# Dessert

3.0.11

Generated by Doxygen 1.8.10

Sun Feb 28 2016 09:19:28

# Contents

# Chapter 1

# Namespace Index

## 1.1 Packages

Here are the packages with brief descriptions (if available):

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

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

# Chapter 3

# Class Index

## 3.1 Class List

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

# Chapter 4

# Namespace Documentation

## 4.1 DIBRIS Namespace Reference

### Namespaces

- namespace Dessert

## 4.2 DIBRIS.Dessert Namespace Reference

### Namespaces

- namespace Core
- namespace Events
- namespace Recording
- namespace Resources

### Classes

- class InterruptUncaughtException
- class PreemptionInfo
- class **Sim**
- class SimEnvironment
- class SimEvent

  *The interface common to each event; it should be used to declare generator methods.*
- class SimProcess

### Enumerations

- enum **TimeUnit** : byte {
  **Nanosecond**, **Microsecond**, **Millisecond**, **Second**,
  **Minute**, **Hour**, **Day** }

## 4.3 DIBRIS.Dessert.Core Namespace Reference

### Classes

- class **ConditionEvaluators**

- class **Default**
- class DessertException
- class **ErrorMessages**
- class FakeReadOnlyList
- class OptimizedSkewHeap
- class SimEntity

  *Represents an entity that belongs to a specific environment. An entity can only be "used" in the environment it belongs to.*
- class SimEvent

  *A stronger typed event, which adds type notation to many properties which are untyped in SimPy.*

## 4.4 DIBRIS.Dessert.Events Namespace Reference

### Classes

- class Call
- class Condition
- interface IInternalCall
- class InnerEvent

  *Represents an event which cannot be "yielded" by any user process. It is used internally to represent special events, like interrupts.*
- class Interrupt
- interface IParentCondition
- class ResourceEvent

  *Models aspects shared by all resource events.*
- class SimEvent
- class StandaloneEvent
- class Timeout

  *An event that is scheduled with a certain delay after its creation.*
  *This event can be used by processes to wait (or hold their state) for delay time steps. It is immediately scheduled at Env.Now + delay and has thus (in contrast to SimEvent<T>) no Success() or Fail() methods.*

### Functions

- delegate bool **ConditionEval**< **T1** > (Condition< T1 > condition)
- delegate bool **ConditionEval**< **T1, T2** > (Condition< T1, T2 > condition)
- delegate bool **ConditionEval**< **T1, T2, T3** > (Condition< T1, T2, T3 > condition)
- delegate bool **ConditionEval**< **T1, T2, T3, T4** > (Condition< T1, T2, T3, T4 > condition)
- delegate bool **ConditionEval**< **T1, T2, T3, T4, T5** > (Condition< T1, T2, T3, T4, T5 > condition)

## 4.5 DIBRIS.Dessert.Recording Namespace Reference

### Classes

- interface IRecordedResource
- interface IRecorder
- class Monitor

  *An instance of this interface preserves a complete time-series of the observed data values, sample, and their associated times, time. It calculates the data summaries using these series only when they are needed. It is slower and uses more memory than Tally. In long simulations its memory demands may be a disadvantage.*
- struct MonitorSample

*Represents a sample recorded inside a Monitor.*

- class RecorderContract
- class Tally

  *An instance of this interface records enough information (such as sums and sums of squares) while the simulation runs to return simple data summaries. This has the advantage of speed and low memory use. However, they do not preserve a time-series usable in more advanced statistical analysis.*

## 4.6 DIBRIS.Dessert.Resources Namespace Reference

### Classes

- class Container
- class FifoWaitQueue
- class FilterStore
- interface IWaitQueue
- class LifoWaitQueue
- class **OrderedCollection**
- class PreemptiveResource
- class PriorityWaitQueue
- class RandomWaitQueue
- class Resource
- class Store
- class **WaitQueue**
- class WaitQueueBase

### Enumerations

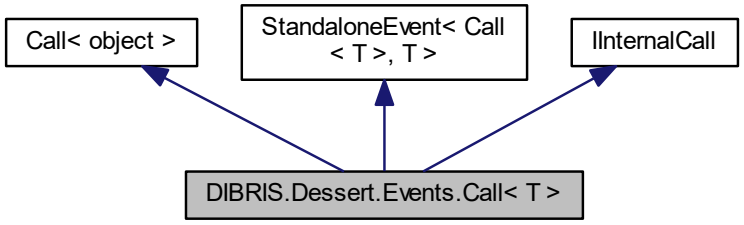
- enum **WaitPolicy** : byte { **FIFO**, **LIFO**, **Priority**, **Random** }

# Chapter 5

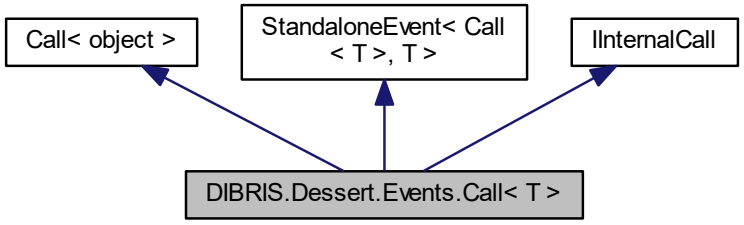
# Class Documentation

## 5.1 DIBRIS.Dessert.Events.Call< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Events.Call< T >:




Collaboration diagram for DIBRIS.Dessert.Events.Call< T >:




### Protected Member Functions

- override void **OnEnd** ()
- override State **ValidStatesMask** ()

### Properties

- override T **Value** `[get]`

### Additional Inherited Members

#### 5.1.1 Detailed Description

Definition at line 31 of file Call.cs.

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

- Events/Call.cs

## 5.2 DIBRIS.Dessert.Events.Condition< T1 > Class Template Reference

#### 5.2.1 Detailed Description

**Type Constraints**

***T1*** **:** ***SimEvent***

Definition at line 77 of file Condition.cs.

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

- Events/Condition.cs

## 5.3 DIBRIS.Dessert.Events.Condition< T1 > Class Template Reference

#### 5.3.1 Detailed Description

**Type Constraints**

***T1*** **:** ***SimEvent***

Definition at line 77 of file Condition.cs.

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

- Events/Condition.cs

## 5.4 DIBRIS.Dessert.Events.Condition< T1 > Class Template Reference

#### 5.4.1 Detailed Description

**Type Constraints**

***T1*** **:** ***SimEvent***

Definition at line 77 of file Condition.cs.

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

- Events/Condition.cs

## 5.5 DIBRIS.Dessert.Events.Condition< T1 > Class Template Reference

### 5.5.1 Detailed Description

**Type Constraints**

***T1*** **:** ***SimEvent***

Definition at line 77 of file Condition.cs.

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

- Events/Condition.cs

## 5.6 DIBRIS.Dessert.Events.Condition< T1 > Class Template Reference

### 5.6.1 Detailed Description

**Type Constraints**

***T1*** **:** ***SimEvent***

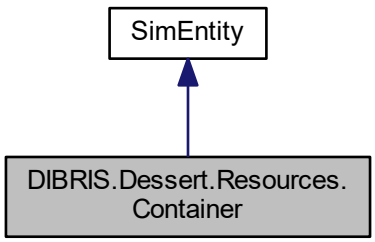
Definition at line 77 of file Condition.cs.

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

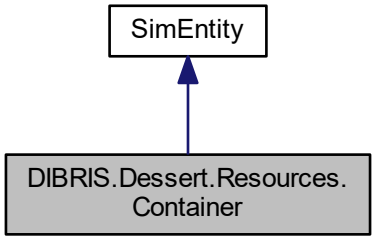
- Events/Condition.cs

## 5.7 DIBRIS.Dessert.Resources.Container Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Container:



Collaboration diagram for DIBRIS.Dessert.Resources.Container:




### Classes

- class GetEvent
- class PutEvent

### Public Member Functions

- GetEvent **Get** (double quantity)
- GetEvent **Get** (double quantity, double priority)
- PutEvent **Put** (double quantity)
- PutEvent **Put** (double quantity, double priority)

### Properties

- double **Capacity** `[get]`
- WaitPolicy **GetPolicy** `[get]`
- IEnumerable< GetEvent > **GetQueue** `[get]`
- double **Level** `[get]`
- WaitPolicy **PutPolicy** `[get]`
- IEnumerable< PutEvent > **PutQueue** `[get]`

### 5.7.1 Detailed Description

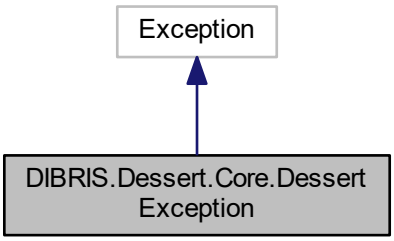
Definition at line 35 of file Container.cs.

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

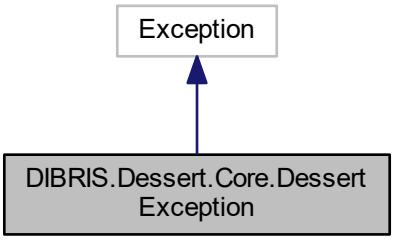
- Resources/Container.cs

## 5.8 DIBRIS.Dessert.Core.DessertException Class Reference

Inheritance diagram for DIBRIS.Dessert.Core.DessertException:




Collaboration diagram for DIBRIS.Dessert.Core.DessertException:




### Public Member Functions

- **DessertException** (string message)

### 5.8.1 Detailed Description

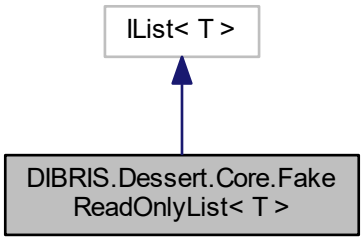
Definition at line 31 of file DessertException.cs.

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

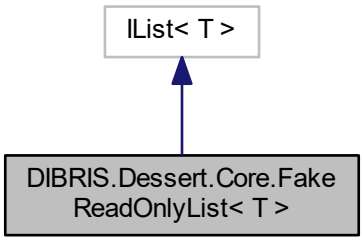
- Core/DessertException.cs

## 5.9 DIBRIS.Dessert.Core.FakeReadOnlyList< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Core.FakeReadOnlyList< T >:




Collaboration diagram for DIBRIS.Dessert.Core.FakeReadOnlyList< T >:




### Public Member Functions

- void **ForceAdd** (T item)
- void **Add** (T item)
- void **Clear** ()
- bool **Contains** (T item)
- void CopyTo (T[ ] array, int arrayIndex)

  *Copies the elements of the T:System.Collections.Generic.ICollection‘1 to an T:System.Array, starting at a particular T:System.Array index.*
- IEnumerator< T > **GetEnumerator** ()
- int IndexOf (T item)

  *Determines the index of a specific item in the FakeReadOnlyList<T>.*
- void **Insert** (int index, T item)
- bool **Remove** (T item)
- void **RemoveAt** (int index)

## Properties

- int **Count** [get]
- bool **IsReadOnly** [get]
- T **this[int index]** [get, set]

### 5.9.1 Detailed Description

Definition at line 33 of file FakeReadOnlyList.cs.

### 5.9.2 Member Function Documentation

#### 5.9.2.1 void DIBRIS.Dessert.Core.FakeReadOnlyList< T >.CopyTo ( T[ ] *array,* int *arrayIndex* )

Copies the elements of the T:System.Collections.Generic.ICollection‘1 to an T:System.Array, starting at a particular T:System.Array index.

**Parameters**

| | |
|---|---|
| *array* | The one-dimensional T:System.Array that is the destination of the elements copied from T:↩ System.Collections.Generic.ICollection‘1. The T:System.Array must have zero-based indexing. |
| *arrayIndex* | The zero-based index in *array* at which copying begins. |

**Exceptions**

| | |
|---|---|
| *T:System.ArgumentNull↩ Exception* | *array* is null. |
| *T:System.ArgumentOutOf↩ RangeException* | *arrayIndex* is less than 0. |
| *T:System.Argument↩ Exception* | The number of elements in the source T:System.Collections.Generic.ICollection‘1 is greater than the available space from *arrayIndex* to the end of the destination *array* . |

Definition at line 103 of file FakeReadOnlyList.cs.

#### 5.9.2.2 int DIBRIS.Dessert.Core.FakeReadOnlyList< T >.IndexOf ( T *item* )

Determines the index of a specific item in the FakeReadOnlyList<T>.

**Parameters**

| | |
|---|---|
| *item* | The object to locate in the FakeReadOnlyList<T>. |

**Returns**

The index of *item* if found in the list; otherwise, -1.

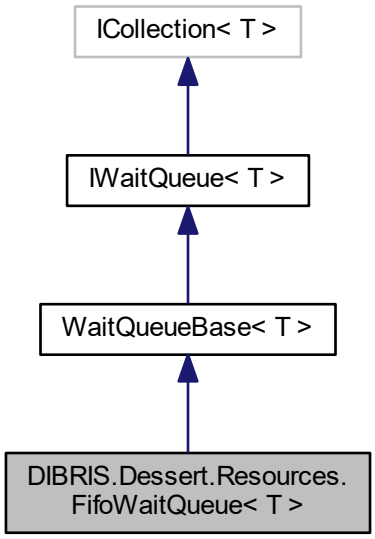
Definition at line 133 of file FakeReadOnlyList.cs.

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

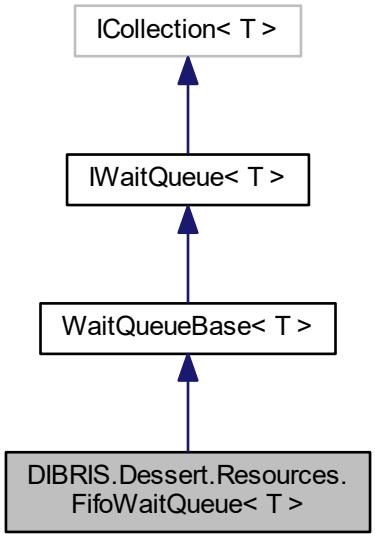
- Core/FakeReadOnlyList.cs

## 5.10 DIBRIS.Dessert.Resources.FifoWaitQueue< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.FifoWaitQueue< T >:




Collaboration diagram for DIBRIS.Dessert.Resources.FifoWaitQueue< T >:



### Public Member Functions

- override void **Add** (T item, double priority)
- override bool **Contains** (T item)
- override IEnumerator< T > **GetEnumerator** ()
- override bool **Remove** (T item)
- override T **RemoveFirst** ()

### Properties

- override int **Count** `[get]`
- override T **First** `[get]`
- override WaitPolicy **Policy** `[get]`

#### 5.10.1 Detailed Description

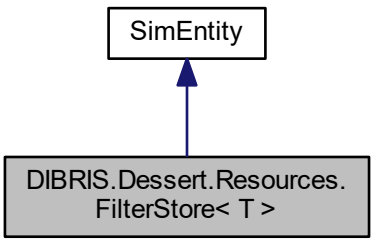
Definition at line 157 of file WaitQueues.cs.

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

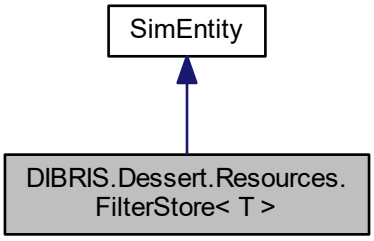
- Resources/WaitQueues.cs

## 5.11 DIBRIS.Dessert.Resources.FilterStore< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.FilterStore< T >:



Collaboration diagram for DIBRIS.Dessert.Resources.FilterStore< T >:




### Classes

- class GetEvent
- class PutEvent

### Public Member Functions

- GetEvent **Get** ()
- GetEvent **Get** (double priority)
- GetEvent **Get** (Predicate< T > filter)
- GetEvent **Get** (Predicate< T > filter, double priority)
- PutEvent **Put** (T item)
- PutEvent **Put** (T item, double putPriority)
- PutEvent **Put** (T item, double putPriority, double itemPriority)

### Properties

- int **Capacity** `[get]`
- int **Count** `[get]`
- WaitPolicy **GetPolicy** `[get]`
- IEnumerable< GetEvent > **GetQueue** `[get]`
- WaitPolicy **ItemPolicy** `[get]`
- IEnumerable< T > **ItemQueue** `[get]`
- WaitPolicy **PutPolicy** `[get]`
- IEnumerable< PutEvent > **PutQueue** `[get]`

#### 5.11.1 Detailed Description

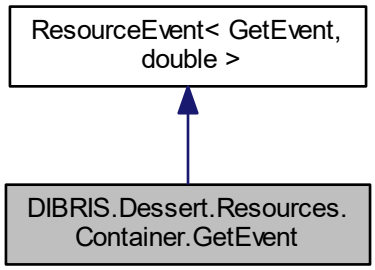
Definition at line 34 of file FilterStore.cs.

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

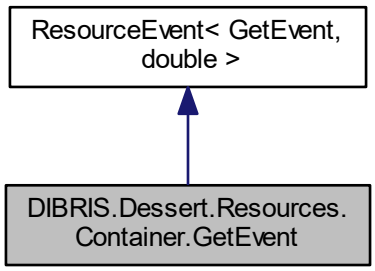
- Resources/FilterStore.cs

## 5.12 DIBRIS.Dessert.Resources.Container.GetEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Container.GetEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.Container.GetEvent:




### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- override double Value `[get]`

  *QUANTITY*

### 5.12.1 Detailed Description

Definition at line 121 of file Container.cs.

### 5.12.2 Property Documentation

#### 5.12.2.1 override double DIBRIS.Dessert.Resources.Container.GetEvent.Value [get]

QUANTITY

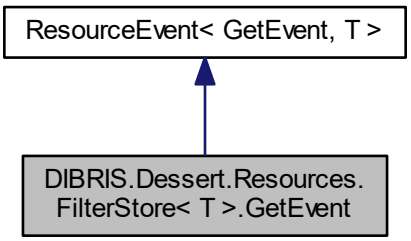
Definition at line 172 of file Container.cs.

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

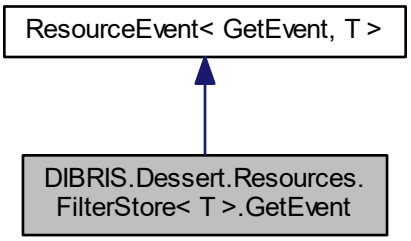
- Resources/Container.cs

## 5.13 DIBRIS.Dessert.Resources.FilterStore< T >.GetEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.FilterStore< T >.GetEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.FilterStore< T >.GetEvent:



### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- Predicate< T > **Filter** `[get]`
- override T **Value** `[get]`

#### 5.13.1 Detailed Description

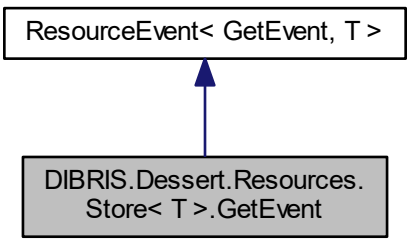
Definition at line 139 of file FilterStore.cs.

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

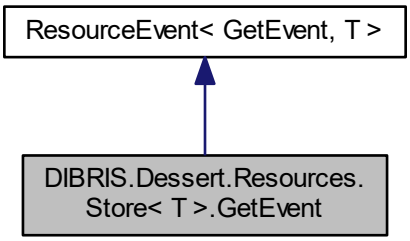
- Resources/FilterStore.cs

## 5.14 DIBRIS.Dessert.Resources.Store< T >.GetEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Store< T >.GetEvent:



Collaboration diagram for DIBRIS.Dessert.Resources.Store< T >.GetEvent:




### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- override T **Value** `[get]`

### 5.14.1 Detailed Description

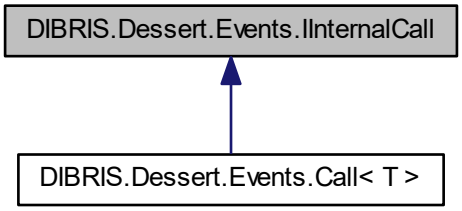
Definition at line 128 of file Store.cs.

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

- Resources/Store.cs

## 5.15 DIBRIS.Dessert.Events.IInternalCall Interface Reference

Inheritance diagram for DIBRIS.Dessert.Events.IInternalCall:




### Public Member Functions

- void **SetValue** (object value)

### Properties

- IInternalCall **PreviousCall** `[get, set]`
- IEnumerator< SimEvent > **Steps** `[get, set]`

### 5.15.1 Detailed Description

Definition at line 95 of file Call.cs.

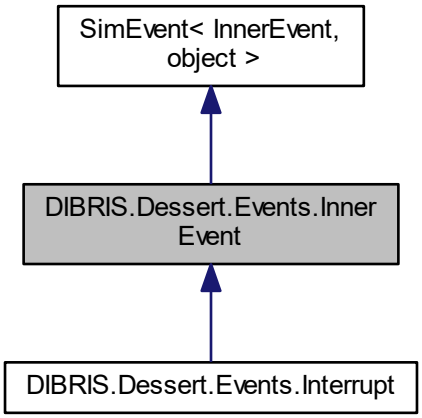
The documentation for this interface was generated from the following file:

- Events/Call.cs

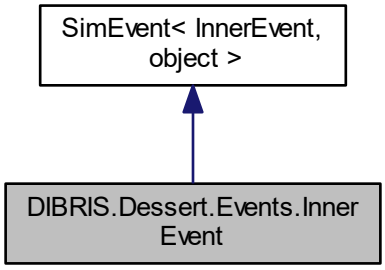
## 5.16 DIBRIS.Dessert.Events.InnerEvent Class Reference

Represents an event which cannot be "yielded" by any user process. It is used internally to represent special events, like interrupts.

Inheritance diagram for DIBRIS.Dessert.Events.InnerEvent:




Collaboration diagram for DIBRIS.Dessert.Events.InnerEvent:




### Protected Member Functions

- override State **ValidStatesMask** ()

### Properties

- override sealed bool **CanHaveParents** `[get]`
- override sealed bool **CanHaveSubscribers** `[get]`

### 5.16.1 Detailed Description

Represents an event which cannot be "yielded" by any user process. It is used internally to represent special events, like interrupts.

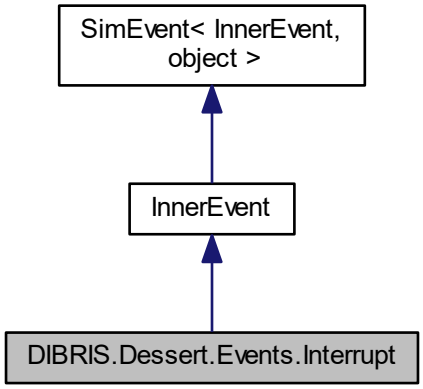
Definition at line 110 of file Templates.cs.

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

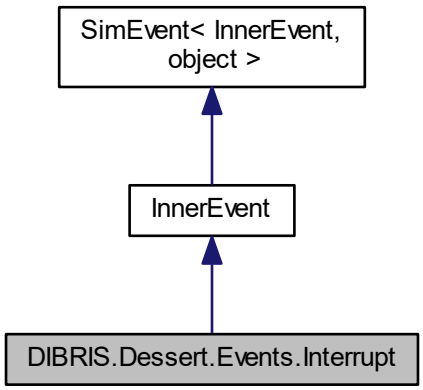
- Events/Templates.cs

## 5.17 DIBRIS.Dessert.Events.Interrupt Class Reference

Inheritance diagram for DIBRIS.Dessert.Events.Interrupt:




Collaboration diagram for DIBRIS.Dessert.Events.Interrupt:




### Protected Member Functions

- override void **OnEnd** ()

- override State **ValidStatesMask** ()

### Properties

- override object **Value** `[get]`

#### 5.17.1 Detailed Description

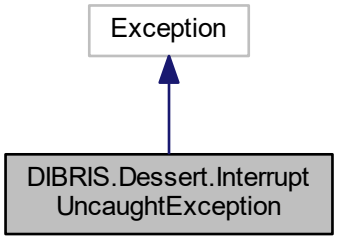
Definition at line 29 of file Interrupt.cs.

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

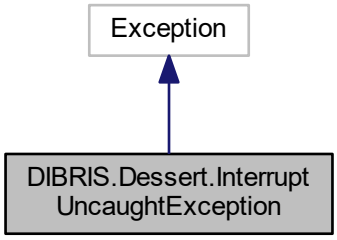
- Events/Interrupt.cs

## 5.18 DIBRIS.Dessert.InterruptUncaughtException Class Reference

Inheritance diagram for DIBRIS.Dessert.InterruptUncaughtException:




Collaboration diagram for DIBRIS.Dessert.InterruptUncaughtException:




#### 5.18.1 Detailed Description

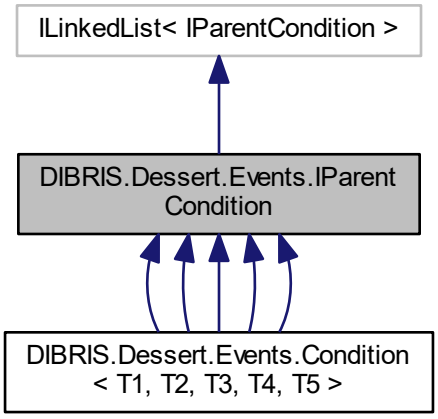
Definition at line 647 of file Sim.cs.

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

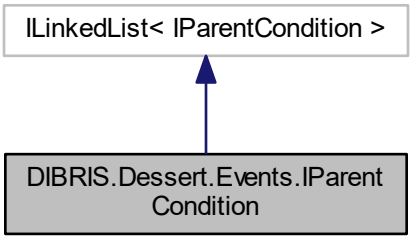
- Sim.cs

## 5.19 DIBRIS.Dessert.Events.IParentCondition Interface Reference

Inheritance diagram for DIBRIS.Dessert.Events.IParentCondition:




Collaboration diagram for DIBRIS.Dessert.Events.IParentCondition:




### Public Member Functions

- void **Trigger** (SimEvent child)

### Properties

- bool **Succeeded** `[get]`

### 5.19.1 Detailed Description

Definition at line 39 of file Condition.cs.

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

- Events/Condition.cs

## 5.20 DIBRIS.Dessert.Recording.IRecordedResource Interface Reference

### Properties

- Tally FulfilledRequestsTally `[get]`

  *An instance of Tally that periodically records the number of requests fulfilled by this resource. The frequency of recordings is given by RecordingFrequency.*
- Double RecordingFrequency `[get]`

  *The frequency at which some tallies of this interface update themselves.*
- Tally UndoneRequestsTally `[get]`

  *An instance of Tally that periodically records the number of requests undone. The frequency of recordings is given by RecordingFrequency.*
- Tally UsageTally `[get]`

  *An instance of Tally that periodically records the number of the users of this resource (given by Resource.Count). The frequency of recordings is given by RecordingFrequency.*
- Tally WaitingTimeTally `[get]`

  *An instance of Tally that records the time waited by each process.*

### 5.20.1 Detailed Description

Definition at line 308 of file IRecorder.cs.

### 5.20.2 Property Documentation

#### 5.20.2.1 Tally DIBRIS.Dessert.Recording.IRecordedResource.FulfilledRequestsTally `[get]`

An instance of Tally that periodically records the number of requests fulfilled by this resource. The frequency of recordings is given by RecordingFrequency.

Definition at line 314 of file IRecorder.cs.

#### 5.20.2.2 Double DIBRIS.Dessert.Recording.IRecordedResource.RecordingFrequency `[get]`

The frequency at which some tallies of this interface update themselves.

Definition at line 319 of file IRecorder.cs.

#### 5.20.2.3 Tally DIBRIS.Dessert.Recording.IRecordedResource.UndoneRequestsTally `[get]`

An instance of Tally that periodically records the number of requests undone. The frequency of recordings is given by RecordingFrequency.

Definition at line 325 of file IRecorder.cs.

#### 5.20.2.4 Tally DIBRIS.Dessert.Recording.IRecordedResource.UsageTally [get]

An instance of Tally that periodically records the number of the users of this resource (given by Resource.Count). The frequency of recordings is given by RecordingFrequency.

Definition at line 332 of file IRecorder.cs.

#### 5.20.2.5 Tally DIBRIS.Dessert.Recording.IRecordedResource.WaitingTimeTally [get]

An instance of Tally that records the time waited by each process.

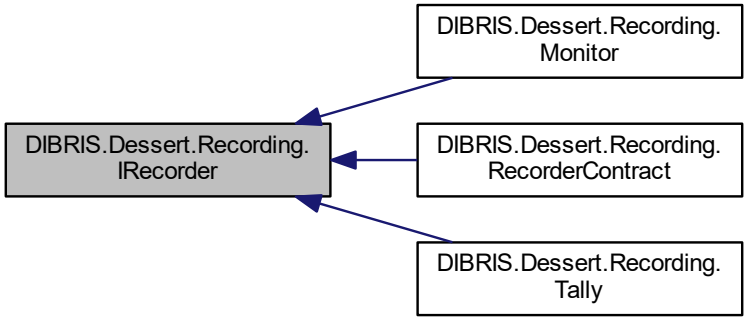
Definition at line 337 of file IRecorder.cs.

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

- Recording/IRecorder.cs

## 5.21 DIBRIS.Dessert.Recording.IRecorder Interface Reference

Inheritance diagram for DIBRIS.Dessert.Recording.IRecorder:




### Public Member Functions

- double Mean ()

  *Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to*
- void Observe (double sample)

  *Records the current value of the variable sample . Since time has not been specified, it is set to SimEnvironment.Now.*
- void Observe (double sample, double time)

  *Records the current value of the variable sample at given time .*
- void Reset ()

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to SimEnvironment.Now.*
- void Reset (double time)

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to time .*
- double StdDev ()

  *Returns the standard deviation of the observations, computed as the square root of Variance.*
- double TimeMean ()

*Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.*

- double TimeMean (double time)

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeStdDev ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeStdDev (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeVariance ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeVariance (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double Total ()

  *Returns the sum of the observed values.*
- double Variance ()

  *Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by*

### Properties

- int Count `[get]`

  *The current number of observations.*
- SimEnvironment Env `[get]`

  *Returns the environment in which this entity was created.*
- double LastTime `[get]`

  *The time of last recording.*
- double StartTime `[get]`

  *The time at which recording has started.*

## 5.21.1 Detailed Description

Definition at line 35 of file IRecorder.cs.

## 5.21.2 Member Function Documentation

### 5.21.2.1 double DIBRIS.Dessert.Recording.IRecorder.Mean ( )

Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to `Total/Count`.

**Returns**

The simple average of the observed values, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

### 5.21.2.2 void DIBRIS.Dessert.Recording.IRecorder.Observe ( double *sample* )

Records the current value of the variable *sample* . Since time has not been specified, it is set to SimEnvironment.↩ Now.

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | Implicitly assigned time is less than the last observation time. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.3 void DIBRIS.Dessert.Recording.IRecorder.Observe ( double *sample,* double *time* )

Records the current value of the variable *sample* at given *time* .

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |
| *time* | The time that will be associated with given value. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than the last observation time. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.4 void DIBRIS.Dessert.Recording.IRecorder.Reset ( )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to Sim↩ Environment.Now.

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.5 void DIBRIS.Dessert.Recording.IRecorder.Reset ( double *time* )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to *time* .

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.6 double DIBRIS.Dessert.Recording.IRecorder.StdDev ( )

Returns the standard deviation of the observations, computed as the square root of Variance.

**Returns**

The standard deviation of the observations, computed as the square root of Variance.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.7 double DIBRIS.Dessert.Recording.IRecorder.TimeMean ( )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.8 double DIBRIS.Dessert.Recording.IRecorder.TimeMean ( double *time* )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange*↩ *Exception* | *time* is less than StartTime. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.9 double DIBRIS.Dessert.Recording.IRecorder.TimeStdDev ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.10 double DIBRIS.Dessert.Recording.IRecorder.TimeStdDev ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.11 double DIBRIS.Dessert.Recording.IRecorder.TimeVariance ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.12 double DIBRIS.Dessert.Recording.IRecorder.TimeVariance ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.13 double DIBRIS.Dessert.Recording.IRecorder.Total ( )

Returns the sum of the observed values.

**Returns**

The sum of the observed values.

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

#### 5.21.2.14 double DIBRIS.Dessert.Recording.IRecorder.Variance ( )

Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by

`Count/(Count - 1).`

**Returns**

The sample variance of the observations, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implemented in DIBRIS.Dessert.Recording.RecorderContract, DIBRIS.Dessert.Recording.Monitor, and DIBRIS.↩ Dessert.Recording.Tally.

### 5.21.3 Property Documentation

#### 5.21.3.1 int DIBRIS.Dessert.Recording.IRecorder.Count [get]

The current number of observations.

Definition at line 41 of file IRecorder.cs.

#### 5.21.3.2 SimEnvironment DIBRIS.Dessert.Recording.IRecorder.Env [get]

Returns the environment in which this entity was created.

Definition at line 47 of file IRecorder.cs.

#### 5.21.3.3 double DIBRIS.Dessert.Recording.IRecorder.LastTime [get]

The time of last recording.

Definition at line 53 of file IRecorder.cs.

#### 5.21.3.4 double DIBRIS.Dessert.Recording.IRecorder.StartTime [get]

The time at which recording has started.

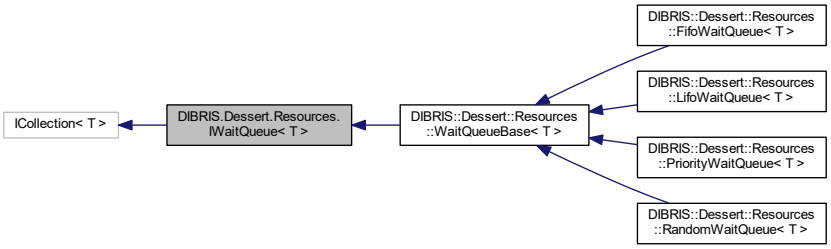
Definition at line 59 of file IRecorder.cs.

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

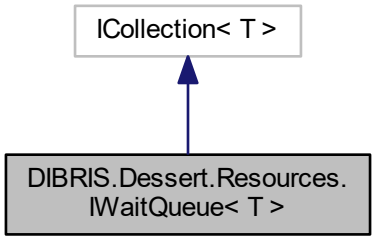
- Recording/IRecorder.cs

## 5.22 DIBRIS.Dessert.Resources.IWaitQueue< T > Interface Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.IWaitQueue< T >:



Collaboration diagram for DIBRIS.Dessert.Resources.IWaitQueue< T >:




### Public Member Functions

- void **Add** (T item, double priority)
- T **RemoveFirst** ()

### Properties

- T **First** `[get]`
- WaitPolicy **Policy** `[get]`

### 5.22.1 Detailed Description

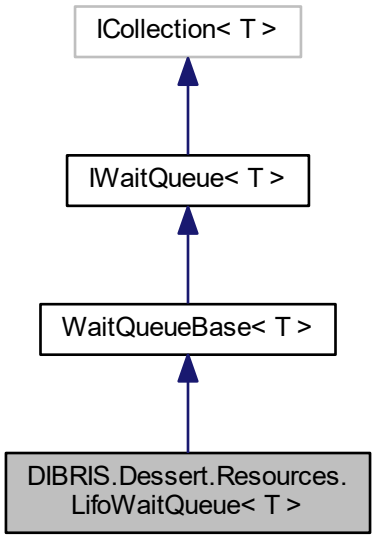
Definition at line 37 of file WaitQueues.cs.

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

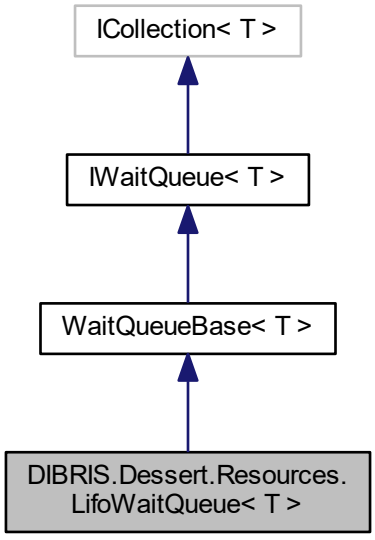
- Resources/WaitQueues.cs

## 5.23 DIBRIS.Dessert.Resources.LifoWaitQueue< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.LifoWaitQueue< T >:




Collaboration diagram for DIBRIS.Dessert.Resources.LifoWaitQueue< T >:



### Public Member Functions

- override void **Add** (T item, double priority)
- override bool **Contains** (T item)
- override IEnumerator< T > **GetEnumerator** ()
- override bool **Remove** (T item)
- override T **RemoveFirst** ()

### Properties

- override int **Count** `[get]`
- override T **First** `[get]`
- override WaitPolicy **Policy** `[get]`

#### 5.23.1 Detailed Description

Definition at line 205 of file WaitQueues.cs.

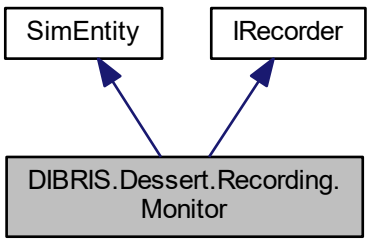
The documentation for this class was generated from the following file:

- Resources/WaitQueues.cs

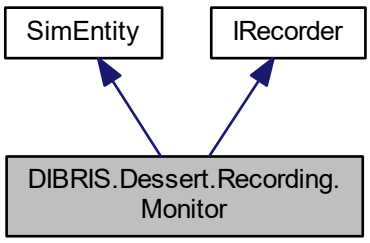
## 5.24 DIBRIS.Dessert.Recording.Monitor Class Reference

An instance of this interface preserves a complete time-series of the observed data values, sample, and their associated times, time. It calculates the data summaries using these series only when they are needed. It is slower and uses more memory than Tally. In long simulations its memory demands may be a disadvantage.

Inheritance diagram for DIBRIS.Dessert.Recording.Monitor:



Collaboration diagram for DIBRIS.Dessert.Recording.Monitor:




## Public Member Functions

- double Mean ()

  *Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to*
- void Observe (double sample)

  *Records the current value of the variable sample . Since time has not been specified, it is set to SimEnvironment.Now.*
- void Observe (double sample, double time)

  *Records the current value of the variable sample at given time .*
- void Reset ()

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to SimEnvironment.Now.*
- void Reset (double time)

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to time .*
- double StdDev ()

  *Returns the standard deviation of the observations, computed as the square root of Variance.*
- double TimeMean ()

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeMean (double time)

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeStdDev ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeStdDev (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeVariance ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeVariance (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double Total ()

  *Returns the sum of the observed values.*
- double Variance ()

  *Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by*

### Properties

- IEnumerable< MonitorSample > Samples `[get]`
- MonitorSample this[int i] `[get]`

  *Returns the i-th sample recorded inside the monitor.*
- int **Count** `[get]`
- double **LastTime** `[get]`
- double **StartTime** `[get]`

## 5.24.1 Detailed Description

An instance of this interface preserves a complete time-series of the observed data values, sample, and their associated times, time. It calculates the data summaries using these series only when they are needed. It is slower and uses more memory than Tally. In long simulations its memory demands may be a disadvantage.

Monitors and tallies may not be bound to a specific SimEnvironment, in order to ease their usage in inter environment recordings; when they are unbounded their SimEntity.Env property points to a dummy environment.
However, please pay attention to the fact that both monitors and tallies are not thread safe: therefore, recall this fact when you use them in a multi threaded simulation scenario.

Definition at line 46 of file Monitor.cs.

## 5.24.2 Member Function Documentation

### 5.24.2.1 double DIBRIS.Dessert.Recording.Monitor.Mean ( )

Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to `Total/Count`.

**Returns**

The simple average of the observed values, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 98 of file Monitor.cs.

### 5.24.2.2 void DIBRIS.Dessert.Recording.Monitor.Observe ( double *sample* )

Records the current value of the variable *sample* . Since time has not been specified, it is set to SimEnvironment.Now.

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | Implicitly assigned time is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 103 of file Monitor.cs.

#### 5.24.2.3 void DIBRIS.Dessert.Recording.Monitor.Observe ( double *sample,* double *time* )

Records the current value of the variable *sample* at given *time* .

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |
| *time* | The time that will be associated with given value. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 108 of file Monitor.cs.

#### 5.24.2.4 void DIBRIS.Dessert.Recording.Monitor.Reset ( )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to Sim↩ Environment.Now.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 113 of file Monitor.cs.

#### 5.24.2.5 void DIBRIS.Dessert.Recording.Monitor.Reset ( double *time* )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to *time* .

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 118 of file Monitor.cs.

#### 5.24.2.6 double DIBRIS.Dessert.Recording.Monitor.StdDev ( )

Returns the standard deviation of the observations, computed as the square root of Variance.

**Returns**

The standard deviation of the observations, computed as the square root of Variance.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 123 of file Monitor.cs.

#### 5.24.2.7 double DIBRIS.Dessert.Recording.Monitor.TimeMean ( )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 128 of file Monitor.cs.

#### 5.24.2.8 double DIBRIS.Dessert.Recording.Monitor.TimeMean ( double *time* )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 133 of file Monitor.cs.

#### 5.24.2.9 double DIBRIS.Dessert.Recording.Monitor.TimeStdDev ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 145 of file Monitor.cs.

#### 5.24.2.10 double DIBRIS.Dessert.Recording.Monitor.TimeStdDev ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 150 of file Monitor.cs.

#### 5.24.2.11 double DIBRIS.Dessert.Recording.Monitor.TimeVariance ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 155 of file Monitor.cs.

#### 5.24.2.12 double DIBRIS.Dessert.Recording.Monitor.TimeVariance ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 160 of file Monitor.cs.

#### 5.24.2.13 double DIBRIS.Dessert.Recording.Monitor.Total ( )

Returns the sum of the observed values.

**Returns**

The sum of the observed values.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 177 of file Monitor.cs.

#### 5.24.2.14 double DIBRIS.Dessert.Recording.Monitor.Variance ( )

Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by

`Count/(Count - 1).`

**Returns**

The sample variance of the observations, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 182 of file Monitor.cs.

### 5.24.3 Property Documentation

#### 5.24.3.1 IEnumerable<MonitorSample> DIBRIS.Dessert.Recording.Monitor.Samples [get]

Definition at line 72 of file Monitor.cs.

#### 5.24.3.2 MonitorSample DIBRIS.Dessert.Recording.Monitor.this[int i] [get]

Returns the i-th sample recorded inside the monitor.

**Parameters**

| | |
|---|---|
| *i* | The index of the sample that has to be retrieved. |

**Returns**

The i-th sample recorded inside the monitor.

Definition at line 82 of file Monitor.cs.

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

- Recording/Monitor.cs

## 5.25 DIBRIS.Dessert.Recording.MonitorSample Struct Reference

Represents a sample recorded inside a Monitor.

### Public Attributes

- readonly double **_sample**
- readonly double **_time**

### Properties

- double Sample [get]

  *The sample recorded in the monitor.*
- double Time [get]

  *The time at which Sample was recorded.*

### 5.25.1 Detailed Description

Represents a sample recorded inside a Monitor.

Definition at line 199 of file Monitor.cs.

### 5.25.2 Property Documentation

#### 5.25.2.1 double DIBRIS.Dessert.Recording.MonitorSample.Sample [get]

The sample recorded in the monitor.

Definition at line 214 of file Monitor.cs.

#### 5.25.2.2 double DIBRIS.Dessert.Recording.MonitorSample.Time [get]

The time at which Sample was recorded.

Definition at line 222 of file Monitor.cs.

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

- Recording/Monitor.cs

## 5.26 DIBRIS.Dessert.Core.OptimizedSkewHeap Class Reference

### Public Member Functions

- **OptimizedSkewHeap** (SimEvent initialRoot)
- void **Add** (SimEvent ev)
- void **RemoveMin** ()

### Properties

- int **Count** [get]
- SimEvent **Min** [get]

### 5.26.1 Detailed Description

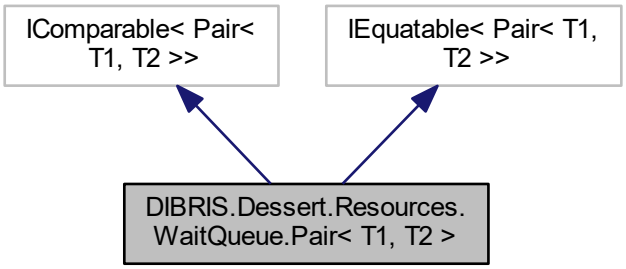
Definition at line 31 of file OptimizedSkewHeap.cs.

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

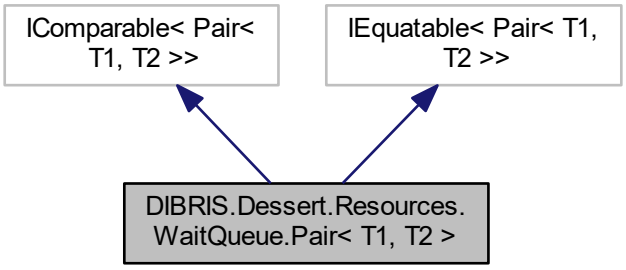
- Core/OptimizedSkewHeap.cs

## 5.27 DIBRIS.Dessert.Resources.WaitQueue.Pair< T1, T2 > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.WaitQueue.Pair< T1, T2 >:



Collaboration diagram for DIBRIS.Dessert.Resources.WaitQueue.Pair< T1, T2 >:




### Public Member Functions

- **Pair** (T1 item1, T2 item2)
- int **CompareTo** (Pair< T1, T2 > other)
- bool **Equals** (Pair< T1, T2 > other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()

### Public Attributes

- readonly T1 **Item1**

#### 5.27.1 Detailed Description

**Type Constraints**

***T2* : *struct***

***T2* : *IComparable*<*T2*>**

Definition at line 120 of file WaitQueues.cs.

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

- Resources/WaitQueues.cs

## 5.28 DIBRIS.Dessert.PreemptionInfo Class Reference

### Properties

- SimProcess By `[get]`
- double UsageSince `[get]`

#### 5.28.1 Detailed Description

Definition at line 379 of file SimProcess.cs.

### 5.28.2 Property Documentation

#### 5.28.2.1 SimProcess DIBRIS.Dessert.PreemptionInfo.By [get]

Definition at line 395 of file SimProcess.cs.

#### 5.28.2.2 double DIBRIS.Dessert.PreemptionInfo.UsageSince [get]

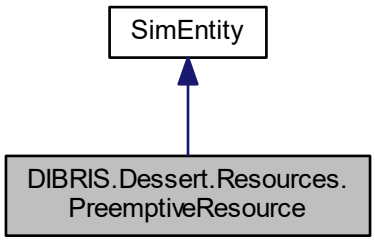
Definition at line 404 of file SimProcess.cs.

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

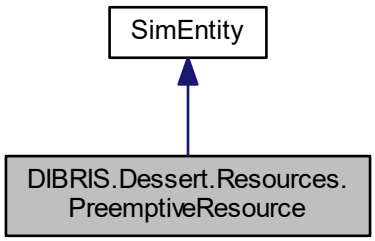
- SimProcess.cs

## 5.29 DIBRIS.Dessert.Resources.PreemptiveResource Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.PreemptiveResource:




Collaboration diagram for DIBRIS.Dessert.Resources.PreemptiveResource:




### Classes

- class ReleaseEvent
- class RequestEvent

### Public Member Functions

- ReleaseEvent **Release** (RequestEvent request)
- RequestEvent **Request** ()
- RequestEvent **Request** (double priority)
- RequestEvent **Request** (double priority, bool preempt)

### Properties

- int **Capacity** `[get]`
- int **Count** `[get]`
- WaitPolicy **RequestPolicy** `[get]`
- IEnumerable< RequestEvent > **RequestQueue** `[get]`
- IEnumerable< RequestEvent > **Users** `[get]`

#### 5.29.1 Detailed Description

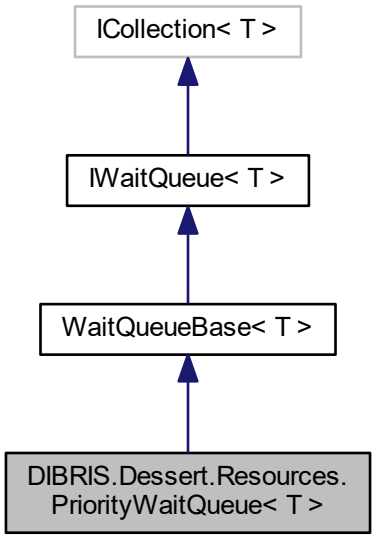
Definition at line 37 of file PreemptiveResource.cs.

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

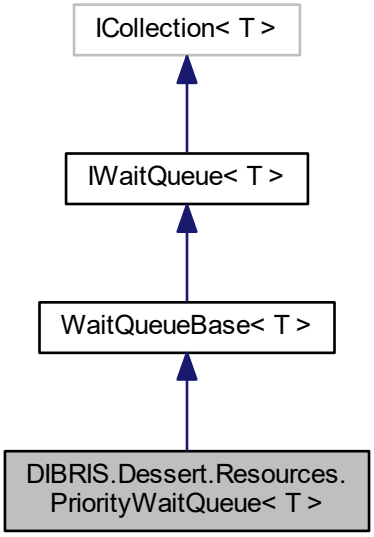
- Resources/PreemptiveResource.cs

## 5.30 DIBRIS.Dessert.Resources.PriorityWaitQueue< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.PriorityWaitQueue< T >:



Collaboration diagram for DIBRIS.Dessert.Resources.PriorityWaitQueue< T >:




### Public Member Functions

- override void **Add** (T item, double priority)
- override bool **Contains** (T item)
- override IEnumerator< T > **GetEnumerator** ()
- override bool **Remove** (T item)
- override T **RemoveFirst** ()

### Properties

- override int **Count** `[get]`
- override T **First** `[get]`
- override WaitPolicy **Policy** `[get]`

#### 5.30.1 Detailed Description

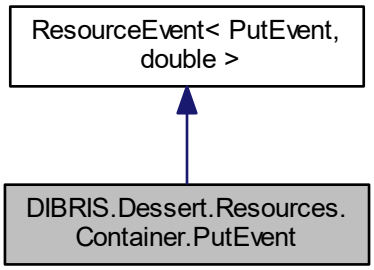
Definition at line 253 of file WaitQueues.cs.

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

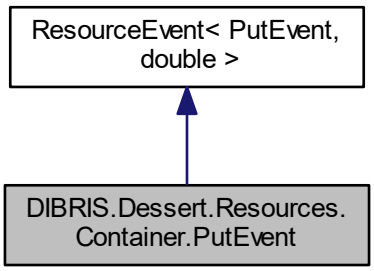
- Resources/WaitQueues.cs

## 5.31 DIBRIS.Dessert.Resources.Container.PutEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Container.PutEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.Container.PutEvent:




### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- override double Value `[get]`

  *QUANTITY*

### 5.31.1 Detailed Description

Definition at line 184 of file Container.cs.

### 5.31.2 Property Documentation

#### 5.31.2.1 override double DIBRIS.Dessert.Resources.Container.PutEvent.Value [get]

QUANTITY

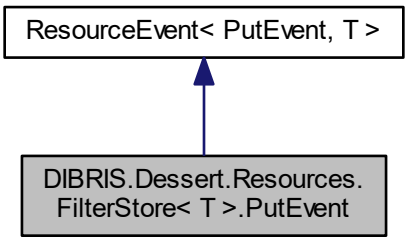
Definition at line 235 of file Container.cs.

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

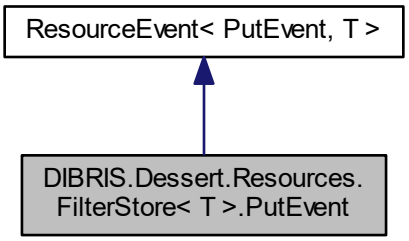
- Resources/Container.cs

## 5.32 DIBRIS.Dessert.Resources.FilterStore< T >.PutEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.FilterStore< T >.PutEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.FilterStore< T >.PutEvent:



### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- T **Item** `[get]`
- double **ItemPriority** `[get]`
- override T **Value** `[get]`

#### 5.32.1 Detailed Description

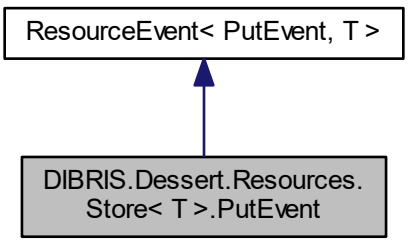
Definition at line 209 of file FilterStore.cs.

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

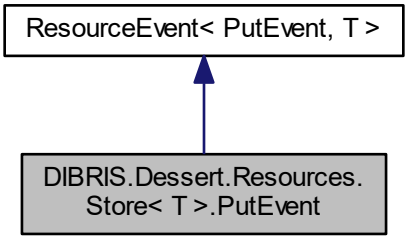
- Resources/FilterStore.cs

## 5.33 DIBRIS.Dessert.Resources.Store< T >.PutEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Store< T >.PutEvent:



Collaboration diagram for DIBRIS.Dessert.Resources.Store< T >.PutEvent:




### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- double **ItemPriority** `[get]`
- override T **Value** `[get]`

### 5.33.1 Detailed Description

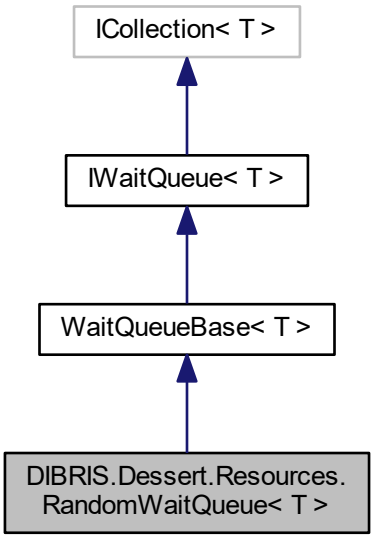
Definition at line 187 of file Store.cs.

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

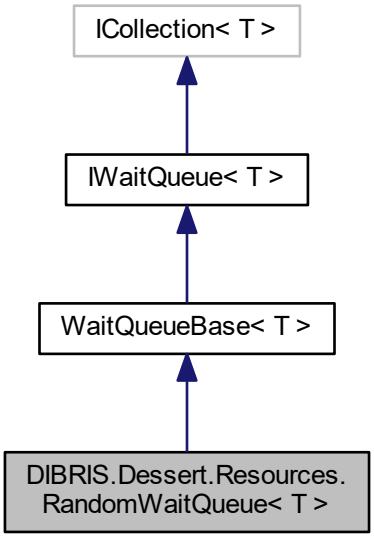
- Resources/Store.cs

## 5.34 DIBRIS.Dessert.Resources.RandomWaitQueue< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.RandomWaitQueue< T >:




Collaboration diagram for DIBRIS.Dessert.Resources.RandomWaitQueue< T >:



### Public Member Functions

- **RandomWaitQueue** (TRandom random)
- override void **Add** (T item, double priority)
- override bool **Contains** (T item)
- override IEnumerator< T > **GetEnumerator** ()
- override bool **Remove** (T item)
- override T **RemoveFirst** ()

### Properties

- override int **Count** `[get]`
- override T **First** `[get]`
- override WaitPolicy **Policy** `[get]`

#### 5.34.1 Detailed Description

Definition at line 316 of file WaitQueues.cs.

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

- Resources/WaitQueues.cs

## 5.35 DIBRIS.Dessert.SimEnvironment.RealTimeOptions Class Reference

Available options for the real-time mode.

### Properties

- bool Enabled `[get, set]`

  *Whether the simulation must be run according to "wall clock" time.*
- double ScalingFactor = false `[get, set]`

  *The real-time scaling factor.*
- IClock WallClock = 1.0 `[get, set]`

  *The "wall clock" used for the real-time simulation.*

#### 5.35.1 Detailed Description

Available options for the real-time mode.

Definition at line 429 of file SimEnvironment.cs.

#### 5.35.2 Property Documentation

##### 5.35.2.1 bool DIBRIS.Dessert.SimEnvironment.RealTimeOptions.Enabled `[get],[set]`

Whether the simulation must be run according to "wall clock" time.

Definition at line 435 of file SimEnvironment.cs.

#### 5.35.2.2 double DIBRIS.Dessert.SimEnvironment.RealTimeOptions.ScalingFactor = false [get], [set]

The real-time scaling factor.

Definition at line 440 of file SimEnvironment.cs.

#### 5.35.2.3 IClock DIBRIS.Dessert.SimEnvironment.RealTimeOptions.WallClock = 1.0 [get], [set]

The "wall clock" used for the real-time simulation.

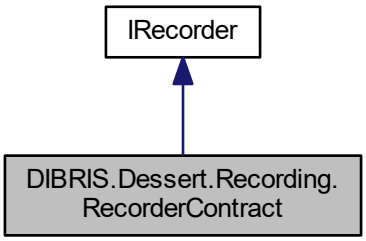
Definition at line 446 of file SimEnvironment.cs.

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

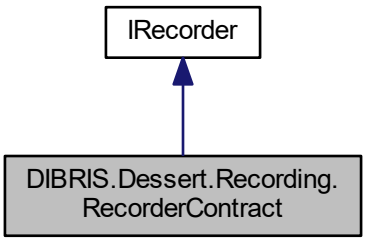
- SimEnvironment.cs

## 5.36 DIBRIS.Dessert.Recording.RecorderContract Class Reference

Inheritance diagram for DIBRIS.Dessert.Recording.RecorderContract:




Collaboration diagram for DIBRIS.Dessert.Recording.RecorderContract:




### Public Member Functions

- abstract double Mean ()

*Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to*

- abstract void Observe (double sample)

  *Records the current value of the variable sample . Since time has not been specified, it is set to SimEnvironment.Now.*
- void Observe (double sample, double time)

  *Records the current value of the variable sample at given time .*
- abstract void Reset ()

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to SimEnvironment.Now.*
- abstract void Reset (double time)

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to time .*
- double StdDev ()

  *Returns the standard deviation of the observations, computed as the square root of Variance.*
- double TimeMean ()

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeMean (double time)

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeStdDev ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeStdDev (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeVariance ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeVariance (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double Total ()

  *Returns the sum of the observed values.*
- double Variance ()

  *Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by*

## Properties

- abstract int **Count** `[get]`
- abstract SimEnvironment **Env** `[get]`
- double **LastTime** `[get]`
- abstract double **StartTime** `[get]`

### 5.36.1 Detailed Description

Definition at line 212 of file IRecorder.cs.

### 5.36.2 Member Function Documentation

#### 5.36.2.1 abstract double DIBRIS.Dessert.Recording.RecorderContract.Mean ( ) `[pure virtual]`

Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to `Total/Count`.

**Returns**

The simple average of the observed values, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

#### 5.36.2.2 abstract void DIBRIS.Dessert.Recording.RecorderContract.Observe ( double *sample* ) [pure virtual]

Records the current value of the variable *sample* . Since time has not been specified, it is set to SimEnvironment.Now.

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange Exception* | Implicitly assigned time is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

#### 5.36.2.3 void DIBRIS.Dessert.Recording.RecorderContract.Observe ( double *sample,* double *time* )

Records the current value of the variable *sample* at given *time* .

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |
| *time* | The time that will be associated with given value. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange Exception* | *time* is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 238 of file IRecorder.cs.

#### 5.36.2.4 abstract void DIBRIS.Dessert.Recording.RecorderContract.Reset ( ) [pure virtual]

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to SimEnvironment.Now.

Implements DIBRIS.Dessert.Recording.IRecorder.

#### 5.36.2.5 abstract void DIBRIS.Dessert.Recording.RecorderContract.Reset ( double *time* ) [pure virtual]

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to *time* .

Implements DIBRIS.Dessert.Recording.IRecorder.

#### 5.36.2.6 double DIBRIS.Dessert.Recording.RecorderContract.StdDev ( )

Returns the standard deviation of the observations, computed as the square root of Variance.

**Returns**

The standard deviation of the observations, computed as the square root of Variance.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 247 of file IRecorder.cs.

#### 5.36.2.7 double DIBRIS.Dessert.Recording.RecorderContract.TimeMean ( )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 253 of file IRecorder.cs.

#### 5.36.2.8 double DIBRIS.Dessert.Recording.RecorderContract.TimeMean ( double *time* )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 259 of file IRecorder.cs.

#### 5.36.2.9 double DIBRIS.Dessert.Recording.RecorderContract.TimeStdDev ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 266 of file IRecorder.cs.

#### 5.36.2.10 double DIBRIS.Dessert.Recording.RecorderContract.TimeStdDev ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 272 of file IRecorder.cs.

#### 5.36.2.11 double DIBRIS.Dessert.Recording.RecorderContract.TimeVariance ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 279 of file IRecorder.cs.

#### 5.36.2.12 double DIBRIS.Dessert.Recording.RecorderContract.TimeVariance ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 285 of file IRecorder.cs.

#### 5.36.2.13 double DIBRIS.Dessert.Recording.RecorderContract.Total ( )

Returns the sum of the observed values.

**Returns**

The sum of the observed values.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 292 of file IRecorder.cs.

#### 5.36.2.14 double DIBRIS.Dessert.Recording.RecorderContract.Variance ( )

Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by

`Count/(Count - 1).`

**Returns**

The sample variance of the observations, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

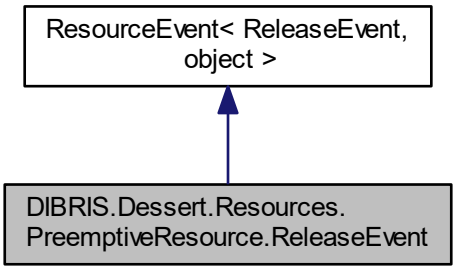
Definition at line 298 of file IRecorder.cs.

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

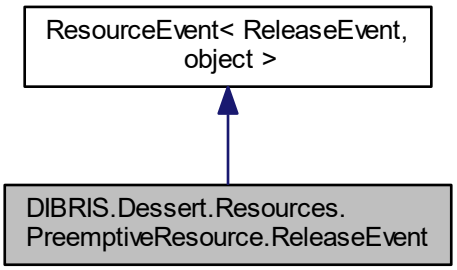
- Recording/IRecorder.cs

## 5.37 DIBRIS.Dessert.Resources.PreemptiveResource.ReleaseEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.PreemptiveResource.ReleaseEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.PreemptiveResource.ReleaseEvent:




### Public Member Functions

- override void **Dispose** ()

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- RequestEvent **Request** `[get]`
- override object **Value** `[get]`

#### 5.37.1 Detailed Description

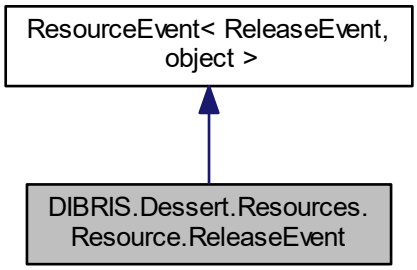
Definition at line 285 of file PreemptiveResource.cs.

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

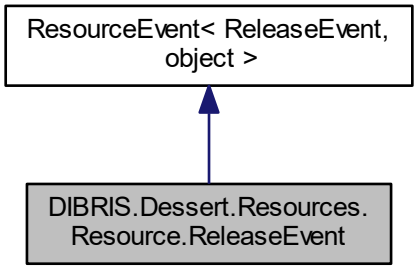
- Resources/PreemptiveResource.cs

## 5.38 DIBRIS.Dessert.Resources.Resource.ReleaseEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Resource.ReleaseEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.Resource.ReleaseEvent:



### Public Member Functions

- override void **Dispose** ()

### Properties

- RequestEvent **Request** `[get]`
- override object **Value** `[get]`

### Additional Inherited Members

#### 5.38.1 Detailed Description

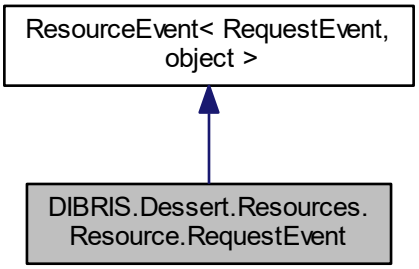
Definition at line 112 of file Resource.cs.

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

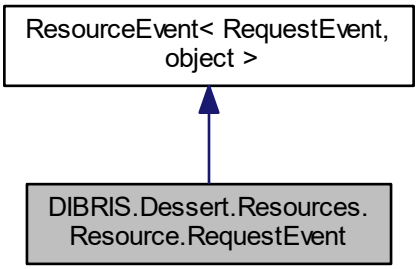
- Resources/Resource.cs

## 5.39 DIBRIS.Dessert.Resources.Resource.RequestEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Resource.RequestEvent:



Collaboration diagram for DIBRIS.Dessert.Resources.Resource.RequestEvent:




### Public Member Functions

- override void **Dispose** ()

### Properties

- Resource **Resource** `[get]`
- override object **Value** `[get]`

### Additional Inherited Members

#### 5.39.1 Detailed Description

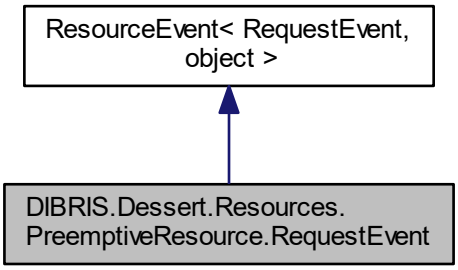
Definition at line 147 of file Resource.cs.

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

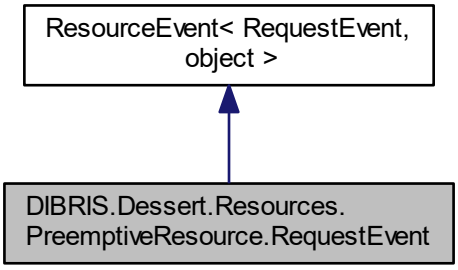
- Resources/Resource.cs

## 5.40 DIBRIS.Dessert.Resources.PreemptiveResource.RequestEvent Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.PreemptiveResource.RequestEvent:




Collaboration diagram for DIBRIS.Dessert.Resources.PreemptiveResource.RequestEvent:




### Public Member Functions

- override void **Dispose** ()

### Properties

- bool **Preempt** `[get]`
- PreemptiveResource **Resource** `[get]`
- double **Time** `[get]`
- override object **Value** `[get]`

### Additional Inherited Members

### 5.40.1 Detailed Description

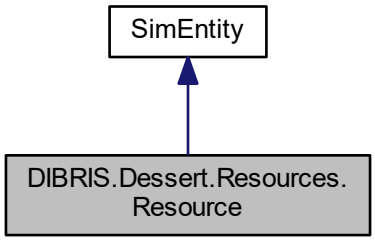
Definition at line 199 of file PreemptiveResource.cs.

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

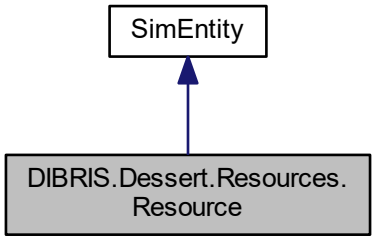
- Resources/PreemptiveResource.cs

## 5.41 DIBRIS.Dessert.Resources.Resource Class Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Resource:




Collaboration diagram for DIBRIS.Dessert.Resources.Resource:




### Classes

- class ReleaseEvent
- class RequestEvent

### Public Member Functions

- ReleaseEvent **Release** (RequestEvent request)
- RequestEvent **Request** ()
- RequestEvent **Request** (double priority)

### Properties

- int **Capacity** [get]
- int **Count** [get]
- WaitPolicy **RequestPolicy** [get]
- IEnumerable< RequestEvent > **RequestQueue** [get]
- IEnumerable< RequestEvent > **Users** [get]

#### 5.41.1 Detailed Description

Definition at line 37 of file Resource.cs.

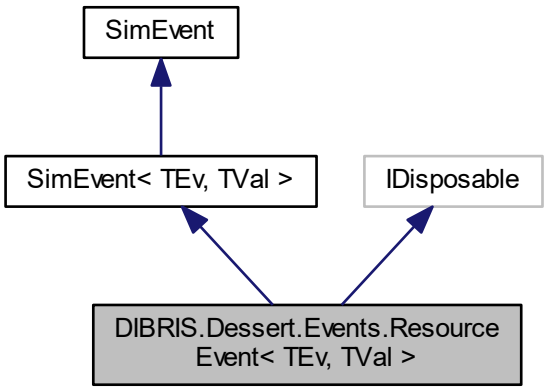
The documentation for this class was generated from the following file:

- Resources/Resource.cs

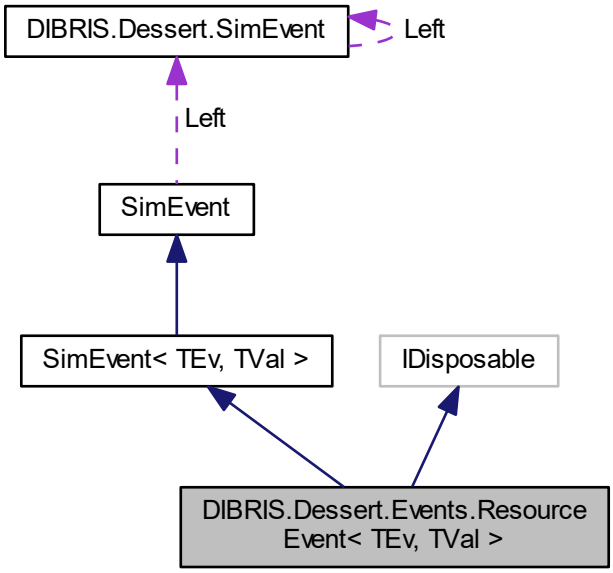
## 5.42 DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal > Class Template Reference

Models aspects shared by all resource events.

Inheritance diagram for DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal >:



Collaboration diagram for DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal >:




### Public Member Functions

- abstract void **Dispose** ()

### Protected Member Functions

- override sealed State ValidStatesMask ()

### Properties

- bool Disposed `[get, protected set]`

  *Returns true if and only if event has been disposed; otherwise, it returns false.*
- double Priority `[get]`

  *The priority assigned to this resource event. Usually, the priority is only considered when the policy is set to Wait↩ Policy.Priority.*

### Additional Inherited Members

### 5.42.1 Detailed Description

Models aspects shared by all resource events.

**Template Parameters**

| | |
|---|---|
| *TEv* | |
| *TVal* | |

**Type Constraints**

***TEv* : *ResourceEvent***

***TEv* : *TEv***

***TEv* : *TVal***

Definition at line 38 of file Templates.cs.

### 5.42.2 Member Function Documentation

#### 5.42.2.1 override sealed State DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal >.ValidStatesMask ( ) [protected], [virtual]

This method is not used in any execution path of Release builds. In fact, this method is just used in Debug.Assert to enforce better integrity.

Reimplemented from DIBRIS.Dessert.SimEvent.

Definition at line 72 of file Templates.cs.

### 5.42.3 Property Documentation

#### 5.42.3.1 bool DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal >.Disposed [get], [protected set]

Returns true if and only if event has been disposed; otherwise, it returns false.

Definition at line 54 of file Templates.cs.

#### 5.42.3.2 double DIBRIS.Dessert.Events.ResourceEvent< TEv, TVal >.Priority [get]

The priority assigned to this resource event. Usually, the priority is only considered when the policy is set to Wait↩ Policy.Priority.

Definition at line 62 of file Templates.cs.

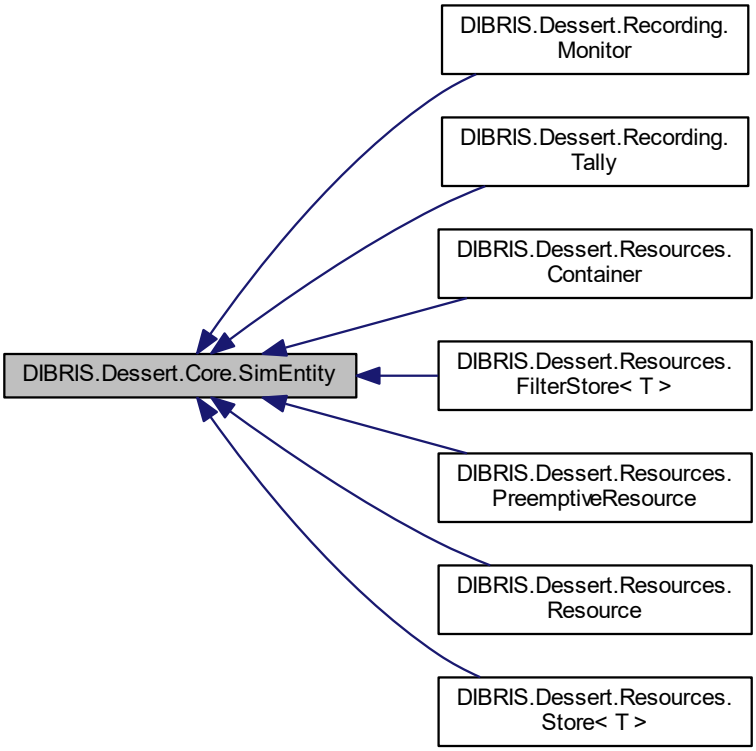
The documentation for this class was generated from the following file:

- Events/Templates.cs

## 5.43 DIBRIS.Dessert.Core.SimEntity Class Reference

Represents an entity that belongs to a specific environment. An entity can only be "used" in the environment it belongs to.

Inheritance diagram for DIBRIS.Dessert.Core.SimEntity:




## Properties

- SimEnvironment Env `[get]`

  *Returns the environment in which this entity was created.*

### 5.43.1 Detailed Description

Represents an entity that belongs to a specific environment. An entity can only be "used" in the environment it belongs to.

Definition at line 35 of file SimEntity.cs.

### 5.43.2 Property Documentation

#### 5.43.2.1 SimEnvironment DIBRIS.Dessert.Core.SimEntity.Env `[get]`

Returns the environment in which this entity was created.

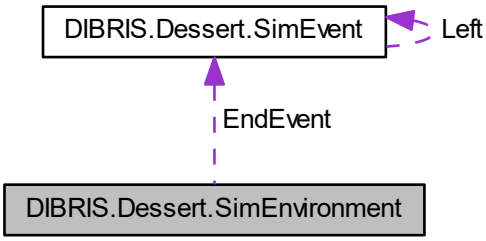
Definition at line 58 of file SimEntity.cs.

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

- Core/SimEntity.cs

## 5.44 DIBRIS.Dessert.SimEnvironment Class Reference

Collaboration diagram for DIBRIS.Dessert.SimEnvironment:




### Classes

- class RealTimeOptions

  *Available options for the real-time mode.*

### Public Member Functions

- bool **IsValidDelay** (double delay)
- SimProcess **Process** (IEnumerable< SimEvent > generator)
- SimProcess **DelayedProcess** (IEnumerable< SimEvent > generator, double delay)
- void **Run** ()
- void **Run** (double until)
- void **Run** (int until)
- void **Run** (SimEvent until)
- override string **ToString** ()
- SimEvent< object > Event ()

  *Returns a new generic event.*
- SimEvent< TVal > Event< TVal > ()

  *Returns a new generic event.*
- SimEvent Exit ()

  *Exits from current process or from current call. If called directly from a process body, then the process is stopped and the optional exit value can be found on SimProcess.Value. Otherwise, if this method is called from a procedure body, then the procedure is stopped.*
- SimEvent Exit (object value)

  *Exits from current process or from current call. If called directly from a process body, then the process is stopped and the optional exit value can be found on SimProcess.Value. Otherwise, if this method is called from a procedure body, then the procedure is stopped and the optional exit value can be found on the event returned by Call.*

### Public Attributes

- double Now => _now

  *Returns current simulation time.*
- TRandom Random => _random

  *A random numbers generator which can be used inside simulations.*

### Properties

- SimProcess ActiveProcess [get]

  *The process that is currently running in the simulation.*
- bool **Ended** [get]
- double Peek [get]

  *Returns the time of the next scheduled event, or double.PositiveInfinity if there is no further event.*
- RealTimeOptions RealTime [get]

  *Options for the real-time mode.*

## 5.44.1 Detailed Description

Definition at line 34 of file SimEnvironment.cs.

## 5.44.2 Member Function Documentation

### 5.44.2.1 SimEvent<object> DIBRIS.Dessert.SimEnvironment.Event ( )

Returns a new generic event.

**Returns**

A new generic event.

Definition at line 362 of file SimEnvironment.cs.

### 5.44.2.2 SimEvent<TVal> DIBRIS.Dessert.SimEnvironment.Event< TVal > ( )

Returns a new generic event.

**Returns**

A new generic event.

Definition at line 374 of file SimEnvironment.cs.

### 5.44.2.3 SimEvent DIBRIS.Dessert.SimEnvironment.Exit ( )

Exits from current process or from current call. If called directly from a process body, then the process is stopped and the optional exit value can be found on SimProcess.Value. Otherwise, if this method is called from a procedure body, then the procedure is stopped.

**Returns**

The exit event that can be yielded to stop a process or a call.

Definition at line 394 of file SimEnvironment.cs.

### 5.44.2.4 SimEvent DIBRIS.Dessert.SimEnvironment.Exit ( object *value* )

Exits from current process or from current call. If called directly from a process body, then the process is stopped and the optional exit value can be found on SimProcess.Value. Otherwise, if this method is called from a procedure body, then the procedure is stopped and the optional exit value can be found on the event returned by Call.

**Parameters**

| | |
|---|---|
| *value* | The optional exit value. |

**Returns**

The exit event that can be yielded to stop a process or a call.

Definition at line 409 of file SimEnvironment.cs.

### 5.44.3 Member Data Documentation

#### 5.44.3.1 double DIBRIS.Dessert.SimEnvironment.Now => _now

Returns current simulation time.

**Returns**

Current simulation time.

Definition at line 300 of file SimEnvironment.cs.

#### 5.44.3.2 TRandom DIBRIS.Dessert.SimEnvironment.Random => _random

A random numbers generator which can be used inside simulations.

Definition at line 354 of file SimEnvironment.cs.

### 5.44.4 Property Documentation

#### 5.44.4.1 SimProcess DIBRIS.Dessert.SimEnvironment.ActiveProcess [get]

The process that is currently running in the simulation.

Definition at line 284 of file SimEnvironment.cs.

#### 5.44.4.2 double DIBRIS.Dessert.SimEnvironment.Peek [get]

Returns the time of the next scheduled event, or double.PositiveInfinity if there is no further event.

**Returns**

The time of the next scheduled event, or double.PositiveInfinity if there is no further event.

Definition at line 341 of file SimEnvironment.cs.

#### 5.44.4.3 RealTimeOptions DIBRIS.Dessert.SimEnvironment.RealTime [get]

Options for the real-time mode.

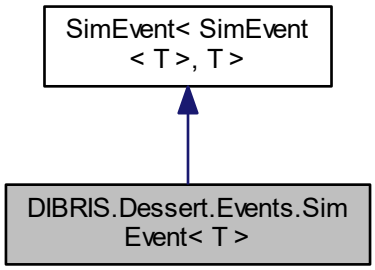
Definition at line 424 of file SimEnvironment.cs.

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

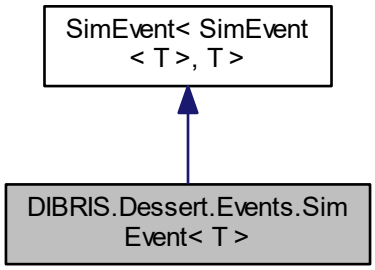
- SimEnvironment.cs

## 5.45 DIBRIS.Dessert.Events.SimEvent< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Events.SimEvent< T >:




Collaboration diagram for DIBRIS.Dessert.Events.SimEvent< T >:




### Public Member Functions

- void Fail ()
- void Fail (T val)
- bool **TryFail** ()
- bool **TryFail** (T val)
- void Succeed ()
- void Succeed (T val)
- bool **TrySucceed** ()
- bool **TrySucceed** (T val)

### Protected Member Functions

- override State **ValidStatesMask** ()

### Properties

- bool **Triggered** [get]
- override State **FinalState** [get]
- override T **Value** [get]

### 5.45.1 Detailed Description

Definition at line 33 of file SimEvent.cs.

### 5.45.2 Member Function Documentation

#### 5.45.2.1 void DIBRIS.Dessert.Events.SimEvent< T >.Fail ( )

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | This event has already succeeded: therefore, it cannot fail anymore. |

Definition at line 63 of file SimEvent.cs.

#### 5.45.2.2 void DIBRIS.Dessert.Events.SimEvent< T >.Fail ( T *val* )

**Parameters**

| | |
|---|---|
| *val* | An object that will be sent to processes waiting for this event to occur. |

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | This event has already succeeded: therefore, it cannot fail anymore. |

Definition at line 78 of file SimEvent.cs.

#### 5.45.2.3 void DIBRIS.Dessert.Events.SimEvent< T >.Succeed ( )

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | This event has already succeeded: therefore, it cannot fail anymore. |

Definition at line 100 of file SimEvent.cs.

#### 5.45.2.4 void DIBRIS.Dessert.Events.SimEvent< T >.Succeed ( T *val* )

**Parameters**

| | |
|---|---|
| *val* | An object that will be sent to processes waiting for this event to occur. |

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | This event has already succeeded: therefore, it cannot succeed anymore. |

Definition at line 115 of file SimEvent.cs.

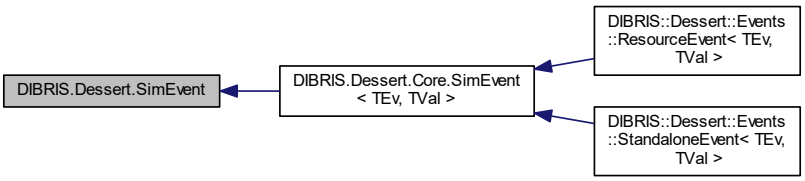
The documentation for this class was generated from the following file:

- Events/SimEvent.cs

## 5.46 DIBRIS.Dessert.SimEvent Class Reference

The interface common to each event; it should be used to declare generator methods.

Inheritance diagram for DIBRIS.Dessert.SimEvent:




Collaboration diagram for DIBRIS.Dessert.SimEvent:


### Static Public Member Functions

- static Condition< SimEvent, SimEvent > operator& (SimEvent ev1, SimEvent ev2)

  *Returns a condition event that is marked as succeeded when both events have been successful.*
- static Condition< SimEvent, SimEvent > operator| (SimEvent ev1, SimEvent ev2)

  *Returns a condition event that is marked as succeeded when any event has been successful.*
- static Condition< SimEvent, SimEvent > **operator&** (SimEvent ev, Condition< SimEvent > c)
- static Condition< SimEvent, SimEvent > **operator|** (SimEvent ev, Condition< SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent > **operator&** (SimEvent ev, Condition< SimEvent, SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent > **operator|** (SimEvent ev, Condition< SimEvent, SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent, SimEvent > **operator&** (SimEvent ev, Condition< SimEvent, SimEvent, SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent, SimEvent > **operator|** (SimEvent ev, Condition< SimEvent, SimEvent, SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent, SimEvent, SimEvent > **operator&** (SimEvent ev, Condition< SimEvent, SimEvent, SimEvent, SimEvent > c)
- static Condition< SimEvent, SimEvent, SimEvent, SimEvent, SimEvent > **operator|** (SimEvent ev, Condition< SimEvent, SimEvent, SimEvent, SimEvent > c)
- static implicit **operator bool** (SimEvent ev)

### Protected Types

- enum State : byte { State.Created = 0, **Succeeded** = 1, **Failed** = 2 }

  *Represents the state of an event.*

### Protected Member Functions

- void **SetFinalState** (State state)
- virtual State ValidStatesMask ()
- virtual void **OnEnd** ()
- abstract object **GetValue** ()

### Protected Attributes

- const State FinalStatesMask = State.Failed | State.Succeeded

  *Returns a mask with the final values that the state attribute can have.*

### Properties

- bool **InFinalState** `[get]`
- SimEnvironment Env `[get]`

  *Returns the environment in which this entity was created.*
- bool Failed `[get]`

  *Returns true if and only if event has failed; otherwise, it returns false.*
- bool Scheduled `[get, set]`

  *Returns true if and only if event has been scheduled; otherwise, it returns false.*
- bool Succeeded `[get]`

  *Returns true if and only if event has succeeded; otherwise, it returns false.*
- object Value `[get]`

  *The value returned by the event. This property contains the value that in SimPy is "sent" to the process through the generator itself; since we cannot do anything similar in .NET, we have to use this property to store that kind of values.*

- virtual bool **CanHaveParents** `[get]`
- virtual bool **CanHaveSubscribers** `[get]`
- virtual State **FinalState** `[get]`

## 5.46.1 Detailed Description

The interface common to each event; it should be used to declare generator methods.

Definition at line 425 of file ConditionOperators.cs.

## 5.46.2 Member Enumeration Documentation

### 5.46.2.1 enum DIBRIS.Dessert.SimEvent.State : byte `[strong],[protected]`

Represents the state of an event.

**Enumerator**

***Created*** Event has been created and it is waiting something to happen. Its value should be zero, since it is not used in checks.

Definition at line 45 of file SimEvent.cs.

### 5.46.3 Member Function Documentation

#### 5.46.3.1 static Condition<SimEvent, SimEvent> DIBRIS.Dessert.SimEvent.operator& ( SimEvent *ev1,* SimEvent *ev2* ) [static]

Returns a condition event that is marked as succeeded when both events have been successful.

**Parameters**

| | |
|---|---|
| *ev1* | The first event which is put in the and condition. |
| *ev2* | The second event which is put in the and condition. |

**Returns**

A condition event that is marked as succeeded when both events have been successful.

Definition at line 436 of file ConditionOperators.cs.

#### 5.46.3.2 static Condition<SimEvent, SimEvent> DIBRIS.Dessert.SimEvent.operator| ( SimEvent *ev1*, SimEvent *ev2* ) [static]

Returns a condition event that is marked as succeeded when any event has been successful.

**Parameters**

| | |
|---|---|
| *ev1* | The first event which is put in the or condition. |
| *ev2* | The second event which is put in the or condition. |

**Returns**

A condition event that is marked as succeeded when any event has been successful.

Definition at line 452 of file ConditionOperators.cs.

#### 5.46.3.3 virtual State DIBRIS.Dessert.SimEvent.ValidStatesMask ( ) [protected], [virtual]

This method is not used in any execution path of Release builds. In fact, this method is just used in Debug.Assert to enforce better integrity.

Reimplemented in DIBRIS.Dessert.Events.StandaloneEvent< TEv, TVal >, and DIBRIS.Dessert.Events.↩ ResourceEvent< TEv, TVal >.

Definition at line 260 of file SimEvent.cs.

### 5.46.4 Member Data Documentation

#### 5.46.4.1 const State DIBRIS.Dessert.SimEvent.FinalStatesMask = State.Failed | State.Succeeded [protected]

Returns a mask with the final values that the state attribute can have.

Definition at line 63 of file SimEvent.cs.

### 5.46.5 Property Documentation

#### 5.46.5.1 SimEnvironment DIBRIS.Dessert.SimEvent.Env [get]

Returns the environment in which this entity was created.

Definition at line 185 of file SimEvent.cs.

#### 5.46.5.2 bool DIBRIS.Dessert.SimEvent.Failed [get]

Returns true if and only if event has failed; otherwise, it returns false.

Definition at line 194 of file SimEvent.cs.

#### 5.46.5.3 bool DIBRIS.Dessert.SimEvent.Scheduled [get],[set]

Returns true if and only if event has been scheduled; otherwise, it returns false.

Event has been scheduled in the agenda, it will call its callbacks (if any) and activate waiting processes as soon as it will be its turn.

Definition at line 205 of file SimEvent.cs.

#### 5.46.5.4 bool DIBRIS.Dessert.SimEvent.Succeeded [get]

Returns true if and only if event has succeeded; otherwise, it returns false.

Definition at line 212 of file SimEvent.cs.

#### 5.46.5.5 object DIBRIS.Dessert.SimEvent.Value [get]

The value returned by the event. This property contains the value that in SimPy is "sent" to the process through the generator itself; since we cannot do anything similar in .NET, we have to use this property to store that kind of values.

As a rule of thumb, the value on this property will be ready only after Succeeded or Failed will be true. However, this property can always be accessed: therefore, please pay attention to the fact that this property will return a null value when a value is not ready or when an event does not have a proper value.

Definition at line 230 of file SimEvent.cs.

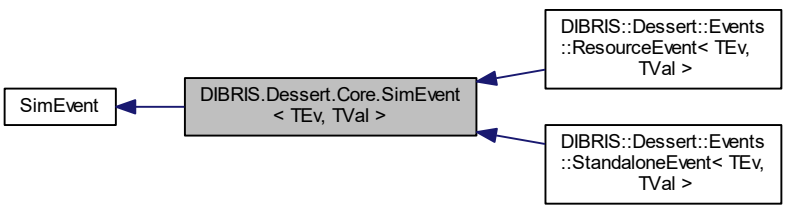
The documentation for this class was generated from the following files:

- ConditionOperators.cs
- SimEvent.cs

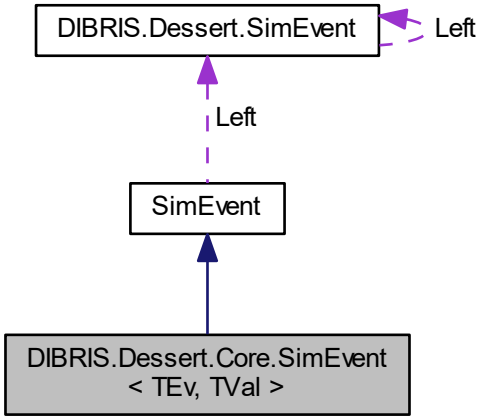
## 5.47 DIBRIS.Dessert.Core.SimEvent< TEv, TVal > Class Template Reference

A stronger typed event, which adds type notation to many properties which are untyped in SimPy.

Inheritance diagram for DIBRIS.Dessert.Core.SimEvent< TEv, TVal >:



Collaboration diagram for DIBRIS.Dessert.Core.SimEvent< TEv, TVal >:




### Protected Member Functions

- override sealed object **GetValue** ()

### Properties

- abstract new TVal Value `[get]`

  *The strongly typed value returned by the event. This property contains the value that in SimPy is "sent" to the process through the generator itself; since we cannot do anything similar in .NET, we have to use this property to store that kind of values.*

- ICollection< Action< TEv > > Callbacks `[get]`

  *Collection of functions that are called when the event is processed.*

### Additional Inherited Members

### 5.47.1 Detailed Description

A stronger typed event, which adds type notation to many properties which are untyped in SimPy.

**Template Parameters**

| | |
|---|---|
| *TEv* | The type of the event which implements this interface. |
| *TVal* | The type of the value returned by this interface. |

This class could not be named "Event" in order to maintain compatibility with Visual Basic code, where "Event" is a language keyword.

**Type Constraints**

***TEv* : *SimEvent***

***TEv* : *TEv***

***TEv* : *TVal***

Definition at line 45 of file SimEvent.cs.

### 5.47.2 Property Documentation

#### 5.47.2.1 ICollection<Action<TEv>> DIBRIS.Dessert.Core.SimEvent< TEv, TVal >.Callbacks [get]

Collection of functions that are called when the event is processed.

Definition at line 86 of file SimEvent.cs.

#### 5.47.2.2 abstract new TVal DIBRIS.Dessert.Core.SimEvent< TEv, TVal >.Value [get]

The strongly typed value returned by the event. This property contains the value that in SimPy is "sent" to the process through the generator itself; since we cannot do anything similar in .NET, we have to use this property to store that kind of values.

As a rule of thumb, the value on this property will be ready only after SimEvent.Succeeded or SimEvent.Failed will be true. However, this property can always be accessed: therefore, please pay attention to the fact that this property will return a null value when a value is not ready or when an event does not have a proper value.

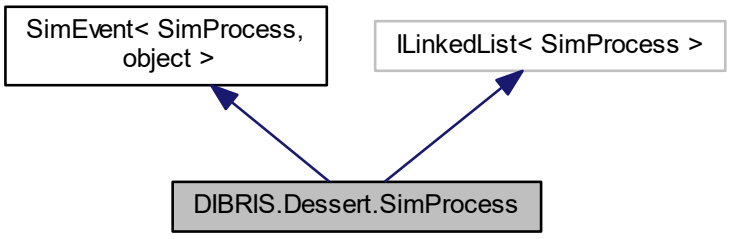
Definition at line 75 of file SimEvent.cs.

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

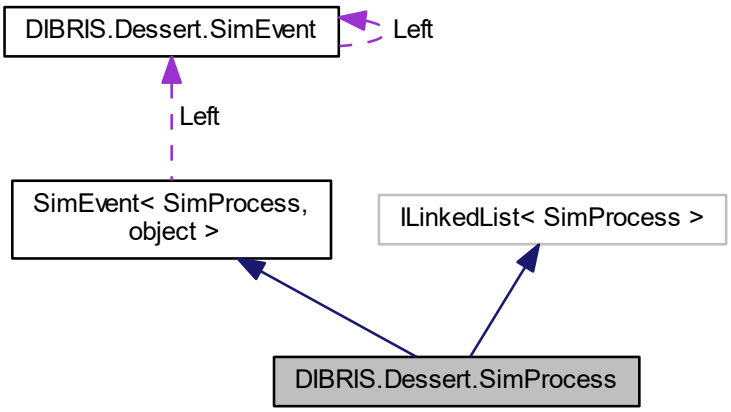
- Core/SimEvent.cs

## 5.48 DIBRIS.Dessert.SimProcess Class Reference

Inheritance diagram for DIBRIS.Dessert.SimProcess:



Collaboration diagram for DIBRIS.Dessert.SimProcess:




## Public Member Functions

- void Interrupt ()
- void Interrupt (object value)
- bool Interrupted ()
- bool Interrupted (out object value)
- bool **Preempted** ()
- bool **Preempted** (out PreemptionInfo info)

## Properties

- bool IsAlive `[get]`

  *Returns whether the process has been processed or not.*
- SimEvent Target `[get]`

  *The event that the process is currently waiting for. May be a null event if the process was just started or interrupted and it did not yet yield a new event.*
- override object **Value** `[get]`

## Additional Inherited Members

### 5.48.1 Detailed Description

This class could not be named "Process" in order to make its usage easier. In fact, commonly used "System" namespace already contains a "Process" class.

This class implements the ILinkedList<T> interface to perform an effective optimization when one event is yielded by a process only, which is a common situation.

Definition at line 48 of file SimProcess.cs.

## 5.48.2 Member Function Documentation

### 5.48.2.1 void DIBRIS.Dessert.SimProcess.Interrupt ( )

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Process it not alive or a process is trying to interrupt itself. |

Definition at line 187 of file SimProcess.cs.

#### 5.48.2.2 void DIBRIS.Dessert.SimProcess.Interrupt ( object *value* )

**Parameters**

| | |
|---|---|
| *value* | |

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Process it not alive or a process is trying to interrupt itself. |

Definition at line 204 of file SimProcess.cs.

#### 5.48.2.3 bool DIBRIS.Dessert.SimProcess.Interrupted ( )

**Returns**

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Process it not alive or a process is trying to query another process for interrupts. |

Definition at line 221 of file SimProcess.cs.

#### 5.48.2.4 bool DIBRIS.Dessert.SimProcess.Interrupted ( out object *value* )

**Parameters**

| | |
|---|---|
| *value* | |

**Returns**

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Process it not alive or a process is trying to query another process for interrupts. |

Definition at line 241 of file SimProcess.cs.

### 5.48.3 Property Documentation

#### 5.48.3.1 bool DIBRIS.Dessert.SimProcess.IsAlive [get]

Returns whether the process has been processed or not.

Definition at line 169 of file SimProcess.cs.

#### 5.48.3.2 SimEvent DIBRIS.Dessert.SimProcess.Target [get]

The event that the process is currently waiting for. May be a null event if the process was just started or interrupted and it did not yet yield a new event.

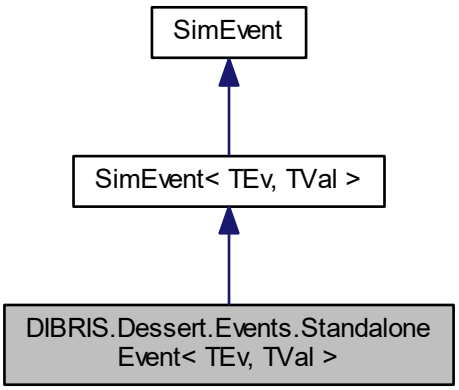
Definition at line 179 of file SimProcess.cs.

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

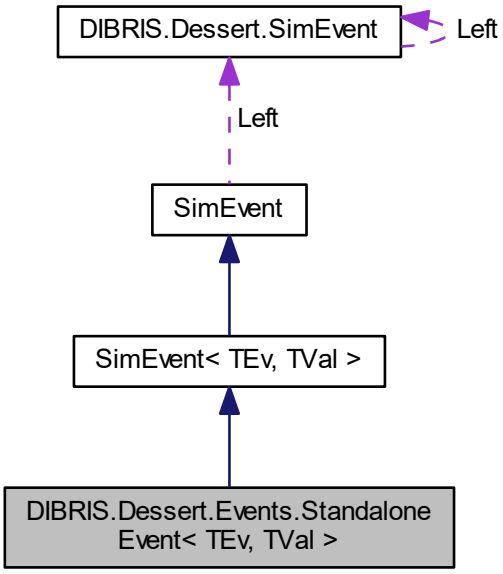
- SimProcess.cs

## 5.49 DIBRIS.Dessert.Events.StandaloneEvent< TEv, TVal > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Events.StandaloneEvent< TEv, TVal >:



Collaboration diagram for DIBRIS.Dessert.Events.StandaloneEvent< TEv, TVal >:




### Protected Member Functions

- override State ValidStatesMask ()

### Properties

- override sealed bool **CanHaveParents** `[get]`
- override sealed bool **CanHaveSubscribers** `[get]`

### Additional Inherited Members

#### 5.49.1 Detailed Description

**Type Constraints**

***TEv* : *SimEvent***

***TEv* : *TEv***

***TEv* : *TVal***

Definition at line 80 of file Templates.cs.

#### 5.49.2 Member Function Documentation

#### 5.49.2.1 override State DIBRIS.Dessert.Events.StandaloneEvent< TEv, TVal >.ValidStatesMask ( )

`[protected], [virtual]`

This method is not used in any execution path of Release builds. In fact, this method is just used in Debug.Assert to enforce better integrity.

Reimplemented from DIBRIS.Dessert.SimEvent.

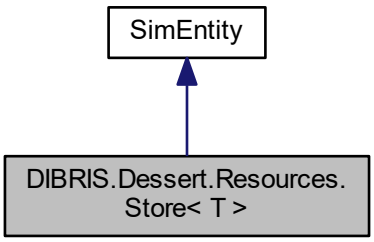
Definition at line 98 of file Templates.cs.

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

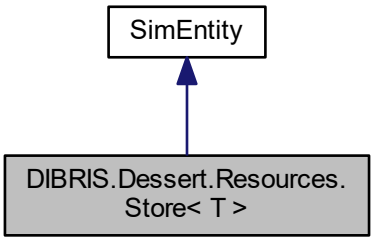
- Events/Templates.cs

## 5.50 DIBRIS.Dessert.Resources.Store< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.Store< T >:




Collaboration diagram for DIBRIS.Dessert.Resources.Store< T >:




### Classes

- class GetEvent
- class PutEvent

### Public Member Functions

- GetEvent **Get** ()
- GetEvent **Get** (double priority)
- PutEvent **Put** (T item)
- PutEvent **Put** (T item, double putPriority)
- PutEvent **Put** (T item, double putPriority, double itemPriority)

### Properties

- int **Capacity** `[get]`
- int **Count** `[get]`
- WaitPolicy **GetPolicy** `[get]`
- IEnumerable< GetEvent > **GetQueue** `[get]`
- WaitPolicy **ItemPolicy** `[get]`
- IEnumerable< T > **ItemQueue** `[get]`
- WaitPolicy **PutPolicy** `[get]`
- IEnumerable< PutEvent > **PutQueue** `[get]`

#### 5.50.1 Detailed Description

Definition at line 33 of file Store.cs.

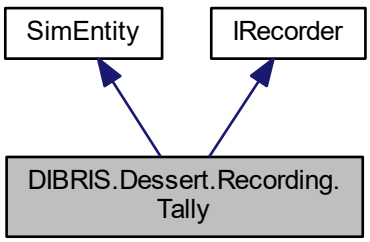
The documentation for this class was generated from the following file:

- Resources/Store.cs

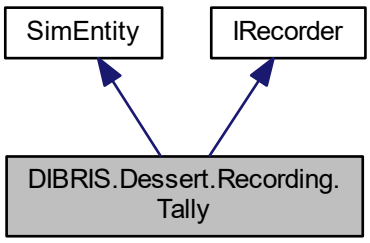
## 5.51 DIBRIS.Dessert.Recording.Tally Class Reference

An instance of this interface records enough information (such as sums and sums of squares) while the simulation runs to return simple data summaries. This has the advantage of speed and low memory use. However, they do not preserve a time-series usable in more advanced statistical analysis.

Inheritance diagram for DIBRIS.Dessert.Recording.Tally:



Collaboration diagram for DIBRIS.Dessert.Recording.Tally:




## Public Member Functions

- double Mean ()

  *Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to*
- void Observe (double sample)

  *Records the current value of the variable sample . Since time has not been specified, it is set to SimEnvironment.Now.*
- void Observe (double sample, double time)

  *Records the current value of the variable sample at given time .*
- void Reset ()

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to SimEnvironment.Now.*
- void Reset (double time)

  *Resets the observations. The recorded data is re-initialized, and the observation starting time is set to time .*
- double StdDev ()

  *Returns the standard deviation of the observations, computed as the square root of Variance.*
- double TimeMean ()

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeMean (double time)

  *Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeStdDev ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeStdDev (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double TimeVariance ()

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.*
- double TimeVariance (double time)

  *Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to time .*
- double Total ()

  *Returns the sum of the observed values.*
- double Variance ()

  *Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by*

### Properties

- int **Count** `[get]`
- double **LastTime** `[get]`
- double **StartTime** `[get]`

### 5.51.1 Detailed Description

An instance of this interface records enough information (such as sums and sums of squares) while the simulation runs to return simple data summaries. This has the advantage of speed and low memory use. However, they do not preserve a time-series usable in more advanced statistical analysis.

Monitors and tallies may not be bound to a specific SimEnvironment, in order to ease their usage in inter environment recordings; when they are unbounded their SimEntity.Env property points to a dummy environment.
However, please pay attention to the fact that both monitors and tallies are not thread safe: therefore, recall this fact when you use them in a multi threaded simulation scenario.

Definition at line 44 of file Tally.cs.

### 5.51.2 Member Function Documentation

#### 5.51.2.1 double DIBRIS.Dessert.Recording.Tally.Mean ( )

Returns the simple average of the observed values, ignoring the times at which they were made. This is equal to `Total/Count`.

**Returns**

The simple average of the observed values, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 85 of file Tally.cs.

#### 5.51.2.2 void DIBRIS.Dessert.Recording.Tally.Observe ( double *sample* )

Records the current value of the variable *sample* . Since time has not been specified, it is set to SimEnvironment.↩ Now.

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange*↩ *Exception* | Implicitly assigned time is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 90 of file Tally.cs.

#### 5.51.2.3 void DIBRIS.Dessert.Recording.Tally.Observe ( double *sample,* double *time* )

Records the current value of the variable *sample* at given *time* .

**Parameters**

| | |
|---|---|
| *sample* | The value that has to be recorded. |
| *time* | The time that will be associated with given value. |

An Monitor retains the two values as a pair (time, sample), while a Tally uses them to update the accumulated statistics.

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than the last observation time. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 95 of file Tally.cs.

#### 5.51.2.4 void DIBRIS.Dessert.Recording.Tally.Reset ( )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to Sim↩ Environment.Now.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 100 of file Tally.cs.

#### 5.51.2.5 void DIBRIS.Dessert.Recording.Tally.Reset ( double *time* )

Resets the observations. The recorded data is re-initialized, and the observation starting time is set to *time* .

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 105 of file Tally.cs.

#### 5.51.2.6 double DIBRIS.Dessert.Recording.Tally.StdDev ( )

Returns the standard deviation of the observations, computed as the square root of Variance.

**Returns**

The standard deviation of the observations, computed as the square root of Variance.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 110 of file Tally.cs.

#### 5.51.2.7 double DIBRIS.Dessert.Recording.Tally.TimeMean ( )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 115 of file Tally.cs.

#### 5.51.2.8 double DIBRIS.Dessert.Recording.Tally.TimeMean ( double *time* )

Returns the time-weighted mean, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 120 of file Tally.cs.

#### 5.51.2.9 double DIBRIS.Dessert.Recording.Tally.TimeStdDev ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 126 of file Tally.cs.

#### 5.51.2.10 double DIBRIS.Dessert.Recording.Tally.TimeStdDev ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 131 of file Tally.cs.

#### 5.51.2.11 double DIBRIS.Dessert.Recording.Tally.TimeVariance ( )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to current time.

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to current time.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 136 of file Tally.cs.

#### 5.51.2.12 double DIBRIS.Dessert.Recording.Tally.TimeVariance ( double *time* )

Returns the time-weighted variance, calculated from time 0 (or the last time Reset() was called) to *time* .

**Returns**

The time-weighted average, calculated from time 0 (or the last time Reset() was called) to *time* .

**Exceptions**

| | |
|---|---|
| *ArgumentOutOfRange↩ Exception* | *time* is less than StartTime. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 141 of file Tally.cs.

#### 5.51.2.13 double DIBRIS.Dessert.Recording.Tally.Total ( )

Returns the sum of the observed values.

**Returns**

The sum of the observed values.

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 148 of file Tally.cs.

#### 5.51.2.14 double DIBRIS.Dessert.Recording.Tally.Variance ( )

Returns the sample variance of the observations, ignoring the times at which they were made. If an unbiased estimate of the population variance is desired, the sample variance should be multiplied by

`Count/(Count - 1).`

**Returns**

The sample variance of the observations, ignoring the times at which they were made.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | There are no observations. |

Implements DIBRIS.Dessert.Recording.IRecorder.

Definition at line 153 of file Tally.cs.

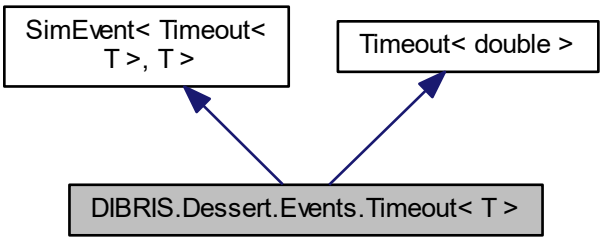
The documentation for this class was generated from the following file:

- Recording/Tally.cs

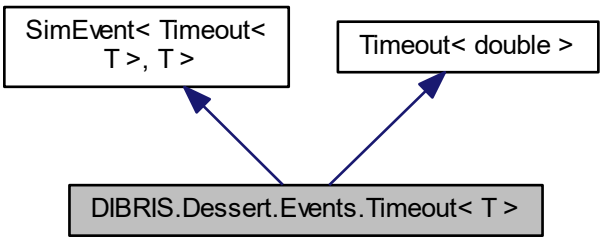
## 5.52 DIBRIS.Dessert.Events.Timeout< T > Class Template Reference

An event that is scheduled with a certain delay after its creation.
This event can be used by processes to wait (or hold their state) for delay time steps. It is immediately scheduled at Env.Now + delay and has thus (in contrast to SimEvent<T>) no Success() or Fail() methods.

Inheritance diagram for DIBRIS.Dessert.Events.Timeout< T >:




Collaboration diagram for DIBRIS.Dessert.Events.Timeout< T >:




### Public Attributes

- override T **Value** => _value

### Protected Member Functions

- override void **OnEnd** ()

### Properties

- double Delay `[get]`
  *The delay at which this event was scheduled.*

### 5.52.1 Detailed Description

An event that is scheduled with a certain delay after its creation.
This event can be used by processes to wait (or hold their state) for delay time steps. It is immediately scheduled at Env.Now + delay and has thus (in contrast to SimEvent<T>) no Success() or Fail() methods.

Definition at line 39 of file Timeout.cs.

### 5.52.2 Property Documentation

#### 5.52.2.1 double DIBRIS.Dessert.Events.Timeout< T >.Delay [get]

The delay at which this event was scheduled.

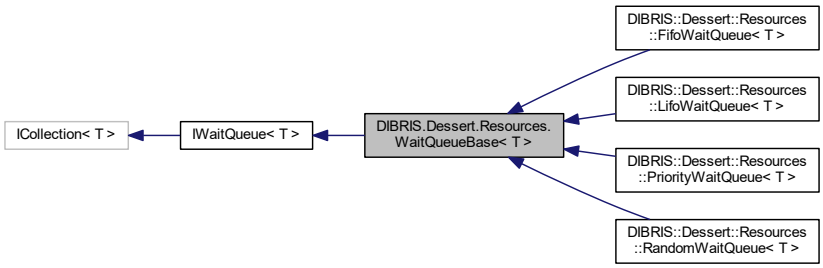
Definition at line 76 of file Timeout.cs.

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

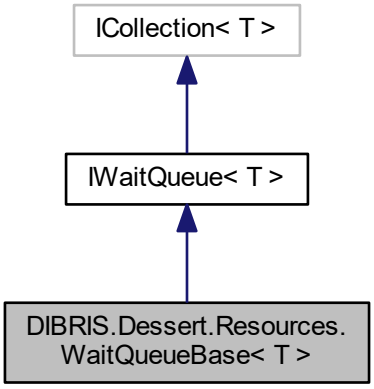
- Events/Timeout.cs

## 5.53 DIBRIS.Dessert.Resources.WaitQueueBase< T > Class Template Reference

Inheritance diagram for DIBRIS.Dessert.Resources.WaitQueueBase< T >:




Collaboration diagram for DIBRIS.Dessert.Resources.WaitQueueBase< T >:




### Public Member Functions

- void **Add** (T item)
- void **Clear** ()

- void **CopyTo** (T[ ] array, int arrayIndex)
- abstract void **Add** (T item, double priority)
- abstract bool **Contains** (T item)
- abstract IEnumerator< T > **GetEnumerator** ()
- abstract bool **Remove** (T item)
- abstract T **RemoveFirst** ()

### Properties

- bool **IsReadOnly** `[get]`
- abstract int **Count** `[get]`
- abstract T **First** `[get]`
- abstract WaitPolicy **Policy** `[get]`

### 5.53.1 Detailed Description

Definition at line 48 of file WaitQueues.cs.

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

- Resources/WaitQueues.cs

# Index