to draw debug lines showing Newton's method iterations.

Returns

The approximate parameter along the spline in range [0, 1].

6.54.4 Property Documentation

6.54.4.1 ControlPoints `NativeArray<SplinePoint> Infohazard.HyperNav.SplinePath.ControlPoints [get]`

List of all the control points of the spline.

6.54.4.2 IsCreated `bool Infohazard.HyperNav.SplinePath.IsCreated [get], [private set]`

Whether an actual spline has been constructed.

6.54.4.3 Length `float Infohazard.HyperNav.SplinePath.Length [get], [private set]`

Length of the spline in world units.

6.54.4.4 PointCount `int Infohazard.HyperNav.SplinePath.PointCount [get], [private set]`

Number of control points on the spline.

The documentation for this struct was generated from the following file:

- Runtime/Spline/SplinePath.cs

6.55 Infohazard.HyperNav.SplinePoint Struct Reference

Represents a point on a spline and the segment that starts with it.

Public Attributes

- float4 Position

  Position of the control point.
- float4 Tangent

  Tangent of the control point.
- float4x4 PositionMatrix

  Matrix to multiply by a time vector along the segment to get a position.
- float4x4 TangentMatrix

  Matrix to multiply by a time vector along the segment to get a tangent.
- long FromVolume

  Volume that leads to the control point.
- long ToVolume

  Volume that the control point leads to.

6.55.1 Detailed Description

Represents a point on a spline and the segment that starts with it.

6.55.2 Member Data Documentation

6.55.2.1 FromVolume `long Infohazard.HyperNav.SplinePoint.FromVolume`

Volume that leads to the control point.

6.55.2.2 Position `float4 Infohazard.HyperNav.SplinePoint.Position`

Position of the control point.

6.55.2.3 PositionMatrix `float4x4 Infohazard.HyperNav.SplinePoint.PositionMatrix`

Matrix to multiply by a time vector along the segment to get a position.

6.55.2.4 Tangent `float4 Infohazard.HyperNav.SplinePoint.Tangent`

Tangent of the control point.

6.55.2.5 TangentMatrix `float4x4 Infohazard.HyperNav.SplinePoint.TangentMatrix`

Matrix to multiply by a time vector along the segment to get a tangent.

6.55.2.6 ToVolume `long Infohazard.HyperNav.SplinePoint.ToVolume`

Volume that the control point leads to.

The documentation for this struct was generated from the following file:

- Runtime/Spline/SplinePath.cs

6.56 Infohazard.HyperNav.Jobs.SplineProjectJob Struct Reference

Job used to find the parameter along a spline that is nearest to the given point.

Public Member Functions

- void Execute ()

    Execute the job.

Public Attributes

- SplinePath Spline

    Spline to query.
- float4 Position

    Position to find nearest parameter to.
- int Iterations

    How many iterations of Newton's Method to perform.
- bool DebugProjection

    Whether to draw debug lines showing each iteration of Newton's Method.
- NativeArray< float > OutPosition

    Where the calculated nearest parameter value is written.

6.56.1 Detailed Description

Job used to find the parameter along a spline that is nearest to the given point.

6.56.2 Member Function Documentation

6.56.2.1 Execute() `void Infohazard.HyperNav.Jobs.SplineProjectJob.Execute ( )`

Execute the job.

6.56.3 Member Data Documentation

6.56.3.1 DebugProjection `bool Infohazard.HyperNav.Jobs.SplineProjectJob.DebugProjection`

Whether to draw debug lines showing each iteration of Newton's Method.

6.56.3.2 Iterations `int Infohazard.HyperNav.Jobs.SplineProjectJob.Iterations`

How many iterations of Newton's Method to perform.

6.56.3.3 OutPosition `NativeArray<float> Infohazard.HyperNav.Jobs.SplineProjectJob.OutPosition`

Where the calculated nearest parameter value is written.

6.56.3.4 Position `float4 Infohazard.HyperNav.Jobs.SplineProjectJob.Position`

Position to find nearest parameter to.

6.56.3.5 Spline `SplinePath Infohazard.HyperNav.Jobs.SplineProjectJob.Spline`

Spline to query.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/SplineProjectJob.cs

6.57 Infohazard.HyperNav.Editor.ThreadSafeIncrementor Class Reference

A value that can be incremented, decremented, or added to in a thread safe way.

Public Member Functions

- ThreadSafeIncrementor (int value=0)

  Create a new ThreadSafeIncrementor with the given initial value.
- int Increment ()

  Add one to the value, then return the new value, as an atomic operation.
- int Decrement ()

  Subtract one from the value, then return the new value, as an atomic operation.
- int Add (int value)

  Add the given value to the value, then return the new value, as an atomic operation.

Properties

- int Value `[get, set]`

  Current value of the incrementor.

6.57.1 Detailed Description

A value that can be incremented, decremented, or added to in a thread safe way.

The operations are pre-increments (equivalent to ++i), meaning the return value is the value after the increment.

6.57.2 Constructor & Destructor Documentation

6.57.2.1 ThreadSafeIncrementor()

```
Infohazard.HyperNav.Editor.ThreadSafeIncrementor.ThreadSafe←
Incrementor (
             int value = 0 )
```

Create a new ThreadSafeIncrementor with the given initial value.

Parameters

value	The initial value.

6.57.3 Member Function Documentation

6.57.3.1 Add()

```
int Infohazard.HyperNav.Editor.ThreadSafeIncrementor.Add (
             int value )
```

Add the given value to the value, then return the new value, as an atomic operation.

Parameters

value	The value to add.

Returns

The new value with value added.

6.57.3.2 Decrement() `int Infohazard.HyperNav.Editor.ThreadSafeIncrementor.Decrement ( )`

Subtract one from the value, then return the new value, as an atomic operation.

Returns

The new value with one subtracted.

6.57.3.3 Increment() `int Infohazard.HyperNav.Editor.ThreadSafeIncrementor.Increment ( )`

Add one to the value, then return the new value, as an atomic operation.

Returns

The new value with one added.

6.57.4 Property Documentation

6.57.4.1 Value `int Infohazard.HyperNav.Editor.ThreadSafeIncrementor.Value [get], [set]`

Current value of the incrementor.

This can be used to set the value in a non-thread-safe manor.

The documentation for this class was generated from the following file:

- Editor/NavVolumeBakingUtil.cs

6.58 Infohazard.HyperNav.Triangle Struct Reference

Represents the indices of a triangle (three vertices by a face) in an indexed mesh.

Public Member Functions

- Triangle (int vertex1, int vertex2, int vertex3)

  Construct a new Triangle with the given indices.
- override bool Equals (object obj)

  Compare to another object.
- bool Equals (Triangle other)

  Compare to another Triangle.
- override int GetHashCode ()

  Get integer for use with hash table.

Properties

- int Vertex1 `[get]`

  First vertex index, which is the lower of the three.
- int Vertex2 `[get]`

  Second vertex index, which is the middle of the three.
- int Vertex3 `[get]`

  Third vertex index, which is the larger of the three.

Private Attributes

- int _minVertex

  (Serialized) First vertex index, which is the lower of the three.
- int _midVertex

  (Serialized) Second vertex index, which is the middle of the three.
- int _maxVertex

  (Serialized) Third vertex index, which is the larger of the three.

6.58.1 Detailed Description

Represents the indices of a triangle (three vertices by a face) in an indexed mesh.

The same Triangle will be created regardless of the order in which indices are fed to the constructor.

6.58.2 Constructor & Destructor Documentation

6.58.2.1 Triangle()

```
Infohazard.HyperNav.Triangle.Triangle (
        int vertex1,
        int vertex2,
        int vertex3 )
```

Construct a new Triangle with the given indices.

The order of the indices doesn't matter; the same Triangle is constructed either way. No two of the indices can be the same.

Parameters

vertex1	First vertex index.
vertex2	Second vertex index.
vertex3	Third vertex index.

6.58.3 Member Function Documentation

6.58.3.1 Equals() `[1/2]` `override bool Infohazard.HyperNav.Triangle.Equals ( object obj )`

Compare to another object.

Parameters

obj	Object to compare to.

Returns

Whether the two objects are equal.

6.58.3.2 Equals() `[2/2]` `bool Infohazard.HyperNav.Triangle.Equals ( Triangle other )`

Compare to another Triangle.

Parameters

other	Triangle to compare to.

Returns

Whether the two triangles are equal.

6.58.3.3 GetHashCode() `override int Infohazard.HyperNav.Triangle.GetHashCode ( )`

Get integer for use with hash table.

Returns

Integer hash code.

6.58.4 Member Data Documentation

6.58.4.1 _maxVertex `int Infohazard.HyperNav.Triangle._maxVertex [private]`

(Serialized) Third vertex index, which is the larger of the three.

6.58.4.2 _midVertex `int Infohazard.HyperNav.Triangle._midVertex [private]`

(Serialized) Second vertex index, which is the middle of the three.

6.58.4.3 _minVertex `int Infohazard.HyperNav.Triangle._minVertex [private]`

(Serialized) First vertex index, which is the lower of the three.

6.58.5 Property Documentation

6.58.5.1 Vertex1 `int Infohazard.HyperNav.Triangle.Vertex1 [get]`

First vertex index, which is the lower of the three.

6.58.5.2 Vertex2 `int Infohazard.HyperNav.Triangle.Vertex2 [get]`

Second vertex index, which is the middle of the three.

6.58.5.3 Vertex3 `int Infohazard.HyperNav.Triangle.Vertex3 [get]`

Third vertex index, which is the larger of the three.

The documentation for this struct was generated from the following file:

- Runtime/Utility/Triangle.cs

6.59 Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T > Struct Template Reference

This is a simple wrapper for unmanaged memory which bypasses Unity's safety checks. This allows arrays to be nested in other arrays (or in structs contained in arrays). Note that you must keep a reference to the original NativeArray, or Unity will detect a memory leak.

Public Member Functions

- unsafe void Dispose ()

  Free the memory if it has been allocated directly.

Static Public Member Functions

- static unsafe UnsafeArrayPtr< T > ToPointer (in NativeArray< T > array)

  Create a pointer to the given NativeArray.

Public Attributes

- readonly int Length

  Length of the array.
- readonly IntPtr Pointer

  Pointer to the start of the array.
- readonly Allocator Allocator

  Allocator used to allocate the array, or None if it was created from a NativeArray.

Properties

- unsafe ref T this[int index] `[get]`

  Get a reference to the element at the given index (can be used to set values as well).

6.59.1 Detailed Description

This is a simple wrapper for unmanaged memory which bypasses Unity's safety checks. This allows arrays to be nested in other arrays (or in structs contained in arrays). Note that you must keep a reference to the original NativeArray, or Unity will detect a memory leak.

Template Parameters

T	Element type of the array.

Type Constraints

T* : *unmanaged

6.59.2 Member Function Documentation

6.59.2.1 Dispose() `unsafe void Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.Dispose ( )`

Free the memory if it has been allocated directly.

If this pointer is wrapping a NativeArray, this does nothing.

6.59.2.2 ToPointer() `static unsafe UnsafeArrayPtr< T > Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.ToPointer (`

```
            in NativeArray< T > array )  [static]
```

Create a pointer to the given NativeArray.

Parameters

array	Array to create a pointer to.

Returns

The created pointer.

6.59.3 Member Data Documentation

6.59.3.1 Allocator `readonly Allocator Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.Allocator`

Allocator used to allocate the array, or None if it was created from a NativeArray.

6.59.3.2 Length `readonly int Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.Length`

Length of the array.

6.59.3.3 Pointer `readonly IntPtr Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.Pointer`

Pointer to the start of the array.

6.59.4 Property Documentation

6.59.4.1 this[int index] `unsafe ref T Infohazard.HyperNav.Jobs.UnsafeArrayPtr< T >.this[int index]  [get]`

Get a reference to the element at the given index (can be used to set values as well).

Parameters

index	The index.

Exceptions

InvalidOperationException	(Dev Only) If underlying array is not set.
IndexOutOfRangeException	(Dev Only) If index is outside the bounds of the array.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/UnsafeArrayPtr.cs

6.60 Infohazard.HyperNav.Jobs.VisitedNodeInfo Struct Reference

The information that has been discovered about a node during pathfinding, which is stored in a table.

Public Attributes

- bool HasPrevious

  If false, this is the first node in the sequence.
- PendingPathNode Previous

  The node to travel from to get the shortest known path to this node.
- float Heuristic

  Distance from this node to the destination.
- float CumulativeCost

  The total path distance of the shortest known path to this node.
- bool Visited

  If node has been visited, the shortest path to this node is finalized.
- float4 Position

  Position to enter the node when coming from the previous node in the best known shortest path.

6.60.1 Detailed Description

The information that has been discovered about a node during pathfinding, which is stored in a table.

6.60.2 Member Data Documentation

6.60.2.1 CumulativeCost `float Infohazard.HyperNav.Jobs.VisitedNodeInfo.CumulativeCost`

The total path distance of the shortest known path to this node.

6.60.2.2 HasPrevious `bool Infohazard.HyperNav.Jobs.VisitedNodeInfo.HasPrevious`

If false, this is the first node in the sequence.

6.60.2.3 Heuristic `float Infohazard.HyperNav.Jobs.VisitedNodeInfo.Heuristic`

Distance from this node to the destination.

6.60.2.4 Position `float4 Infohazard.HyperNav.Jobs.VisitedNodeInfo.Position`

Position to enter the node when coming from the previous node in the best known shortest path.

6.60.2.5 Previous `PendingPathNode Infohazard.HyperNav.Jobs.VisitedNodeInfo.Previous`

The node to travel from to get the shortest known path to this node.

6.60.2.6 Visited `bool Infohazard.HyperNav.Jobs.VisitedNodeInfo.Visited`

If node has been visited, the shortest path to this node is finalized.

The documentation for this struct was generated from the following file:

- Runtime/Jobs/NavPathJob.cs