Coroutine Manager Pro

1.0

Mon Jan 4 2016 15:58:08

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

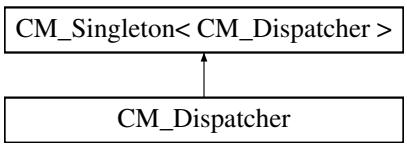
Chapter 3

Class Documentation

3.1 CM_Dispatcher Class Reference

Used to start coroutines in the main thread from seperate threads.

Inheritance diagram for CM_Dispatcher:



Public Member Functions

- void AddToExecuteQueue (IEnumerator job)

  Adds to execute queue. Coroutines in the queue are executed during the next timestep in the update method.

Additional Inherited Members

3.1.1 Detailed Description

Used to start coroutines in the main thread from seperate threads.

3.1.2 Member Function Documentation

3.1.2.1 void CM_Dispatcher.AddToExecuteQueue (IEnumerator *job*)

Adds to execute queue. Coroutines in the queue are executed during the next timestep in the update method.

Parameters

job	Job.

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

- CM_Dispatcher.cs

3.2 CM_GlobalCoroutineManager< T > Class Template Reference

The base class of Global Coroutine Managers. Provides access to a singular global copy of the class.

Protected Member Functions

- virtual CM_Job **MakeJob** (string id, IEnumerator routine, bool addListenerToOnComplete=true)
- virtual CM_Job **MakeJob** (CM_Job job)
- void **AutoGenerateJobId** (CM_Job job)
- abstract void **HandlejobComplete** (object sender, CM_JobEventArgs e)

Properties

- static T Global `[get]`

  Access to a global instance.

3.2.1 Detailed Description

The base class of Global Coroutine Managers. Provides access to a singular global copy of the class.

Type Constraints

T* : *new()

3.2.2 Property Documentation

3.2.2.1 T CM_GlobalCoroutineManager< T >.Global `[static],[get]`

Access to a global instance.

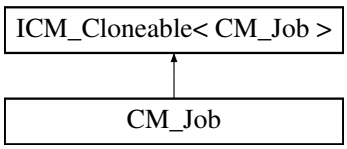
The documentation for this class was generated from the following file:

- CM_GlobalCoroutineManager.cs

3.3 CM_Job Class Reference

The main coroutine job class. Encapsulates the behaviour for a single coroutine job. Provides access to status (i.e. running, paused, killed etc),

Inheritance diagram for CM_Job:



Public Member Functions

- CM_Job Clone ()

  Clone this instance.

- CM_Job[ ] Clone (int numOfCopies)

  Clone this instance.
- CM_Job Start ()

  Start this instance. Runs the coroutine immediately.
- CM_Job Start (float delayInSeconds)

  Start the specified instance after delayInSeconds. The coroutine is added to CM_Dispatcher job queue to be executed in the next timestep as a coroutine cannot be started in a seperate thread.
- CM_Job Repeat ()

  Sets this instance to repeat. The job is repeated when it has finished processing.
- CM_Job Repeat (int numOfTimes)

  Sets this instance to repeat. The job is repeated a set number of times.
- CM_Job **StopRepeat** ()
- CM_Job StopRepeat (float delayInSeconds)

  Stops the repeat after a specified delay in seconds.
- CM_Job Pause ()

  Pause this instance.
- CM_Job Pause (float delayInSeconds)

  Pause the specified instance after delayInSeconds.
- CM_Job Resume ()

  Resume this instance.
- CM_Job Resume (float delayInSeconds)

  Resume the specified instance after delayInSeconds.
- void Kill ()

  Kill this instance. Stops the running coroutine.
- void Kill (float delayInSeconds)

  Kill this instance. Stops the running coroutine after delayInSeconds.
- CM_Job AddChild (CM_Job childJob)

  Adds a child job.
- CM_Job AddChild (IEnumerator childJob)

  Create a new job using the provided Enumerator and adds as a child job.
- CM_Job RemoveChildJob (CM_Job childJob)

  Removes a child job if present.
- CM_Job NotifyOnJobFinishedRunning (EventHandler< CM_JobEventArgs > e)

  Subscribes to the jobFinishedRunning event
- CM_Job RemoveNotifyOnJobFinishedRunning (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the jobFinishedRunning event
- CM_Job NotifyOnJobStarted (EventHandler< CM_JobEventArgs > e)

  Subscribes to the jobStarted event.
- CM_Job RemoveNotifyOnJobStarted (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the jobStarted event.
- CM_Job NotifyOnJobPaused (EventHandler< CM_JobEventArgs > e)

  Subscribes to the job paused event.
- CM_Job RemoveNotifyOnJobPaused (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the job paused event.
- CM_Job NotifyOnJobResumed (EventHandler< CM_JobEventArgs > e)

  Subscribes to the job resumed event.
- CM_Job RemoveNotifyOnJobResumed (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the job resumed event.
- CM_Job NotifyOnJobComplete (EventHandler< CM_JobEventArgs > e)

  Subscribes to the the jobComplete event.
- CM_Job RemoveNotifyOnJobComplete (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the the jobComplete event.

- CM_Job NotifyOnChildJobStarted (EventHandler< CM_JobEventArgs > e)

  Subscribes to the the childJobsStarted event.
- CM_Job RemoveNotifyOnChildJobStarted (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the the childJobsStarted event.
- CM_Job NotifyOnChildJobComplete (EventHandler< CM_JobEventArgs > e)

  Subscribes to the the childJobsComplete event.
- CM_Job RemoveNotifyOnChildJobComplete (EventHandler< CM_JobEventArgs > e)

  Unsubscribes to the the childJobsComplete event.

Static Public Member Functions

- static CM_Job Make (IEnumerator coroutine)

  Returns an initialised CM_Job instance. Provides static access to class.
- static CM_Job Make (IEnumerator coroutine, string id)

  Returns an initialised CM_Job instance with the specified id. Provides static access to class.
- static CM_Job[ ] Builder (params IEnumerator[ ] coroutines)

  Builds the specified coroutines into CM_Job instances.

Protected Member Functions

- void OnJobFinishedRunning (CM_JobEventArgs e)

  Raises the job finished running event.
- void OnJobStarted (CM_JobEventArgs e)

  Raises the job started event.
- void OnJobComplete (CM_JobEventArgs e)

  Raises the job complete event.
- void OnJobPaused (CM_JobEventArgs e)

  Raises the job paused event.
- void OnJobResumed (CM_JobEventArgs e)

  Raises the job resumed event.
- void OnChildJobsStarted (CM_JobEventArgs e)

  Raises the child jobs started event.
- void OnChildJobsComplete (CM_JobEventArgs e)

  Raises the child jobs complete event.

Properties

- string id `[get, set]`

  Gets or sets the identifier. The identifier is a unique key used by CM_JobManager to reference individual jobs.
- bool running `[get]`

  Gets a value indicating whether this CM_Job is running.
- bool paused `[get]`

  Gets a value indicating whether this CM_Job is paused.
- bool jobKilled `[get]`

  Gets a value indicating whether this CM_Job job was killed or was allowed to complete.
- bool repeating `[get]`

  Gets a value indicating whether this CM_Job is repeating.
- int numOfTimesExecuted `[get]`

  Gets the number of times this job has been executed.
- IEnumerator coroutine `[get]`

  Gets the coroutine of this job.

3.3.1 Detailed Description

The main coroutine job class. Encapsulates the behaviour for a single coroutine job. Provides access to status (i.e. running, paused, killed etc),

3.3.2 Member Function Documentation

3.3.2.1 CM_Job CM_Job.AddChild (CM_Job *childJob*)

Adds a child job.

Returns

The child.

Parameters

childJob	Child job.

3.3.2.2 CM_Job CM_Job.AddChild (IEnumerator *childJob*)

Create a new job using the provided Enumerator and adds as a child job.

Returns

The child.

Parameters

childJob	Child job.

3.3.2.3 static CM_Job [ ] CM_Job.Builder (params IEnumerator[ ] *coroutines*) [static]

Builds the specified coroutines into CM_Job instances.

Parameters

coroutines	The built jobs.

3.3.2.4 CM_Job CM_Job.Clone ( )

Clone this instance.

3.3.2.5 CM_Job [ ] CM_Job.Clone (int *numOfCopies*)

Clone this instance.

Parameters

numOfCopies	Number of copies to create.

3.3.2.6 void CM_Job.Kill ( )

Kill this instance. Stops the running coroutine.

3.3.2.7 void CM_Job.Kill (float *delayInSeconds*)

Kill this instance. Stops the running coroutine after delayInSeconds.

Parameters

delayInSeconds	Delay in seconds until instance killed.

3.3.2.8 static CM_Job CM_Job.Make (IEnumerator *coroutine*) [static]

Returns an initialised CM_Job instance. Provides static access to class.

Parameters

coroutine	Coroutine.

3.3.2.9 static CM_Job CM_Job.Make (IEnumerator *coroutine*, string *id*) [static]

Returns an initialised CM_Job instance with the specified id. Provides static access to class.

Parameters

coroutine	Coroutine.
id	Identifier.

3.3.2.10 CM_Job CM_Job.NotifyOnChildJobComplete (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the the childJobsComplete event.

Parameters

e	The eventhandler to be invoked on event.

3.3.2.11 CM_Job CM_Job.NotifyOnChildJobStarted (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the the childJobsStarted event.

Parameters

e	The eventhandler to be invoked on event.

3.3.2.12 CM_Job CM_Job.NotifyOnJobComplete (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the the jobComplete event.

Parameters

e	The eventhandler to be invoked on event.

3.3.2.13 CM_Job CM_Job.NotifyOnJobFinishedRunning (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the jobFinishedRunning event

Parameters

e	The eventhandler to be invoked on event.

3.3.2.14 CM_Job CM_Job.NotifyOnJobPaused (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the job paused event.

Returns

The on job paused.

Parameters

e	E.

3.3.2.15 CM_Job CM_Job.NotifyOnJobResumed (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the job resumed event.

Returns

The on job paused.

Parameters

e	E.

3.3.2.16 CM_Job CM_Job.NotifyOnJobStarted (EventHandler< CM_JobEventArgs > *e*)

Subscribes to the jobStarted event.

Parameters

e	The eventhandler to be invoked on event.

3.3.2.17 void CM_Job.OnChildJobsComplete (CM_JobEventArgs *e*) [protected]

Raises the child jobs complete event.

Parameters

e	E.

3.3.2.18 void CM_Job.OnChildJobsStarted (CM_JobEventArgs *e*) [protected]

Raises the child jobs started event.

Parameters

e	E.

3.3.2.19 void CM_Job.OnJobComplete (CM_JobEventArgs *e*) [protected]

Raises the job complete event.

Parameters

e	E.

3.3.2.20 void CM_Job.OnJobFinishedRunning (CM_JobEventArgs *e*) [protected]

Raises the job finished running event.

Parameters

e	E.

3.3.2.21 void CM_Job.OnJobPaused (CM_JobEventArgs *e*) [protected]

Raises the job paused event.

Parameters

e	E.

3.3.2.22 void CM_Job.OnJobResumed (CM_JobEventArgs *e*) [protected]

Raises the job resumed event.

Parameters

e	E.

3.3.2.23 void CM_Job.OnJobStarted (CM_JobEventArgs *e*) [protected]

Raises the job started event.

Parameters

e	E.

3.3.2.24 CM_Job CM_Job.Pause ( )

Pause this instance.

3.3.2.25 CM_Job CM_Job.Pause (float *delayInSeconds*)

Pause the specified instance after delayInSeconds.

Parameters

delayInSecods	Delay in secods until instance is paused.

3.3.2.26 CM_Job CM_Job.RemoveChildJob (CM_Job *childJob*)

Removes a child job if present.

Returns

The child job.

Parameters

childJob	Child job.

3.3.2.27 CM_Job CM_Job.RemoveNotifyOnChildJobComplete (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the the childJobsComplete event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.28 CM_Job CM_Job.RemoveNotifyOnChildJobStarted (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the the childJobsStarted event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.29 CM_Job CM_Job.RemoveNotifyOnJobComplete (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the the jobComplete event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.30 CM_Job CM_Job.RemoveNotifyOnJobFinishedRunning (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the jobFinishedRunning event

Parameters

e	The eventhandler to be invoked on event.

3.3.2.31 CM_Job CM_Job.RemoveNotifyOnJobPaused (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the job paused event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.32 CM_Job CM_Job.RemoveNotifyOnJobResumed (EventHandler< CM_JobEventArgs > e)

Unsubscribes to the job resumed event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.33 CM_Job CM_Job.RemoveNotifyOnJobStarted (EventHandler< CM_JobEventArgs > *e*)

Unsubscribes to the jobStarted event.

Parameters

e	The eventhandler to be unsubscribed.

3.3.2.34 CM_Job CM_Job.Repeat ( )

Sets this instance to repeat. The job is repeated when it has finished processing.

3.3.2.35 CM_Job CM_Job.Repeat (int *numOfTimes*)

Sets this instance to repeat. The job is repeated a set number of times.

3.3.2.36 CM_Job CM_Job.Resume ( )

Resume this instance.

3.3.2.37 CM_Job CM_Job.Resume (float *delayInSeconds*)

Resume the specified instance after delayInSeconds.

Parameters

delayInSecods	Delay in secods until instance is resumed.

3.3.2.38 CM_Job CM_Job.Start ( )

Start this instance. Runs the coroutine immediately.

3.3.2.39 CM_Job CM_Job.Start (float *delayInSeconds*)

Start the specified instance after delayInSeconds. The coroutine is added to CM_Dispatcher job queue to be executed in the next timestep as a coroutine cannot be started in a seperate thread.

Parameters

delayInSecods	Delay in secods until instance is processed.

3.3.2.40 CM_Job CM_Job.StopRepeat (float *delayInSeconds*)

Stops the repeat after a specified delay in seconds.

Returns

The repeat.

Parameters

delayInSeconds	Delay in seconds.

3.3.3 Property Documentation

3.3.3.1 IEnumerator CM_Job.coroutine [get]

Gets the coroutine of this job.

The coroutine.

3.3.3.2 string CM_Job.id [get], [set]

Gets or sets the identifier. The identifier is a unique key used by CM_JobManager to reference individual jobs.

The identifier.

3.3.3.3 bool CM_Job.jobKilled [get]

Gets a value indicating whether this CM_Job job was killed or was allowed to complete.

`true` if job killed; otherwise, `false`.

3.3.3.4 int CM_Job.numOfTimesExecuted [get]

Gets the number of times this job has been executed.

The number of times executed.

3.3.3.5 bool CM_Job.paused [get]

Gets a value indicating whether this CM_Job is paused.

`true` if paused; otherwise, `false`.

3.3.3.6 bool CM_Job.repeating [get]

Gets a value indicating whether this CM_Job is repeating.

`true` if repeating; otherwise, `false`.

3.3.3.7 bool CM_Job.running [get]

Gets a value indicating whether this CM_Job is running.

`true` if running; otherwise, `false`.

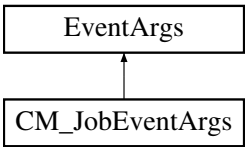
The documentation for this class was generated from the following file:

- CM_Job.cs

3.4 CM_JobEventArgs Class Reference

Arguements used in events raised by CM_Job.

Inheritance diagram for CM_JobEventArgs:



Public Member Functions

- CM_JobEventArgs (CM_Job job, CM_Job[ ] childJobs)

  Initializes a new instance of the CM_JobEventArgs class.

Properties

- CM_Job job `[get]`

  Gets the current job.
- CM_Job[ ] childJobs `[get]`

  Gets the child jobs (if any).
- bool hasChildJobs `[get]`

  Gets a value indicating whether this CM_JobEventArgs has child jobs.

3.4.1 Detailed Description

Arguements used in events raised by CM_Job.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 CM_JobEventArgs.CM_JobEventArgs (CM_Job *job*, CM_Job[ ] *childJobs*)

Initializes a new instance of the CM_JobEventArgs class.

Parameters

job	Job.
childJobs	Child jobs.

3.4.3 Property Documentation

3.4.3.1 CM_Job [ ] CM_JobEventArgs.childJobs `[get]`

Gets the child jobs (if any).

The child jobs.

3.4.3.2 bool CM_JobEventArgs.hasChildJobs `[get]`

Gets a value indicating whether this CM_JobEventArgs has child jobs.

`true` if has child jobs; otherwise, `false`.

3.4.3.3 CM_Job CM_JobEventArgs.job [get]

Gets the current job.

The job.

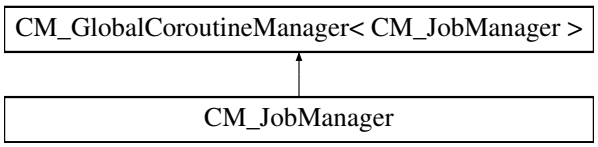
The documentation for this class was generated from the following file:

- CM_JobEventArgs.cs

3.5 CM_JobManager Class Reference

The main job manager class. Encapsulates the behaviour for global and local job managers. Provides access to events and public access to stored jobs.

Inheritance diagram for CM_JobManager:



Public Member Functions

- CM_JobManager AddJob (CM_Job job)

  Adds a job to the job manager.
- CM_JobManager AddJob (string id, IEnumerator routine)

  Creates a job with the specified id and routine and adds to job manager.
- CM_JobManager AddJob (IList< CM_Job > jobs)

  Adds the provided jobs to the job manager.
- CM_JobManager AddJob (params CM_Job[] jobs)

  Adds the provided jobs to the job manager.
- CM_JobManager RemoveJob (CM_Job job)

  Removes the job if owned by this instance of job manager.
- CM_JobManager RemoveJob (string id)

  Removes the job if owned by this instance of job manager.
- CM_JobManager StartCoroutine (CM_Job job)

  Starts the specified coroutine if owned by this instance of job manager. Job is searched using job id.
- CM_JobManager StartCoroutine (string id)

  Starts the specified coroutine if owned by this instance of job manager.
- CM_JobManager StartAll ()

  Starts all jobs owned by this instance.
- CM_JobManager StartAll (float delayInSeconds)

  Starts all jobs owned by this instance after delay in seconds.
- CM_JobManager StopCoroutine (CM_Job job)

  Stops the specified coroutine if owned by this instance of job manager. Job is searched using job id.
- CM_JobManager StopCoroutine (string id)

  Stops the specified coroutine if owned by this instance of job manager.
- CM_JobManager PauseCoroutine (CM_Job job)

  Pauses the specified coroutine if owned by this instance of job manager. Job is searched using job id.
- CM_JobManager PauseCoroutine (string id)

Stops the specified coroutine if owned by this instance of job manager.

- CM_JobManager ResumeCoroutine (CM_Job job)

  Resumes the specified coroutine if owned by this instance of job manager. Job is searched using job id.
- CM_JobManager ResumeCoroutine (string id)

  Stops the specified coroutine if owned by this instance of job manager.
- CM_JobManager PauseAll ()

  Pauses all jobs owned by this instance. Raises allJobsPaused event.
- CM_JobManager PauseAll (float delayInSeconds)

  Pauses all jobs owned by this instance after delay in seconds. Raises allJobsPaused event.
- CM_JobManager ResumeAll ()

  Resumes all jobs owned by this instance. Raises allJobsResumedEvent.
- CM_JobManager ResumeAll (float delayInSeconds)

  Resumes all jobs owned by this instance after delay in seconds. Raises allJobsResumedEvent.
- CM_JobManager KillAll ()

  Kills all jobs owned by this instance. Raises allJobsKilled event.
- CM_JobManager KillAll (float delayInSeconds)

  Kills all jobs owned by this instance after delay in seconds. Raises allJobsKilled event.
- CM_JobManager ClearJobList ()

  Clears the job list owned by this instance. It does not kill the jobs so they will continue to run. Raises allJobsCleared event.
- bool HasJob (string id)

  Determines whether this instance has the job with the specified id.
- bool IsRunning (CM_Job job)

  Determines whether the specified job is executing.
- bool IsRunning (string id)

  Determines whether the specified job is executing.
- CM_JobManager NotifyOnJobAdded (EventHandler< CM_JobManagerJobEditedEventArgs > e)

  Subscribes to the the jobAdded event.
- CM_JobManager RemoveNotifyOnJobAdded (EventHandler< CM_JobManagerJobEditedEventArgs > e)

  Unsubscribes to the the jobAdded event.
- CM_JobManager NotifyOnJobRemoved (EventHandler< CM_JobManagerJobEditedEventArgs > e)

  Subscribes to the the jobRemoved event.
- CM_JobManager RemoveNotifyOnJobRemoved (EventHandler< CM_JobManagerJobEditedEventArgs > e)

  Unsubscribes to the the jobRemoved event.
- CM_JobManager NotifyOnAllJobsPaused (EventHandler< CM_JobManagerEventArgs > e)

  Subscribes to the the jobPaused event.
- CM_JobManager RemoveNotifyOnAllJobsPaused (EventHandler< CM_JobManagerEventArgs > e)

  Unsubscribes to the the jobPaused event.
- CM_JobManager NotifyOnAllJobsResumed (EventHandler< CM_JobManagerEventArgs > e)

  Subscribes to the the jobResumed event.
- CM_JobManager RemoveNotifyOnAllJobsResumed (EventHandler< CM_JobManagerEventArgs > e)

  Unsubscribes to the the jobResumed event.
- CM_JobManager NotifyOnAllJobsKilled (EventHandler< CM_JobManagerEventArgs > e)

  Subscribes to the the allJobsKilled event.
- CM_JobManager RemoveNotifyOnAllJobsKilled (EventHandler< CM_JobManagerEventArgs > e)

  Unsubscribes to the the allJobsKilled event.
- CM_JobManager NotifyOnAllJobsCleared (EventHandler< CM_JobManagerEventArgs > e)

  Subscribes to the the allJobsCleared event.
- CM_JobManager RemoveNotifyOnAllJobsCleared (EventHandler< CM_JobManagerEventArgs > e)

  Unsubscribes to the the allJobsCleared event.

- void OnJobRemoved (CM_JobManagerJobEditedEventArgs e)

  Raises the job removed event.
- void OnAllJobsResumed (CM_JobManagerEventArgs e)

  Raises the all jobs resumed event.
- void OnAllJobsPaused (CM_JobManagerEventArgs e)

  Raises the all jobs paused event.
- void OnAllJobsKilled (CM_JobManagerEventArgs e)

  Raises the all jobs killed event.
- void OnAllJobsCleared (CM_JobManagerEventArgs e)

  Raises the all jobs cleared event.

Static Public Member Functions

- static CM_JobManager Make ()

  Returns an initialised CM_JobManager instance. Provides static access to class.

Protected Member Functions

- void OnJobAdded (CM_JobManagerJobEditedEventArgs e)

  Raises the job added event.
- override void **HandlejobComplete** (object sender, CM_JobEventArgs e)

Events

- EventHandler< CM_JobManagerJobEditedEventArgs > jobAdded

  Raised when job added.
- EventHandler< CM_JobManagerJobEditedEventArgs > jobRemoved

  Raised when job removed.
- EventHandler< CM_JobManagerEventArgs > allJobsKilled

  Raised when all jobs killed.
- EventHandler< CM_JobManagerEventArgs > allJobsResumed

  Raised when all jobs resumed.
- EventHandler< CM_JobManagerEventArgs > allJobsPaused

  Raised when all jobs paused.
- EventHandler< CM_JobManagerEventArgs > allJobsCleared

  Raised when all jobs cleared.

Additional Inherited Members

3.5.1 Detailed Description

The main job manager class. Encapsulates the behaviour for global and local job managers. Provides access to events and public access to stored jobs.

3.5.2 Member Function Documentation

3.5.2.1 CM_JobManager CM_JobManager.AddJob (CM_Job *job*)

Adds a job to the job manager.

Returns

The job.

Parameters

job	Job.

3.5.2.2 CM_JobManager CM_JobManager.AddJob (string *id,* IEnumerator *routine*)

Creates a job with the specified id and routine and adds to job manager.

Returns

The job manager.

Parameters

id	Identifier of job.
routine	Routine.

3.5.2.3 CM_JobManager CM_JobManager.AddJob (IList< CM_Job > *jobs*)

Adds the provided jobs to the job manager.

Returns

The job manager.

Parameters

jobs	Jobs to add to this instance.

3.5.2.4 CM_JobManager CM_JobManager.AddJob (params CM_Job[ ] *jobs*)

Adds the provided jobs to the job manager.

Returns

The job manager.

Parameters

jobs	Jobs to add to this instance.

3.5.2.5 CM_JobManager CM_JobManager.ClearJobList ( )

Clears the job list owned by this instance. It does not kill the jobs so they will continue to run. Raises allJobsCleared event.

Returns

The job list.

3.5.2.6 bool CM_JobManager.HasJob (string *id*)

Determines whether this instance has the job with the specified id.

Returns

`true` if this instance has a job with the specified id; otherwise, `false`.

Parameters

id	Identifier.

3.5.2.7 bool CM_JobManager.IsRunning (CM_Job *job*)

Determines whether the specified job is executing.

Returns

`true` if the job is currently running; otherwise, `false`.

Parameters

job	Job.

3.5.2.8 bool CM_JobManager.IsRunning (string *id*)

Determines whether the specified job is executing.

Returns

`true` if the job is currently running; otherwise, `false`.

Parameters

job	Job ID.

3.5.2.9 CM_JobManager CM_JobManager.KillAll ( )

Kills all jobs owned by this instance. Raises allJobsKilled event.

Returns

The all.

3.5.2.10 CM_JobManager CM_JobManager.KillAll (float *delayInSeconds*)

Kills all jobs owned by this instance after delay in seconds. Raises allJobsKilled event.

Returns

The all.

Parameters

delayInSeconds	Delay in seconds.

3.5.2.11 static CM_JobManager CM_JobManager.Make ( ) [static]

Returns an initialised CM_JobManager instance. Provides static access to class.

3.5.2.12 CM_JobManager CM_JobManager.NotifyOnAllJobsCleared (EventHandler< CM_JobManagerEventArgs > *e*)

Subscribes to the the allJobsCleared event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.13 CM_JobManager CM_JobManager.NotifyOnAllJobsKilled (EventHandler< CM_JobManagerEventArgs > *e*)

Subscribes to the the allJobsKilled event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.14 CM_JobManager CM_JobManager.NotifyOnAllJobsPaused (EventHandler< CM_JobManagerEventArgs > *e*)

Subscribes to the the jobPaused event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.15 CM_JobManager CM_JobManager.NotifyOnAllJobsResumed (EventHandler< CM_JobManagerEventArgs > *e*)

Subscribes to the the jobResumed event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.16 CM_JobManager CM_JobManager.NotifyOnJobAdded (EventHandler< CM_JobManagerJobEditedEventArgs > *e*)

Subscribes to the the jobAdded event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.17 CM_JobManager CM_JobManager.NotifyOnJobRemoved (EventHandler< CM_JobManagerJobEdited↩ EventArgs > *e*)

Subscribes to the the jobRemoved event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.18 void CM_JobManager.OnAllJobsCleared (CM_JobManagerEventArgs *e*)

Raises the all jobs cleared event.

Parameters

e	E.

3.5.2.19 void CM_JobManager.OnAllJobsKilled (CM_JobManagerEventArgs *e*)

Raises the all jobs killed event.

Parameters

e	E.

3.5.2.20 void CM_JobManager.OnAllJobsPaused (CM_JobManagerEventArgs *e*)

Raises the all jobs paused event.

Parameters

e	E.

3.5.2.21 void CM_JobManager.OnAllJobsResumed (CM_JobManagerEventArgs *e*)

Raises the all jobs resumed event.

Parameters

e	E.

3.5.2.22 void CM_JobManager.OnJobAdded (CM_JobManagerJobEditedEventArgs *e*) [protected]

Raises the job added event.

Parameters

e	E.

3.5.2.23 void CM_JobManager.OnJobRemoved (CM_JobManagerJobEditedEventArgs *e*)

Raises the job removed event.

Parameters

e	E.

3.5.2.24 CM_JobManager CM_JobManager.PauseAll ( )

Pauses all jobs owned by this instance. Raises allJobsPaused event.

Returns

The all.

3.5.2.25 CM_JobManager CM_JobManager.PauseAll (float *delayInSeconds*)

Pauses all jobs owned by this instance after delay in seconds. Raises allJobsPaused event.

Returns

The all.

Parameters

delayInSeconds	Delay in seconds.

3.5.2.26 CM_JobManager CM_JobManager.PauseCoroutine (CM_Job *job*)

Pauses the specified coroutine if owned by this instance of job manager. Job is searched using job id.

Returns

The job manager.

Parameters

job	Job.

3.5.2.27 CM_JobManager CM_JobManager.PauseCoroutine (string *id*)

Stops the specified coroutine if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.2.28 CM_JobManager CM_JobManager.RemoveJob (CM_Job *job*)

Removes the job if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.2.29 CM_JobManager CM_JobManager.RemoveJob (string *id*)

Removes the job if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.2.30 CM_JobManager CM_JobManager.RemoveNotifyOnAllJobsCleared (EventHandler< CM_JobManagerEventArgs > *e*)

Unsubscribes to the the allJobsCleared event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.31 CM_JobManager CM_JobManager.RemoveNotifyOnAllJobsKilled (EventHandler< CM_JobManagerEventArgs > *e*)

Unsubscribes to the the allJobsKilled event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.32 CM_JobManager CM_JobManager.RemoveNotifyOnAllJobsPaused (EventHandler< CM_JobManagerEventArgs > *e*)

Unsubscribes to the the jobPaused event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.33 CM_JobManager CM_JobManager.RemoveNotifyOnAllJobsResumed (EventHandler< CM_JobManagerEventArgs > *e*)

Unsubscribes to the the jobResumed event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.34 CM_JobManager CM_JobManager.RemoveNotifyOnJobAdded (EventHandler< CM_JobManagerJobEditedEventArgs > *e*)

Unsubscribes to the the jobAdded event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.35 CM_JobManager CM_JobManager.RemoveNotifyOnJobRemoved (EventHandler< CM_JobManagerJobEditedEventArgs > *e*)

Unsubscribes to the the jobRemoved event.

Parameters

e	The eventhandler to be invoked on event.

3.5.2.36 CM_JobManager CM_JobManager.ResumeAll ( )

Resumes all jobs owned by this instance. Raises allJobsResumedEvent.

Returns

The all.

3.5.2.37 CM_JobManager CM_JobManager.ResumeAll (float *delayInSeconds*)

Resumes all jobs owned by this instance after delay in seconds. Raises allJobsResumedEvent.

Returns

The all.

Parameters

delayInSeconds	Delay in seconds.

3.5.2.38 CM_JobManager CM_JobManager.ResumeCoroutine (CM_Job *job*)

Resumes the specified coroutine if owned by this instance of job manager. Job is searched using job id.

Returns

The job manager.

Parameters

job	Job.

3.5.2.39 CM_JobManager CM_JobManager.ResumeCoroutine (string *id*)

Stops the specified coroutine if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.2.40 CM_JobManager CM_JobManager.StartAll ( )

Starts all jobs owned by this instance.

Returns

The all.

3.5.2.41 CM_JobManager CM_JobManager.StartAll (float *delayInSeconds*)

Starts all jobs owned by this instance after delay in seconds.

Returns

The all.

Parameters

delayInSeconds	Delay in seconds.

3.5.2.42 CM_JobManager CM_JobManager.StartCoroutine (CM_Job *job*)

Starts the specified coroutine if owned by this instance of job manager. Job is searched using job id.

Returns

The job manager.

Parameters

job	Job.

3.5.2.43 CM_JobManager CM_JobManager.StartCoroutine (string *id*)

Starts the specified coroutine if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.2.44 CM_JobManager CM_JobManager.StopCoroutine (CM_Job *job*)

Stops the specified coroutine if owned by this instance of job manager. Job is searched using job id.

Returns

The job manager.

Parameters

job	Job.

3.5.2.45 CM_JobManager CM_JobManager.StopCoroutine (string *id*)

Stops the specified coroutine if owned by this instance of job manager.

Returns

The job manager.

Parameters

job	Job.

3.5.3 Event Documentation

3.5.3.1 EventHandler<CM_JobManagerEventArgs> CM_JobManager.allJobsCleared [protected]

Raised when all jobs cleared.

3.5.3.2 EventHandler<CM_JobManagerEventArgs> CM_JobManager.allJobsKilled [protected]

Raised when all jobs killed.

3.5.3.3 EventHandler<CM_JobManagerEventArgs> CM_JobManager.allJobsPaused [protected]

Raised when all jobs paused.

3.5.3.4 EventHandler<CM_JobManagerEventArgs> CM_JobManager.allJobsResumed [protected]

Raised when all jobs resumed.

3.5.3.5 EventHandler<CM_JobManagerJobEditedEventArgs> CM_JobManager.jobAdded [protected]

Raised when job added.

3.5.3.6 EventHandler<CM_JobManagerJobEditedEventArgs> CM_JobManager.jobRemoved [protected]

Raised when job removed.

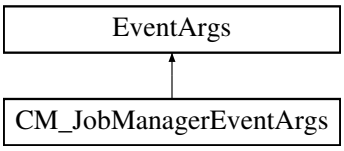
The documentation for this class was generated from the following file:

- CM_JobManager.cs

3.6 CM_JobManagerEventArgs Class Reference

Arguements used in events raised by CM_JobManager.

Inheritance diagram for CM_JobManagerEventArgs:



Public Member Functions

- CM_JobManagerEventArgs (Dictionary< string, CM_Job > ownedJobs)

  Initializes a new instance of the CM_JobManagerEventArgs class.

Properties

- CM_Job[ ] ownedJobs `[get]`

  Gets the owned jobs of the job manager at the time of the event being raised.
- CM_Job[ ] runningJobs `[get]`

  Gets the currently running jobs of the job manager at the time of the event being raised.
- CM_Job[ ] pausedJobs `[get]`

  Gets the paused jobs of the job manager at the time of the event being raised.

3.6.1 Detailed Description

Arguements used in events raised by CM_JobManager.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 CM_JobManagerEventArgs.CM_JobManagerEventArgs (Dictionary< string, CM_Job > *ownedJobs*)

Initializes a new instance of the CM_JobManagerEventArgs class.

Parameters

ownedJobs	Owned jobs.

3.6.3 Property Documentation

3.6.3.1 CM_Job [ ] CM_JobManagerEventArgs.ownedJobs `[get]`

Gets the owned jobs of the job manager at the time of the event being raised.

The owned jobs.

3.6.3.2 CM_Job [ ] CM_JobManagerEventArgs.pausedJobs `[get]`

Gets the paused jobs of the job manager at the time of the event being raised.

The paused jobs.

3.6.3.3 CM_Job [ ] CM_JobManagerEventArgs.runningJobs [get]

Gets the currently running jobs of the job manager at the time of the event being raised.

The running jobs.

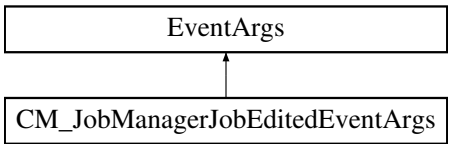
The documentation for this class was generated from the following file:

- CM_JobManagerEventArgs.cs

3.7 CM_JobManagerJobEditedEventArgs Class Reference

Arguements used in events raised by CM_JobManager.

Inheritance diagram for CM_JobManagerJobEditedEventArgs:



Public Member Functions

- CM_JobManagerJobEditedEventArgs (Dictionary< string, CM_Job > ownedJobs, CM_Job jobEdited)

  Initializes a new instance of the CM_JobManagerJobEditedEventArgs class.

Properties

- CM_JobManagerEventArgs otherArgs [get]

  Gets the other arguments.
- CM_Job jobEdited [get]

  Gets the job currently changed that caused the event to be raised.

3.7.1 Detailed Description

Arguements used in events raised by CM_JobManager.

3.7.2 Constructor & Destructor Documentation

3.7.2.1 CM_JobManagerJobEditedEventArgs.CM_JobManagerJobEditedEventArgs (Dictionary< string, CM_Job > *ownedJobs*, CM_Job *jobEdited*)

Initializes a new instance of the CM_JobManagerJobEditedEventArgs class.

Parameters

ownedJobs	Owned jobs.
jobEdited	Job edited.

3.7.3 Property Documentation

3.7.3.1 CM_Job CM_JobManagerJobEditedEventArgs.jobEdited [get]

Gets the job currently changed that caused the event to be raised.

The job edited.

3.7.3.2 CM_JobManagerEventArgs CM_JobManagerJobEditedEventArgs.otherArgs [get]

Gets the other arguments.

The other arguments.

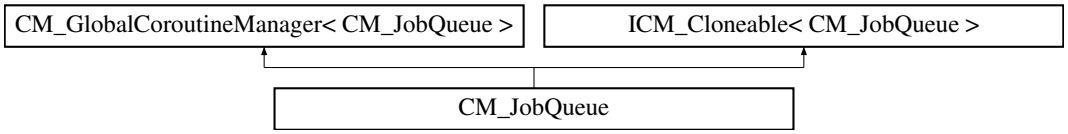
The documentation for this class was generated from the following file:

- CM_JobManagerJobEditedEventArgs.cs

3.8 CM_JobQueue Class Reference

The main job queue class. Encapsulates all behaviour related to queueing a job. Provides access to events, and status (i.e. running, repeating).

Inheritance diagram for CM_JobQueue:



Public Member Functions

- CM_JobQueue Clone ()

  Clone this instance.
- CM_JobQueue[ ] Clone (int numOfCopies)

  Clone this instance the specified numOfCopies.
- CM_JobQueue Enqueue (CM_JobQueue other)

  Enqueues the specified other queue. Adds the jobs from one queue to this queue and also adds the other queues event subscriptions.
- CM_JobQueue Enqueue (params CM_Job[ ] jobs)

  Enqueues the specified jobs.
- CM_JobQueue Enqueue (IList< CM_Job > jobs)

  Enqueues the specified jobs.
- CM_JobQueue Enqueue (string id, IEnumerator routine)

  Creates a new job with specified id and coroutine and adds job to queue.
- CM_JobQueue Enqueue (IEnumerator routine)

  Creates a new job with specified id and coroutine and adds job to queue.
- CM_JobQueue Enqueue (CM_Job job)

  Enqueues the specified job.
- CM_JobQueue Start ()

  Start this instance of the queue immediately.
- CM_JobQueue Start (float delayInSeconds)

Start the specified instance after delayInSeconds.

- CM_JobQueue Repeat ()

  Sets this instance to repeat. The job is repeated when it has finished processing.

- CM_JobQueue Repeat (int numOfTimes)

  Sets this instance to repeat a number of times. The job is repeated when it has finished processing.

- CM_JobQueue StopRepeat ()

  Stops the repeat.

- CM_JobQueue StopRepeat (float delayInSeconds)

  Stops the repeat after a specified delay in seconds.

- CM_JobQueue Pause ()

  Pauses this instance.

- CM_JobQueue Pause (float delayInSeconds)

  Pause this instance after the specified delayInSeconds.

- CM_JobQueue Resume ()

  Resume this instance immediately.

- CM_JobQueue Resume (float delayInSeconds)

  Resume the instance after the specified delayInSeconds.

- CM_JobQueue ContinousRunning ()

  Set the queue to run continously.

- CM_JobQueue StopContinousRunning ()

  Stops the continous running of this queue.

- CM_JobQueue KillAll ()

  Kill all currently queued jobs immediately. Clears queue list.

- CM_JobQueue KillAll (float delayInSeconds)

  Kill all currently queued jobs after the specified delayInSeconds.

- CM_JobQueue KillCurrent ()

  Kills the current running job immediately.

- CM_JobQueue KillCurrent (float delayInSeconds)

  Kills the current running job after the specified delayInSeconds.

- CM_JobQueue NotifyOnQueueStarted (EventHandler< CM_QueueEventArgs > e)

  Subscribes to the queue started event.

- CM_JobQueue RemoveNotifyOnQueueStarted (EventHandler< CM_QueueEventArgs > e)

  Unsubscribes to the the queue started event.

- CM_JobQueue NotifyOnQueueComplete (EventHandler< CM_QueueEventArgs > e)

  Subscribes to the queue completed event.

- CM_JobQueue RemoveNotifyOnQueueComplete (EventHandler< CM_QueueEventArgs > e)

  Unsubscribes to the queue completed event.

- CM_JobQueue NotifyOnJobProcessed (EventHandler< CM_QueueEventArgs > e)

  Subscribes to the the job processed event.

- CM_JobQueue RemoveNotifyOnJobProcessed (EventHandler< CM_QueueEventArgs > e)

  Unsubscribes to the the job processed event. This event is invoked every time a job in the queue has finished running.

Static Public Member Functions

- static CM_JobQueue Make ()

  Returns an initialised CM_JobQueue instance. Provides static access to class.

Protected Member Functions

- void OnQueueStarted (CM_QueueEventArgs e)

  Raises the queue started event.
- void OnQueueComplete (CM_QueueEventArgs e)

  Raises the queue complete event.
- void OnJobProcessed (CM_QueueEventArgs e)

  Raises the job processed event.
- override void HandlejobComplete (object sender, CM_JobEventArgs e)

  Invoked whenever a queued job has finished processing. Handles maintenance of queue and raising OnJob↩ Processed and OnQueueComplete events.

Properties

- bool repeating `[get]`

  Gets a value indicating whether this CM_JobQueue is repeating.
- int numOfTimesExecuted `[get]`

  Gets the number of times this queue executed (used if repeating).
- bool running `[get]`

  Gets a value indicating whether this CM_JobQueue is running.
- bool continousRunning `[get]`

  Gets a value indicating whether this CM_JobQueue is running continously i.e. will not stop running until Stop↩ ContinousRunning is called.

Events

- EventHandler< CM_QueueEventArgs > queueStarted

  Raised when queue started.
- EventHandler< CM_QueueEventArgs > queueComplete

  Raised when queue complete.
- EventHandler< CM_QueueEventArgs > jobProcessed

  Raised when a job in the queue has finished.

3.8.1 Detailed Description

The main job queue class. Encapsulates all behaviour related to queueing a job. Provides access to events, and status (i.e. running, repeating).

3.8.2 Member Function Documentation

3.8.2.1 CM_JobQueue CM_JobQueue.Clone ( )

Clone this instance.

3.8.2.2 CM_JobQueue [ ] CM_JobQueue.Clone (int *numOfCopies*)

Clone this instance the specified numOfCopies.

Parameters

numOfCopies	Number of copies.

3.8.2.3 CM_JobQueue CM_JobQueue.ContinousRunning ( )

Set the queue to run continously.

3.8.2.4 CM_JobQueue CM_JobQueue.Enqueue (CM_JobQueue *other*)

Enqueues the specified other queue. Adds the jobs from one queue to this queue and also adds the other queues event subscriptions.

Parameters

other	Other.

3.8.2.5 CM_JobQueue CM_JobQueue.Enqueue (params CM_Job[ ] *jobs*)

Enqueues the specified jobs.

Parameters

jobs	Jobs.

3.8.2.6 CM_JobQueue CM_JobQueue.Enqueue (IList< CM_Job > *jobs*)

Enqueues the specified jobs.

Parameters

jobs	Jobs.

3.8.2.7 CM_JobQueue CM_JobQueue.Enqueue (string *id,* IEnumerator *routine*)

Creates a new job with specified id and coroutine and adds job to queue.

Parameters

id	Job Identifier.
routine	Routine.

3.8.2.8 CM_JobQueue CM_JobQueue.Enqueue (IEnumerator *routine*)

Creates a new job with specified id and coroutine and adds job to queue.

Parameters

id	Job Identifier.
routine	Routine.

3.8.2.9 CM_JobQueue CM_JobQueue.Enqueue (CM_Job *job*)

Enqueues the specified job.

Parameters

job	Job.

3.8.2.10 override void CM_JobQueue.HandlejobComplete (object *sender,* CM_JobEventArgs *e*) [protected], [virtual]

Invoked whenever a queued job has finished processing. Handles maintenance of queue and raising OnJob↩ Processed and OnQueueComplete events.

Parameters

sender	Sender.
e	E.

Implements CM_GlobalCoroutineManager< CM_JobQueue >.

3.8.2.11 CM_JobQueue CM_JobQueue.KillAll ( )

Kill all currently queued jobs immediately. Clears queue list.

3.8.2.12 CM_JobQueue CM_JobQueue.KillAll (float *delayInSeconds*)

Kill all currently queued jobs after the specified delayInSeconds.

Parameters

delayInSeconds	Delay in seconds.

3.8.2.13 CM_JobQueue CM_JobQueue.KillCurrent ( )

Kills the current running job immediately.

Returns

The current.

3.8.2.14 CM_JobQueue CM_JobQueue.KillCurrent (float *delayInSeconds*)

Kills the current running job after the specified delayInSeconds.

Returns

The current.

Parameters

delayInSeconds	Delay in seconds.

3.8.2.15 static CM_JobQueue CM_JobQueue.Make ( ) [static]

Returns an initialised CM_JobQueue instance. Provides static access to class.

3.8.2.16 CM_JobQueue CM_JobQueue.NotifyOnJobProcessed (EventHandler< CM_QueueEventArgs > *e*)

Subscribes to the the job processed event.

Parameters

e	The event handler to be invoked on event.

3.8.2.17 CM_JobQueue CM_JobQueue.NotifyOnQueueComplete (EventHandler< CM_QueueEventArgs > *e*)

Subscribes to the queue completed event.

Parameters

e	The event handler to be invoked on event.

3.8.2.18 CM_JobQueue CM_JobQueue.NotifyOnQueueStarted (EventHandler< CM_QueueEventArgs > *e*)

Subscribes to the queue started event.

Parameters

e	The event handler to be invoked on event.

3.8.2.19 void CM_JobQueue.OnJobProcessed (CM_QueueEventArgs *e*) [protected]

Raises the job processed event.

Parameters

e	E.

3.8.2.20 void CM_JobQueue.OnQueueComplete (CM_QueueEventArgs *e*) [protected]

Raises the queue complete event.

Parameters

e	E.

3.8.2.21 void CM_JobQueue.OnQueueStarted (CM_QueueEventArgs *e*) [protected]

Raises the queue started event.

Parameters

e	E.

3.8.2.22 CM_JobQueue CM_JobQueue.Pause ( )

Pauses this instance.

3.8.2.23 CM_JobQueue CM_JobQueue.Pause (float *delayInSeconds*)

Pause this instance after the specified delayInSeconds.

Parameters

delayInSeconds	Delay in seconds.

3.8.2.24 CM_JobQueue CM_JobQueue.RemoveNotifyOnJobProcessed (EventHandler< CM_QueueEventArgs > *e*)

Unsubscribes to the the job processed event. This event is invoked every time a job in the queue has finished running.

Parameters

e	The event handler to be invoked on event.

3.8.2.25 CM_JobQueue CM_JobQueue.RemoveNotifyOnQueueComplete (EventHandler< CM_QueueEventArgs > *e*)

Unsubscribes to the queue completed event.

Parameters

e	The event handler to be invoked on event.

3.8.2.26 CM_JobQueue CM_JobQueue.RemoveNotifyOnQueueStarted (EventHandler< CM_QueueEventArgs > *e*)

Unsubscribes to the the queue started event.

Parameters

e	The event handler to be invoked on event.

3.8.2.27 CM_JobQueue CM_JobQueue.Repeat ( )

Sets this instance to repeat. The job is repeated when it has finished processing.

3.8.2.28 CM_JobQueue CM_JobQueue.Repeat (int *numOfTimes*)

Sets this instance to repeat a number of times. The job is repeated when it has finished processing.

3.8.2.29 CM_JobQueue CM_JobQueue.Resume ( )

Resume this instance immediately.

3.8.2.30 CM_JobQueue CM_JobQueue.Resume (float *delayInSeconds*)

Resume the instance after the specified delayInSeconds.

Parameters

delayInSeconds	Delay in seconds.

3.8.2.31 CM_JobQueue CM_JobQueue.Start ( )

Start this instance of the queue immediately.

3.8.2.32 CM_JobQueue CM_JobQueue.Start (float *delayInSeconds*)

Start the specified instance after delayInSeconds.

Parameters

delayInSeconds	Delay in seconds.

3.8.2.33 CM_JobQueue CM_JobQueue.StopContinousRunning ( )

Stops the continous running of this queue.

3.8.2.34 CM_JobQueue CM_JobQueue.StopRepeat ( )

Stops the repeat.

Returns

The repeat.

3.8.2.35 CM_JobQueue CM_JobQueue.StopRepeat (float *delayInSeconds*)

Stops the repeat after a specified delay in seconds.

Returns

The repeat.

Parameters

delayInSeconds	Delay in seconds.

3.8.3 Property Documentation

3.8.3.1 bool CM_JobQueue.continousRunning [get]

Gets a value indicating whether this CM_JobQueue is running continously i.e. will not stop running until StopContinousRunning is called.

`true` if continous running; otherwise, `false`.

3.8.3.2 int CM_JobQueue.numOfTimesExecuted [get]

Gets the number of times this queue executed (used if repeating).

The number of times executed.

3.8.3.3 bool CM_JobQueue.repeating [get]

Gets a value indicating whether this CM_JobQueue is repeating.

`true` if repeating; otherwise, `false`.

3.8.3.4 bool CM_JobQueue.running [get]

Gets a value indicating whether this CM_JobQueue is running.

`true` if running; otherwise, `false`.

3.8.4 Event Documentation

3.8.4.1 EventHandler<CM_QueueEventArgs> CM_JobQueue.jobProcessed [protected]

Raised when a job in the queue has finished.

3.8.4.2 EventHandler<CM_QueueEventArgs> CM_JobQueue.queueComplete [protected]

Raised when queue complete.

3.8.4.3 EventHandler<CM_QueueEventArgs> CM_JobQueue.queueStarted [protected]

Raised when queue started.

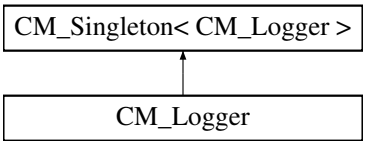
The documentation for this class was generated from the following file:

- CM_JobQueue.cs

3.9 CM_Logger Class Reference

Simple logging class used by the Coroutine Manager.

Inheritance diagram for CM_Logger:



Classes

- class Message

  Encapsulates a message used by CM_Logger.

Public Types

- enum Status { **Log**, **Warning**, **Error** }

  The status level of the message.

Public Member Functions

- void Log (object message)

  Log the specified message with default log status.
- void Log (object context, object message)

  Log the message with the specified context.
- void Log (object message, Status status)

  Uses a lookup to either write a log, warning, or error based on the status.

Additional Inherited Members

3.9.1 Detailed Description

Simple logging class used by the Coroutine Manager.

3.9.2 Member Enumeration Documentation

3.9.2.1 enum CM_Logger.Status [strong]

The status level of the message.

3.9.3 Member Function Documentation

3.9.3.1 void CM_Logger.Log (object *message*)

Log the specified message with default log status.

Parameters

message	Message.

3.9.3.2 void CM_Logger.Log (object *context*, object *message*)

Log the message with the specified context.

Parameters

context	Context i.e. the calling class.
message	Message.

3.9.3.3 void CM_Logger.Log (object *message*, Status *status*)

Uses a lookup to either write a log, warning, or error based on the status.

Parameters

message	Message.
status	Status.

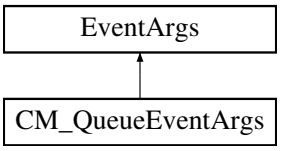
The documentation for this class was generated from the following file:

- CM_Logger.cs

3.10 CM_QueueEventArgs Class Reference

Arguements used by events raised by CM_JobQueue.

Inheritance diagram for CM_QueueEventArgs:



Public Member Functions

- CM_QueueEventArgs (CM_Job[ ] queuedJobs, CM_Job[ ] completedJobs, CM_JobQueue jobQueue)

  Initializes a new instance of the CM_QueueEventArgs class.

Properties

- bool hasJobsInQueue `[get]`

  Gets a value indicating whether this CM_QueueEventArgs has jobs in queue.
- CM_Job[ ] queuedJobs `[get]`

  Gets the queued jobs.
- bool hasCompletedJobs `[get]`

  Gets a value indicating whether this CM_QueueEventArgs has completed jobs.
- CM_Job[ ] completedJobs `[get]`

  Gets the completed jobs.
- CM_JobQueue **jobQueue** `[get]`

3.10.1 Detailed Description

Arguements used by events raised by CM_JobQueue.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 CM_QueueEventArgs.CM_QueueEventArgs (CM_Job[ ] *queuedJobs*, CM_Job[ ] *completedJobs*, CM_JobQueue *jobQueue*)

Initializes a new instance of the CM_QueueEventArgs class.

Parameters

queuedJobs	Queued jobs.
completedJobs	Completed jobs.

3.10.3 Property Documentation

3.10.3.1 CM_Job [ ] CM_QueueEventArgs.completedJobs `[get]`

Gets the completed jobs.

The completed jobs.

3.10.3.2 bool CM_QueueEventArgs.hasCompletedJobs `[get]`

Gets a value indicating whether this CM_QueueEventArgs has completed jobs.

`true` if has completed jobs; otherwise, `false`.

3.10.3.3 bool CM_QueueEventArgs.hasJobsInQueue [get]

Gets a value indicating whether this CM_QueueEventArgs has jobs in queue.

`true` if has jobs in queue; otherwise, `false`.

3.10.3.4 CM_Job [ ] CM_QueueEventArgs.queuedJobs [get]

Gets the queued jobs.

The queued jobs.

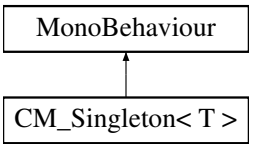
The documentation for this class was generated from the following file:

- CM_QueueEventArgs.cs

3.11 CM_Singleton< T > Class Template Reference

A base class for any Singleton. Provides global singular access to a MonoBehaviour.

Inheritance diagram for CM_Singleton< T >:



Protected Member Functions

- virtual void **OnDestroy** ()
- virtual void **OnApplicationQuit** ()

Properties

- static bool IsDestroyed [get]
  Gets a value indicating whether this instance is destroyed.
- static T instance [get]
  Gets the instance. The instance is created if not currently past of the scene.

3.11.1 Detailed Description

A base class for any Singleton. Provides global singular access to a MonoBehaviour.

Type Constraints

T* : *MonoBehaviour

3.11.2 Property Documentation

3.11.2.1 T CM_Singleton< T >.instance [static], [get]

Gets the instance. The instance is created if not currently past of the scene.

The instance.

3.11.2.2 bool CM_Singleton< T >.IsDestroyed [static],[get]

Gets a value indicating whether this instance is destroyed.

`true` if is destroyed; otherwise, `false`.

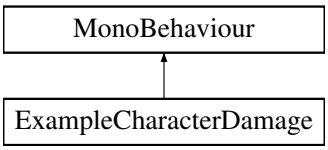
The documentation for this class was generated from the following file:

- CM_Singleton.cs

3.12 ExampleCharacterDamage Class Reference

Example character with action queue.

Inheritance diagram for ExampleCharacterDamage:



Public Member Functions

- void AddActionToQueue (CM_Job action)

  Adds the action to the current queue. An action can be any coroutine you want.
- IEnumerator ApplyDamage (string damageType, float time)

  Simulates the application of damage over time.
- IEnumerator RestoreHealth (float time)

  Simulates restoring of health over time.

Public Attributes

- float **health** = 20f

3.12.1 Detailed Description

Example character with action queue.

3.12.2 Member Function Documentation

3.12.2.1 void ExampleCharacterDamage.AddActionToQueue (CM_Job *action*)

Adds the action to the current queue. An action can be any coroutine you want.

Parameters

action	Action.

3.12.2.2 IEnumerator ExampleCharacterDamage.ApplyDamage (string *damageType*, float *time*)

Simulates the application of damage over time.

Returns

The damage.

Parameters

damageType	Damage type.
time	Time.

3.12.2.3 IEnumerator ExampleCharacterDamage.RestoreHealth (float *time*)

Simulates restoring of health over time.

Returns

The health.

Parameters

time	Time.

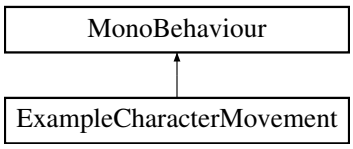
The documentation for this class was generated from the following file:

- ExampleCharacterDamage.cs

3.13 ExampleCharacterMovement Class Reference

Example Script. A simple script showing how you can use CM_JobQueue to create easily repeatable character movement.

Inheritance diagram for ExampleCharacterMovement:



Public Attributes

- float moveSpeed = 10f

  The movement speed of the character.

3.13.1 Detailed Description

Example Script. A simple script showing how you can use CM_JobQueue to create easily repeatable character movement.

3.13.2 Member Data Documentation

3.13.2.1 float ExampleCharacterMovement.moveSpeed = 10f

The movement speed of the character.

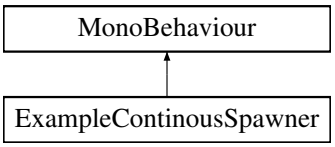
The documentation for this class was generated from the following file:

- ExampleCharacterMovement.cs

3.14 ExampleContinousSpawner Class Reference

Example Script. Used to spawn a number of objects using CM_Job.

Inheritance diagram for ExampleContinousSpawner:



Public Attributes

- GameObject prefab

  The prefab to spawn.
- int numToSpawn = 200

  The number of objects to spawn.
- float timeBetweenSpawns = 0.1f

  The time between spawns.

3.14.1 Detailed Description

Example Script. Used to spawn a number of objects using CM_Job.

3.14.2 Member Data Documentation

3.14.2.1 int ExampleContinousSpawner.numToSpawn = 200

The number of objects to spawn.

3.14.2.2 GameObject ExampleContinousSpawner.prefab

The prefab to spawn.

3.14.2.3 float ExampleContinousSpawner.timeBetweenSpawns = 0.1f

The time between spawns.

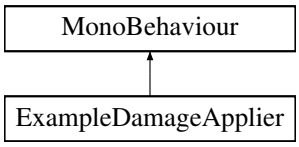
The documentation for this class was generated from the following file:

- ExampleContinousSpawner.cs

3.15 ExampleDamageApplier Class Reference

Applies damage actions to the example character.

Inheritance diagram for ExampleDamageApplier:



Public Attributes

- ExampleCharacterDamage character

  The character to apply damage actions to.

3.15.1 Detailed Description

Applies damage actions to the example character.

3.15.2 Member Data Documentation

3.15.2.1 ExampleCharacterDamage ExampleDamageApplier.character

The character to apply damage actions to.

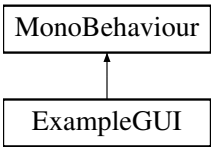
The documentation for this class was generated from the following file:

- ExampleDamageApplier.cs

3.16 ExampleGUI Class Reference

Simple GUI controller. Uses a CM_JobQueue to enqueue a number of gui actions.

Inheritance diagram for ExampleGUI:



Public Attributes

- Text titleText

  THe GUI text

3.16.1 Detailed Description

Simple GUI controller. Uses a CM_JobQueue to enqueue a number of gui actions.

3.16.2 Member Data Documentation

3.16.2.1 Text ExampleGUI.titleText

THe GUI text

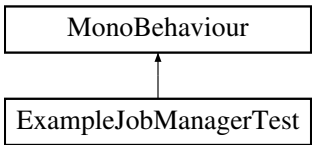
The documentation for this class was generated from the following file:

- ExampleGUI.cs

3.17 ExampleJobManagerTest Class Reference

Includes a number of methods to test the functionality of the CM_JobManager class. Each method showcases a particular functionality of the CM_JobManager that you can implement in your own projects.

Inheritance diagram for ExampleJobManagerTest:



Public Member Functions

- void GlobalJobManagerEventTest ()

  Subscribes to each of the global managers events. NotifyOnJobAdded: called everytime a job is added to the global manager, NotifyOnJobRemoved: called everytime a job is removed, NotifyOnAllJobsKilled: called when CM_JobManager.KillAll is invoked, NotifyOnAllJobsResumed: called when CM_JobManager.ResumeAll is called, NotifyOnAllJobsPaused: called when CM_JobManager.PauseAll is invoked, and NotifyOnAllJobsCleared: called when CM_JobManager.ClearJobList is invoked.
- void GlobalJobStartTest ()

  Starts the test job.
- void GlobalJobPauseTest ()

  Pauses the test job.
- void GlobalJobResumeTest ()

  Resumes the test job.
- void GlobalJobStopTest ()

  Stops the test job. This also removes the reference from the JobManager.
- void GlobalStartAllTest ()

  Starts all jobs owned by the global JobManager.
- void DelayedPauseAllTest ()

  Pauses all jobs owned by the global JobManager after 1 second has passed.
- void DelayedResumeAllTest ()

  Resumes all jobs owned by the global JobManager after 1.5 seconds have passed.
- void DelayedKillAllTest ()

  Kills all jobs owned by the global JobManager after 2 seconds have passed. This also removes all references to those jobs from the JobManager.
- void LocalJobManagerTest ()

  Local job manager test. Creates a new local JobManager, subscribes to CM_JobManager.jobAdded and CM_JobManager.jobRemoved events, adds a test job to the local JobManager, and finally starts, pauses, resumes, and stops this test job. This is used to show that anything you can do with the glocal JobManager you can also do with a local JobManager. This is useful if you want to create seperate JobManagers for seperate parts of your codebase.

3.17.1 Detailed Description

Includes a number of methods to test the functionality of the CM_JobManager class. Each method showcases a particular functionality of the CM_JobManager that you can implement in your own projects.

3.17.2 Member Function Documentation

3.17.2.1 void ExampleJobManagerTest.DelayedKillAllTest ( )

Kills all jobs owned by the global JobManager after 2 seconds have passed. This also removes all references to those jobs from the JobManager.

3.17.2.2 void ExampleJobManagerTest.DelayedPauseAllTest ( )

Pauses all jobs owned by the global JobManager after 1 second has passed.

3.17.2.3 void ExampleJobManagerTest.DelayedResumeAllTest ( )

Resumes all jobs owned by the global JobManager after 1.5 seconds have passed.

3.17.2.4 void ExampleJobManagerTest.GlobalJobManagerEventTest ( )

Subscribes to each of the global managers events. NotifyOnJobAdded: called everytime a job is added to the global manager, NotifyOnJobRemoved: called everytime a job is removed, NotifyOnAllJobsKilled: called when CM_JobManager.KillAll is invoked, NotifyOnAllJobsResumed: called when CM_JobManager.ResumeAll is called, NotifyOnAllJobsPaused: called when CM_JobManager.PauseAll is invoked, and NotifyOnAllJobsCleared: called when CM_JobManager.ClearJobList is invoked.

3.17.2.5 void ExampleJobManagerTest.GlobalJobPauseTest ( )

Pauses the test job.

3.17.2.6 void ExampleJobManagerTest.GlobalJobResumeTest ( )

Resumes the test job.

3.17.2.7 void ExampleJobManagerTest.GlobalJobStartTest ( )

Starts the test job.

3.17.2.8 void ExampleJobManagerTest.GlobalJobStopTest ( )

Stops the test job. This also removes the reference from the JobManager.

3.17.2.9 void ExampleJobManagerTest.GlobalStartAllTest ( )

Starts all jobs owned by the global JobManager.

3.17.2.10 void ExampleJobManagerTest.LocalJobManagerTest ( )

Local job manager test. Creates a new local JobManager, subscribes to CM_JobManager.jobAdded and CM_↩ JobManager.jobRemoved events, adds a test job to the local JobManager, and finally starts, pauses, resumes, and stops this test job. This is used to show that anything you can do with the glocal JobManager you can also do with a local JobManager. This is useful if you want to create seperate JobManagers for seperate parts of your codebase.

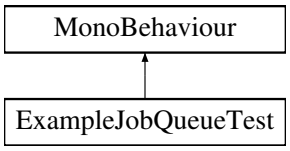
The documentation for this class was generated from the following file:

- ExampleJobManagerTest.cs

3.18 ExampleJobQueueTest Class Reference

Includes a number of methods to test the functionality of the CM_JobQueue class. Each method showcases a particular functionality of the CM_JobQueue that you can implement in your own projects. Each method returns an ienumerator so that it can added to a seperate job queue to be run in sequence for test purposes.

Inheritance diagram for ExampleJobQueueTest:



Public Member Functions

- IEnumerator SimpleGlobalQueueTest ()

  Creates a list of jobs, adds them to the global queue and starts global queue. The global queue can be accessed from any script.
- IEnumerator SimpleLocalQueueTest ()

  Creates a list of jobs, adds them to a newly created local queue and starts the queue.
- IEnumerator LocalQueueDelayedStartTest ()

  Creates and starts a queue after a delay of two seconds.
- IEnumerator LocalQueueDelayedPauseAndResumeTest ()

  Creates a local queue, starts queue, pauses queue after 1 seconds, and lastly resumes queue after 3 seconds.
- IEnumerator QueueEventTest ()

  Subscribes to the global queue "queueStarted", "jobProcessed", and "queueComplete" events.
- IEnumerator DelayedKillCurrentTest ()

  Creates a local queue and sets the current job in the queue to be killed after three seconds. This kills the currently running job but the queue still executes.
- IEnumerator DelayedKillAllTest ()

  Creates a local queue and sets all jobs in the queue to be killed after one second. This kills all jobs and clears the queue.
- IEnumerator AddLocalQueueToGlobalQueueTest ()

  Creates a local queue and then adds it to the glocal queue. The event subscriptions are also added to the global queue.
- IEnumerator AddRepeatingJobToQueueTest ()

  Adds a repeating job to a queue. Queue will not progress if a repeating job is added until that job is manually killed or has reached its set number of times to repeat.
- IEnumerator SetNumberRepeatingQueueTest ()

  Creates a new local queue, adds a number of test jobs and sets the queue to repeat two times.
- IEnumerator TimedRepeatingQueueTest ()

Creates a new local queue, sets it to repeat and then stop repeating after 1 seconds.

- IEnumerator ClonedRepeatingQueueTest ()

  Creates a local queue and sets it to repeat twice. The queue is then cloned and started. The new cloned queue will contain the original queues repeat status and event subscriptions.

- IEnumerator MultipleClonedQueueTest ()

  Creates a local queue and clones the queue twice. Both of the cloned queues are then started.

3.18.1 Detailed Description

Includes a number of methods to test the functionality of the CM_JobQueue class. Each method showcases a particular functionality of the CM_JobQueue that you can implement in your own projects. Each method returns an ienumerator so that it can added to a seperate job queue to be run in sequence for test purposes.

3.18.2 Member Function Documentation

3.18.2.1 IEnumerator ExampleJobQueueTest.AddLocalQueueToGlobalQueueTest ( )

Creates a local queue and then adds it to the glocal queue. The event subscriptions are also added to the global queue.

3.18.2.2 IEnumerator ExampleJobQueueTest.AddRepeatingJobToQueueTest ( )

Adds a repeating job to a queue. Queue will not progress if a repeating job is added until that job is manually killed or has reached its set number of times to repeat.

3.18.2.3 IEnumerator ExampleJobQueueTest.ClonedRepeatingQueueTest ( )

Creates a local queue and sets it to repeat twice. The queue is then cloned and started. The new cloned queue will contain the original queues repeat status and event subscriptions.

3.18.2.4 IEnumerator ExampleJobQueueTest.DelayedKillAllTest ( )

Creates a local queue and sets all jobs in the queue to be killed after one second. This kills all jobs and clears the queue.

3.18.2.5 IEnumerator ExampleJobQueueTest.DelayedKillCurrentTest ( )

Creates a local queue and sets the current job in the queue to be killed after three seconds. This kills the currently running job but the queue still executes.

3.18.2.6 IEnumerator ExampleJobQueueTest.LocalQueueDelayedPauseAndResumeTest ( )

Creates a local queue, starts queue, pauses queue after 1 seconds, and lastly resumes queue after 3 seconds.

3.18.2.7 IEnumerator ExampleJobQueueTest.LocalQueueDelayedStartTest ( )

Creates and starts a queue after a delay of two seconds.

3.18.2.8 IEnumerator ExampleJobQueueTest.MultipleClonedQueueTest ( )

Creates a local queue and clones the queue twice. Both of the cloned queues are then started.

3.18.2.9 IEnumerator ExampleJobQueueTest.QueueEventTest ( )

Subscribes to the global queue "queueStarted", "jobProcessed", and "queueComplete" events.

3.18.2.10 IEnumerator ExampleJobQueueTest.SetNumberRepeatingQueueTest ( )

Creates a new local queue, adds a number of test jobs and sets the queue to repeat two times.

3.18.2.11 IEnumerator ExampleJobQueueTest.SimpleGlobalQueueTest ( )

Creates a list of jobs, adds them to the global queue and starts global queue. The global queue can be accessed from any script.

3.18.2.12 IEnumerator ExampleJobQueueTest.SimpleLocalQueueTest ( )

Creates a list of jobs, adds them to a newly created local queue and starts the queue.

3.18.2.13 IEnumerator ExampleJobQueueTest.TimedRepeatingQueueTest ( )

Creates a new local queue, sets it to repeat and then stop repeating after 1 seconds.

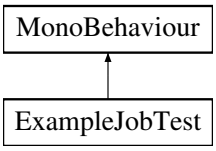
The documentation for this class was generated from the following file:

- ExampleJobQueueTest.cs

3.19 ExampleJobTest Class Reference

Includes a number of methods to test the functionality of the CM_Job class. Each method showcases a particular functionality of the CM_Job class that you can implement in your own projects. Each method returns an ienumerator so that it can added to a job queue to be run in sequence for test purposes.

Inheritance diagram for ExampleJobTest:



Public Member Functions

- IEnumerator SimpleJobTest ()

  Creates and starts a job.
- IEnumerator JobTestWithDelayedStart ()

  Creates and starts a job after a delay.
- IEnumerator JobTestWithDelayedPause ()

  Creates coroutine job, starts job immediately and then pauses coroutine after 4 seconds. This paused job is then stored and can be resumed/killed from any class that has a reference to that job.
- IEnumerator JobTestWithDelayedResume ()

  Creates new job, pauses after 1 second, and resumes after 3 seconds.
- IEnumerator JobTestWithDelayedKill ()

Creates coroutine job, starts job immediately and then sets the job to be killed after 4 seconds.

- IEnumerator JobTestWithStartAndEndEvents ()

  Creates new job, subscribes to the job start and end events and then starts job to be run immediately.

- IEnumerator ChildJobTest ()

  Child job test. Creates new job, adds two children (parent will not complete until children have finished processing) and starts the job.

- IEnumerator SingleCloneJobTest ()

  Creates a new job, clones job, and then runs clone.

- IEnumerator MultipleCloneJobTest ()

  Creates a new job, clones five copies of the original job, and then starts a random clone.

- IEnumerator InfinitelyRepeatableJobTest ()

  Creates and starts a job that will repeat for 3 seconds. The job complete event is subscribed to, this is used to display how many times the job will be repeated.

- IEnumerator MultipleRepeatableJobTest ()

  Creates and starts a job that will repeat three times. The job complete event is subscribed to, this is used to display how many times the job will be repeated.

- IEnumerator MutltipleRepeatableJobTestWithChild ()

  Creates and starts a job that will repeat three times. Adds a child job that will also repeat three times. The job complete event and child job complete events are subscribed to, this is used to display how many times the job will be repeated.

3.19.1 Detailed Description

Includes a number of methods to test the functionality of the CM_Job class. Each method showcases a particular functionality of the CM_Job class that you can implement in your own projects. Each method returns an ienumerator so that it can added to a job queue to be run in sequence for test purposes.

3.19.2 Member Function Documentation

3.19.2.1 IEnumerator ExampleJobTest.ChildJobTest ( )

Child job test. Creates new job, adds two children (parent will not complete until children have finished processing) and starts the job.

3.19.2.2 IEnumerator ExampleJobTest.InfinitelyRepeatableJobTest ( )

Creates and starts a job that will repeat for 3 seconds. The job complete event is subscribed to, this is used to display how many times the job will be repeated.

3.19.2.3 IEnumerator ExampleJobTest.JobTestWithDelayedKill ( )

Creates coroutine job, starts job immediately and then sets the job to be killed after 4 seconds.

3.19.2.4 IEnumerator ExampleJobTest.JobTestWithDelayedPause ( )

Creates coroutine job, starts job immediately and then pauses coroutine after 4 seconds. This paused job is then stored and can be resumed/killed from any class that has a reference to that job.

3.19.2.5 IEnumerator ExampleJobTest.JobTestWithDelayedResume ( )

Creates new job, pauses after 1 second, and resumes after 3 seconds.

3.19.2.6 IEnumerator ExampleJobTest.JobTestWithDelayedStart ( )

Creates and starts a job after a delay.

3.19.2.7 IEnumerator ExampleJobTest.JobTestWithStartAndEndEvents ( )

Creates new job, subscribes to the job start and end events and then starts job to be run immediately.

3.19.2.8 IEnumerator ExampleJobTest.MultipleCloneJobTest ( )

Creates a new job, clones five copies of the original job, and then starts a random clone.

3.19.2.9 IEnumerator ExampleJobTest.MultipleRepeatableJobTest ( )

Creates and starts a job that will repeat three times. The job complete event is subscribed to, this is used to display how many times the job will be repeated.

3.19.2.10 IEnumerator ExampleJobTest.MutltipleRepeatableJobTestWithChild ( )

Creates and starts a job that will repeat three times. Adds a child job that will also repeat three times. The job complete event and child job complete events are subscribed to, this is used to display how many times the job will be repeated.

3.19.2.11 IEnumerator ExampleJobTest.SimpleJobTest ( )

Creates and starts a job.

3.19.2.12 IEnumerator ExampleJobTest.SingleCloneJobTest ( )

Creates a new job, clones job, and then runs clone.

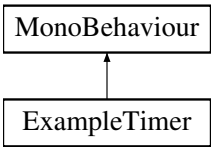
The documentation for this class was generated from the following file:

- ExampleJobTest.cs

3.20 ExampleTimer Class Reference

Creates new coroutine job to update a text object with time since startup. Adds job to global job manager so that it can be paused and resumed as required.

Inheritance diagram for ExampleTimer:



3.20.1 Detailed Description

Creates new coroutine job to update a text object with time since startup. Adds job to global job manager so that it can be paused and resumed as required.

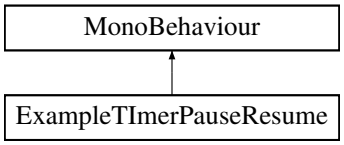
The documentation for this class was generated from the following file:

- ExampleTimer.cs

3.21 ExampleTImerPauseResume Class Reference

Pauses all jobs associated with the JobManager when user clicks left mouse button and resumes all jobs with user clicks right mouse button.

Inheritance diagram for ExampleTImerPauseResume:



3.21.1 Detailed Description

Pauses all jobs associated with the JobManager when user clicks left mouse button and resumes all jobs with user clicks right mouse button.

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

- ExampleTImerPauseResume.cs

3.22 ICM_Cloneable< T > Interface Template Reference

An interface for all classes used by the coroutine manager that can be cloned.

Public Member Functions

- T Clone ()

  Clone this instance.
- T[ ] Clone (int numOfCopies)

  Clone the specified numOfCopies.

3.22.1 Detailed Description

An interface for all classes used by the coroutine manager that can be cloned.

3.22.2 Member Function Documentation

3.22.2.1 T ICM_Cloneable< T >.Clone ( )

Clone this instance.

3.22.2.2 T [ ] ICM_Cloneable< T >.Clone (int *numOfCopies*)

Clone the specified numOfCopies.

Parameters

numOfCopies	Number of copies.

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

- ICM_Cloneable.cs

3.23 CM_Logger.Message Class Reference

Encapsulates a message used by CM_Logger.

Public Member Functions

- Message (object invoker, object content)

  Initializes a new instance of the CM_Logger+Message class.
- override string ToString ()

  Returns a System.String that represents the current CM_Logger+Message.

Properties

- object content `[get]`

  Gets the content of the message.
- object invoker `[get]`

  Gets the invoker of the message.

3.23.1 Detailed Description

Encapsulates a message used by CM_Logger.

3.23.2 Constructor & Destructor Documentation

3.23.2.1 CM_Logger.Message.Message (object *invoker,* object *content*)

Initializes a new instance of the CM_Logger+Message class.

Parameters

invoker	Invoker.
content	Content.

3.23.3 Member Function Documentation

3.23.3.1 override string CM_Logger.Message.ToString ( )

Returns a System.String that represents the current CM_Logger+Message.

Returns

A System.String that represents the current CM_Logger+Message.

3.23.4 Property Documentation

3.23.4.1 object CM_Logger.Message.content [get]

Gets the content of the message.

The content.

3.23.4.2 object CM_Logger.Message.invoker [get]

Gets the invoker of the message.

The invoker.

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

- CM_Logger.cs

Index