Quantum Namespace Reference

Classes

class	CallbackChecksumComputed Callback called when a checksum has been computed. More...
class	CallbackChecksumError Callback called on a checksum error. More...
class	CallbackChecksumErrorFrameDump Callback called when due to a checksum error a frame is dumped. More...
class	CallbackEventCanceled Callback called when an event raised in a predicted frame was canceled in a verified frame due to a roll-back / missed prediction. Synchronised events are only raised on verified frames and thus will never be canceled; this is useful to graciously discard non-sync'ed events in the view. More...
class	CallbackEventConfirmed Callback called when an event was confirmed by a verified frame. More...
class	CallbackGameDestroyed Callback called when the game was destroyed. More...
class	CallbackGameResynced Callback called when the game has been re-synchronized from a snapshot. More...
class	CallbackGameStarted Callback called when the game has been started. More...
class	CallbackInputConfirmed Callback when local input was confirmed. More...
class	CallbackPluginDisconnect Callback called when the local client is disconnected by the plugin. More...
class	CallbackPollInput Callback called when the simulation queries local input. More...
class	CallbackSimulateFinished Callback called when frame simulation has completed. More...
class	CallbackUpdateView Callback guaranteed to be called every rendered frame. More...
struct	CharacterController2D
struct	CharacterController2DMovement Result of a 2D KCC raw movement query. More...
struct	CharacterController3D

struct	CharacterController3DMovement Result of a 3D KCC raw movement query. More...
struct	CollisionInfo2D Info about a collision between two 2D physics colliders. More...
struct	CollisionInfo3D Info about a collision between two 3D physics colliders. More...
struct	EntityRef Quantum entity reference. More...
struct	ExitInfo2D Info about two entities that were colliding in the 2D Physics. More...
struct	ExitInfo3D Info about two entities that were colliding in the 3D Physics. More...
class	Frame The user implementation of FrameBase that resides in the project quantum_state and has access to all user relevant classes. More...
interface	ISignalOnCollision2D Interface for receiving callbacks once per frame while two non-trigger 2D colliders are touching. More...
interface	ISignalOnCollision3D Interface for receiving callbacks once per frame while two non-trigger 3D colliders are touching. More...
interface	ISignalOnCollisionEnter2D Interface for receiving callbacks once two non-trigger 2D colliders start touching. More...
interface	ISignalOnCollisionEnter3D Interface for receiving callbacks once two non-trigger 3D colliders start touching. More...
interface	ISignalOnCollisionExit2D Interface for receiving callbacks once two non-trigger 2D colliders stop touching. More...
interface	ISignalOnCollisionExit3D Interface for receiving callbacks once two non-trigger 3D colliders stop touching. More...
interface	ISignalOnNavMeshMoveAgent Signal is called when the agent should move. The desired direction is influence by avoidance. More...
interface	ISignalOnNavMeshSearchFailed Signal is fired when the agent could not find a path in the agent update after using NavMeshSteeringAgent.SetTarget(Core.FrameBase, FPVector2, NavMesh, bool) More...
interface	ISignalOnNavMeshWaypointReached

	Signal is fired when an agent reaches a waypoint. More...
interface	ISignalOnTrigger2D Interface for receiving callbacks once per frame while a non-trigger and a trigger 2D colliders are touching. More...
interface	ISignalOnTrigger3D Interface for receiving callbacks once per frame while a non-trigger and a trigger 3D colliders are touching. More...
interface	ISignalOnTriggerEnter2D Interface for receiving callbacks once a non-trigger and a trigger 2D colliders start touching. More...
interface	ISignalOnTriggerEnter3D Interface for receiving callbacks once a non-trigger and a trigger 3D colliders start touching. More...
interface	ISignalOnTriggerExit2D Interface for receiving callbacks once a non-trigger and a trigger 2D colliders stop touching. More...
interface	ISignalOnTriggerExit3D Interface for receiving callbacks once a non-trigger and a trigger 3D colliders stop touching. More...
class	Navigation Navigation API More...
class	NavMesh The asset object that contains a Quantum navigation mesh. The object loads an additional data file during the Loaded(IResourceManager, Native.Allocator). This is because of size limitations when loading the data with Unity serialization. More...
class	NavMeshAgentConfig The configuration file for navmesh agent components. More...
struct	NavMeshAgentSteeringData Navmesh agent steering data passed into callbacks. More...
struct	NavMeshAvoidanceAgent (requires SteeringAgent and PathfinderAgent) More...
struct	NavMeshPathfinder The NavMeshAgent is an entity component for automated navmesh navigation and steering. More...
struct	NavMeshRegionMask Internally stores a unsigned long to be able to toggle 64 different regions. More...
struct	NavMeshSteeringAgent Requires NavMeshPathfinder component. More...
struct	PhysicsJoints2D

	A component holding one or more Physics2D.Joint, defining connections between a 2D Physics Body and anchors according to velocity and/or position constraints. More...
struct	PhysicsJoints3D A component holding one or more Joint3D, defining connections between a 3D Physics Body and anchors according to velocity and/or position constraints. More...
struct	PlayerRef Represents a Quantum player. More...
class	QuantumGame QuantumGame acts as an interface to the simulation from the client code's perspective. More...
class	QuantumGameFlags This class contains values for flags that will be accessible with QuantumGame.GameFlags. Built-in flags control some aspects of QuantumGame inner workings, without affecting the simulation outcome. More...
class	RuntimeConfig In contrast to the SimulationConfig, which has only static configuration data, the RuntimeConfig holds information that can be different from game to game. More...
struct	Shape2D Defines a 2D shape with Type and data disposed in a union-like structure. All shapes have a UserTag, BroadRadius and Centroid. All non-compound shapes have a LocalTransform and their Centroid always match their local transform position. More...
struct	Shape3D Defines a 3D shape with Type and data disposed in a union-like structure. All shapes have a UserTag, BroadRadius and Centroid. All non-compound shapes have a LocalTransform and their Centroid always match their local transform position. More...
class	SimulationConfig The SimulationConfig holds parameters used in the ECS layer and inside core systems like physics and navigation. More...
struct	StaticColliderData Information about a static collider. More...
struct	Transform2D The Transform2D is an entity component providing position and rotation a 2D object. More...
struct	Transform3D The Transform3D is an entity component providing position and rotation for a 3D object. More...
struct	TriggerInfo2D

	Info about a collision between a trigger and a non-trigger 2D physics colliders. More...
struct	TriggerInfo3D Info about a collision between a trigger and a non-trigger 3D physics colliders. More...

Enumerations

enum	CallbackFlags : int Represents which collision callbacks will be called for an entity. More...
enum	SimulationUpdateTime The type of measuring time progressions to update the local simulation. More...

Enumeration Type Documentation

◆ CallbackFlags

enum Quantum.CallbackFlags : int

strong

Represents which collision callbacks will be called for an entity.

By default, no callbacks are called unless at least one of the entities involved in a collision have the respective flag set.

The callbacks are called for every entity involved in a collision that has the respective collision type flag set.

No collision is checked between two kinematic colliders that are both trigger or both non-trigger.

Enumerator	
None	Set None to stop receiving callbacks for an entity.
OnDynamicCollision	Called once per frame while two non-trigger colliders are touching. Related signals: ISignalOnCollision2D and ISignalOnCollision3D.
OnDynamicCollisionEnter	Called once two non-trigger colliders start touching. Related signals: ISignalOnCollisionEnter2D and ISignalOnCollisionEnter3D.
OnDynamicCollisionExit	Called once two non-trigger colliders stop touching. Related signals: ISignalOnCollisionExit2D and ISignalOnCollisionExit3D.
OnStaticCollision	Called once per frame while a non-trigger collider is touching a non-trigger static collider. Related signals: ISignalOnCollision2D and ISignalOnCollision3D.
OnStaticCollisionEnter	Called once a non-trigger collider start touching a non-trigger static collider. Related signals: ISignalOnCollisionEnter2D and ISignalOnCollisionEnter3D.
OnStaticCollisionExit	Called once a non-trigger collider stop touching a non-trigger static collider. Related signals: ISignalOnCollisionExit2D and ISignalOnCollisionExit3D.
OnDynamicTrigger	Called once per frame while a trigger collider is touching a non-trigger collider.

	Related signals: ISignalOnTrigger2D and ISignalOnTrigger3D.
OnDynamicTriggerEnter	Called once a trigger collider start touching a non-trigger collider. Related signals: ISignalOnTriggerEnter2D and ISignalOnTriggerEnter3D.
OnDynamicTriggerExit	Called once a trigger collider stop touching a non-trigger collider. Related signals: ISignalOnTriggerExit2D and ISignalOnTriggerExit3D.
OnStaticTrigger	Called once per frame while a non-trigger collider is touching a trigger static collider. Related signals: ISignalOnTrigger2D and ISignalOnTrigger3D.
OnStaticTriggerEnter	Called once a non-trigger collider start touching a trigger static collider. Related signals: ISignalOnTriggerEnter2D and ISignalOnTriggerEnter3D.
OnStaticTriggerExit	Called once a non-trigger collider stop touching a trigger static collider. Related signals: ISignalOnTriggerExit2D and ISignalOnTriggerExit3D.

◆ SimulationUpdateTime

enum Quantum.SimulationUpdateTime

strong

The type of measuring time progressions to update the local simulation.

Caveat: Changing it will make every client use the setting which might be undesirable when only used for debugging.

Enumerator	
Default	Internal stopwatch. Recommended for releasing games.
EngineDeltaTime	Engine (Unity) delta time. Extremely useful when pausing the Unity simulation during debugging for example. Caveat: the setting can cause issues with time synchronization when initializing online matches: the time tracking can be inaccurate under load (e.g.level loading) and result in a lot of large extra time syncs request and canceled inputs for a client when starting an online game.
EngineUnscaledDeltaTime	Engine unscaled delta time.