TAP Auction Site Component

Generated by Doxygen 1.9.3

Chapter 1

TAP Auction Site Component

1.1 Introduction

The namespace TAP22_23.AuctionSite contains the declarations of a set of interfaces modeling the required types for managing toy auction sites (think ebay, think smaller, smaller, smaller...).

Implementations of the interfaces in AlarmClock.Interfaces will be provided by the AuctionSite clients and are not part of this project.

Only interfaces in AuctionSite.Interfaces are implemented by the AuctionSite component.

Using the IHostFactory a client of this component can initialize or load a hosting service for auction sites.

On such a hosting system IHost, new sites can be created and existing ones can be loaded and made available to their final users.

An ISite represents a site for online auctions, managing users, their sessions, and their auctions. It has a unique name, which is a non empty nor null string, and defines

- the timezone used to get the time;
- the minimum increment allowed in bidding;
- the time out of inactive sessions.

It is the root of an aggregate, in the sense that it owns its users, sessions and auctions, that have no meaning outside their site.

Users, represented by IUser, must have a valid session to act on a site, and no user can have two valid sessions at the same time.

Sessions, represented by ISession, start with a login, and may be deleted by an explicit logout, or may time out because of a prolonged inactivity. Each time users access their sessions by an explicit login, by creating an auction or by bidding on an auction, the expiration time is reset to the site expiration time. Every five minutes expired sessions must be deleted from the database.

Time management is provided by a required component implementing the interface TAP22-23.AlarmClock.↩ Interfaces. The main type in such component is IAlarmClock, and it represents a clock synchronized on a specific time zone. IAlarmClock has methods for checking the current time and setting an alarm, that is, firing the ringing event at a given time.

This is just an overview; each type and each method has its own, more detailed, specific documentation.

1.2 Global requirements

Names and Id, when provided for a type, are unique within the context where the type is meaningful. Thus, for instance, the name of an auction site is unique in a host, as well as the name of a user within an auction site (but the same user name can be repeated in different auction sites).

Any attempt to create two different elements of the same type with the same name (or Id) must fail by throwing AuctionSiteNameAlreadyInUseException.

Each method that receives:

- a `null` argument must throw the exception `AuctionSiteArgumentNullException`, unless explicitly specified
- a string that is too short or too long must throw `AuctionSiteArgumentException`; the allowed string length ranges are contained in DomainConstraints

Any method invocation on a deleted object, that is an object whose persistent counterpart has been removed, must throw `AuctionSiteInvalidOperationException`.

Any generic failure to persist or retrieve the data to/from the DB must be communicated by throwing `AuctionSiteUnavailableDbException`.

Any detected attempt to save an entity on the DB which has been concurrently modified must be communicated by throwing `AuctionSiteConcurrentChangeException`. Concurrency management is not required for a project to be acceptable. But, it is an element of evaluation, and if you want to do it, then you must use `AuctionSiteConcurrentChangeException`.

Your implementation must not throw any exception that has not been explicitly listed here.

Please note that these requirements are intentionally *not* repeated for each and every method of the specification, and must be met by all your methods (implementing the interfaces described by this document; private methods can behave as they please ;-))

1.3 Implementation requirements

You're required to provide a Visual Studio 2022 solution for .NET 6 containing an implementation of this specification. The solution must be pushed on your personal repository created on GitHub when you accept the invitation to the project assignment.

All type declarations must be contained in a namespace whose name correspond to your family name (if it includes unacceptable chars, replace them by _).

With the obvious exception of connecting with the DB referenced by the connection string passed to the methods of the IHostFactory, it is forbidden to open or create files/registry keys/..., or establish database/network/... connections and so on.

To interact with the databases, you must use EF Core 6 and your DbContext class must extend TAP22-23.AuctionSite.Interfaces.TapDb

Your DLL must not depend on any library, other than the TAP22-23.AuctionSite.Interfaces, TAP22-23.AlarmClock.Interfaces, EF Core 6 and standard .NET assemblies. You can use other libraries only if they have been explicitly approved in the TAP Forum by the teacher (thus, if you think there is a useful library out there, just ask on the forum if you can use it... the answer will most probably be yes, but you have nonetheless to explicitly ask for it).

Before delivering your work, please keep in mind that passing the tests is a minimum requirement only. The fact that your implementation passes these test does not imply, of course, its correctness. Indeed, many other tests and code inspection will be used to evaluate your work and it will be evaluated *once*.

Chapter 2

Namespace Documentation

2.1 TAP22_23.AlarmClock.Interface Namespace Reference

An elementary time management component needed by the AuctionSite component.

Classes

- interface IAlarm

  An alarm, raising a RingingEvent every tot seconds, where tot is defined by the alarm object constructor
- interface IAlarmClock

  A clock synchronized on the given Timezone
- interface IAlarmClockFactory

  The alarm clock factory

2.1.1 Detailed Description

An elementary time management component needed by the AuctionSite component.

This component provides types to represent alarm clock and alarms.

The implementation of the interfaces defined by this component is not part of the project. Objects implementing them will be provided by the client code of the AuctionSite component via dependency injection

2.2 TAP22_23.AuctionSite.Interface Namespace Reference

Classes

- class AuctionSiteArgumentException

  Thrown when an argument of a method call violates the method preconditions
- class AuctionSiteArgumentNullException

  Thrown when an unacceptable null argument is used for a method call
- class AuctionSiteArgumentOutOfRangeException

  Thrown when an argument of a method call does not meet the expected value range for the corresponding parameter

- class AuctionSiteConcurrentChangeException

  Thrown to notify an attempt to save an entity that has been modified concurrently in the DB (concurrency management is not required; but, if you want to do it use this exception)
- class AuctionSiteException

  The interfaces provided by the AuctionSite component.
- class AuctionSiteInexistentNameException

  Thrown to notify an attempt to use its name to access an entity that is not available in the database
- class AuctionSiteInvalidOperationException

  Thrown to notify an attempt at invoking an operation on an object in an invalid state (for instance a deleted entity)
- class AuctionSiteNameAlreadyInUseException

  Thrown to notify an attempt to create an entity whose name is already in use. For instance, a site with the same name of another site managed by the same host, or a user with the same name of another user on the same site.
- class AuctionSiteUnavailableDbException

  Thrown to notify that no connection with the database has been established.
- class AuctionSiteUnavailableTimeMachineException

  Thrown to notify an attempt to create an auction with expiration date in the past.
- class DomainConstraints

  Numeric constants used to express constraints on names, passwords, and time zone
- interface IAuction

  The auctions managed by sites. Equals on auctions must be true iff the two auctions belong to the same site and have the same Id.
- interface IHost

  The hosting service A Host own a database and uses it to save the data of the sites it manages. Any Host may create and load its sites.
- interface IHostFactory

  This component factory. All its methods have a parameter `connectionString` representing the connection string of a Microsoft SQL Server DB, used to permanently store the data of the auction sites managed by a specific Host and to implicitly characterize the Host. Many different objects of type IHost may at the same time represent the same Host, concurrently working on the same database.
- interface ISession

  The sessions of the users on the site. It may become invalid (expire) if the user is idle for the expiration time set for the site. Equals on sessions must be true iff the two sessions have the same Id.
- interface ISite

  The auction sites. Equals on sites must be true iff the two sites have the same Name.
- interface IUser

  The users of the auction system. Equals on users must be true iff the two users belong to the same site and have the same Username.
- class TapDbContext

  To be extended by the component DbContext, to simplify advanced testing

Chapter 3

Class Documentation - Alarm clock component

3.1 IAlarmClockFactory Interface Reference

The alarm clock factory

Public Member Functions

- IAlarmClock InstantiateAlarmClock (int timezone)

  Instantiate a new alarm clock with the given timezone .

3.1.1 Detailed Description

The alarm clock factory

3.1.2 Member Function Documentation

3.1.2.1 InstantiateAlarmClock()

```
IAlarmClock InstantiateAlarmClock (
            int timezone )
```

Instantiate a new alarm clock with the given *timezone* .

Parameters

timezone	An integer between -12 and 12 (inclusive).

Returns

A new alarm clock.

Exceptions

AuctionSiteArgumentOutOfRangeException	When *timezone* is not in range.

3.2 IAlarmClock Interface Reference

A clock synchronized on the given Timezone

Public Member Functions

- IAlarm InstantiateAlarm (int frequencyInMs)
  Instantiate an alarm.

Properties

- int Timezone `[get]`
  The time zone for an alarm clock
- DateTime Now `[get]`
  The current time

3.2.1 Detailed Description

A clock synchronized on the given Timezone

3.2.2 Member Function Documentation

3.2.2.1 InstantiateAlarm()

```
IAlarm InstantiateAlarm (
            int frequencyInMs )
```

Instantiate an alarm.

Parameters

frequencyInMs	Frequency in milliseconds.

Returns

A new alarm.

Exceptions

AuctionSiteArgumentOutOfRangeException	If the frequency is not positive.

3.2.3 Property Documentation

3.2.3.1 Now

```
DateTime Now  [get]
```

The current time

3.2.3.2 Timezone

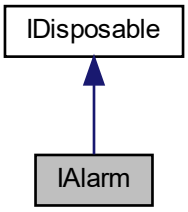
```
int Timezone  [get]
```

The time zone for an alarm clock

3.3 IAlarm Interface Reference

An alarm, raising a RingingEvent every tot seconds, where tot is defined by the alarm object constructor

Inheritance diagram for IAlarm:



Events

- Action RingingEvent

3.3.1 Detailed Description

An alarm, raising a RingingEvent every tot seconds, where tot is defined by the alarm object constructor

3.3.2 Event Documentation

3.3.2.1 RingingEvent

```
Action RingingEvent
```

Chapter 4

Class Documentation - Auction site component

4.1 DomainConstraints Class Reference

Numeric constants used to express constraints on names, passwords, and time zone

Static Public Attributes

- const int MinSiteName = 1

  The minimal length of a string to be a well formed site name
- const int MaxSiteName = 128

  The maximal length of a string to be a well formed site name
- const int MinUserName = 3

  The minimal length of a string to be a well formed user name
- const int MaxUserName = 64

  The maximal length of a string to be a well formed user name
- const int MinUserPassword = 4

  The minimal length of a string to be an acceptable password
- const int MinTimeZone = -12

  The minimal value acceptable as a time zone
- const int MaxTimeZone = 12

  The maximal value acceptable as a time zone

4.1.1 Detailed Description

Numeric constants used to express constraints on names, passwords, and time zone

4.1.2 Member Data Documentation

4.1.2.1 MaxSiteName

```
const int MaxSiteName = 128  [static]
```

The maximal length of a string to be a well formed site name

4.1.2.2 MaxTimeZone

```
const int MaxTimeZone = 12  [static]
```

The maximal value acceptable as a time zone

4.1.2.3 MaxUserName

```
const int MaxUserName = 64  [static]
```

The maximal length of a string to be a well formed user name

4.1.2.4 MinSiteName

```
const int MinSiteName = 1  [static]
```

The minimal length of a string to be a well formed site name

4.1.2.5 MinTimeZone

```
const int MinTimeZone = -12  [static]
```

The minimal value acceptable as a time zone

4.1.2.6 MinUserName

```
const int MinUserName = 3  [static]
```

The minimal length of a string to be a well formed user name

4.1.2.7 MinUserPassword

```
const int MinUserPassword = 4  [static]
```

The minimal length of a string to be an acceptable password

4.2 IHostFactory Interface Reference

This component factory. All its methods have a parameter `connectionString` representing the connection string of a Microsoft SQL Server DB, used to permanently store the data of the auction sites managed by a specific Host and to implicitly characterize the Host. Many different objects of type IHost may at the same time represent the same Host, concurrently working on the same database.

Public Member Functions

- void CreateHost (string connectionString)

  Creates a new DB (dropping existing previous version, if any), and initialize it with all the necessary DB elements for the Host.
- IHost LoadHost (string connectionString, IAlarmClockFactory alarmClockFactory)

  Yields the Host managing a group of Sites having their data resident on the same database.

4.2.1 Detailed Description

This component factory. All its methods have a parameter `connectionString` representing the connection string of a Microsoft SQL Server DB, used to permanently store the data of the auction sites managed by a specific Host and to implicitly characterize the Host. Many different objects of type IHost may at the same time represent the same Host, concurrently working on the same database.

4.2.2 Member Function Documentation

4.2.2.1 CreateHost()

```
void CreateHost (
            string connectionString )
```

Creates a new DB (dropping existing previous version, if any), and initialize it with all the necessary DB elements for the Host.

Parameters

connectionString	A valid connection string for a Microsoft SQL Server DB.

Exceptions

AuctionSiteArgumentNullException	If *connectionString* is null.
AuctionSiteUnavailableDbException	If *connectionString* is (non-null but) malformed, the DB server is not responding or returns an unexpected error.

4.2.2.2 LoadHost()

```
IHost LoadHost (
            string connectionString,
            IAlarmClockFactory alarmClockFactory )
```

Yields the Host managing a group of Sites having their data resident on the same database.

Parameters

connectionString	The connection string.
alarmClockFactory	The alarm clock factory.

Returns

A new instance for the host based on that database.

Exceptions

AuctionSiteUnavailableDbException	The connection string is (non-null but) malformed, the DB server is not responding or returns an unexpected error.
AuctionSiteArgumentNullException	If *connectionString* or *alarmClockFactory* are null.

4.3 IHost Interface Reference

The hosting service A Host own a database and uses it to save the data of the sites it manages. Any Host may create and load its sites.

Public Member Functions

- IEnumerable<(string Name, int TimeZone)> GetSiteInfos ()

  Yields the names and corresponding time zones of managed sites.
- void CreateSite (string name, int timezone, int sessionExpirationTimeInSeconds, double minimumBid↩ Increment)

  Create a new site, identified by its name.
- ISite LoadSite (string name)

  Yields the ISite object corresponding to an existing Site.

4.3.1 Detailed Description

The hosting service A Host own a database and uses it to save the data of the sites it manages. Any Host may create and load its sites.

4.3.2 Member Function Documentation

4.3.2.1 CreateSite()

```
void CreateSite (
            string name,
            int timezone,
            int sessionExpirationTimeInSeconds,
            double minimumBidIncrement )
```

Create a new site, identified by its name.

Parameters

name	The name of the site (a unique identifier, whose length is between DomainConstraints.MinSiteName and DomainConstraints.MaxSiteName chars).
timezone	The timezone, an integer between DomainConstraints.MinTimeZone and DomainConstraints.MaxTimeZone (inclusive).
sessionExpirationTimeInSeconds	Session timeout, in seconds, a positive number.
minimumBidIncrement	Minimum bid increment, a positive number.

Exceptions

AuctionSiteNameAlreadyInUseException	Thrown if the name of the site is already in use as name of an existing site.
AuctionSiteArgumentNullException	If *name* is null.
AuctionSiteArgumentException	If *name* is not null, but its length is (strictly) smaller than DomainConstraints.MinSiteName or (strictly) larger than DomainConstraints.MaxSiteName.
AuctionSiteArgumentOutOfRangeException	If *timezone* is (strictly) smaller than DomainConstraints.MinTimeZone or (strictly) larger than DomainConstraints.MaxTimeZone or if *sessionExpirationTimeInSeconds* or *minimumBidIncrement* are not positive.
AuctionSiteUnavailableDbException	The DB server is not responding or returns an unexpected error.

4.3.2.2 GetSiteInfos()

```
IEnumerable<(string Name, int TimeZone)> GetSiteInfos ( )
```

Yields the names and corresponding time zones of managed sites.

Returns

The names of the managed sites and their time zones.

Exceptions

AuctionSiteUnavailableDbException	If the DB server is not responding or returns an unexpected error.

4.3.2.3 LoadSite()

```
ISite LoadSite (
            string name )
```

Yields the ISite object corresponding to an existing Site.

Parameters

name	The name of the site.

Returns

A new instance for the site.

Exceptions

AuctionSiteUnavailableDbException	The DB server is not responding or returns an unexpected error.
AuctionSiteArgumentNullException	If *name* is null.
AuctionSiteArgumentException	If *name* is not null, but its length is (strictly) smaller than DomainConstraints.MinSiteName or (strictly) larger than DomainConstraints.MaxSiteName.
AuctionSiteInexistentNameException	If *name* is a non-null string of the correct length, but the corresponding site is not present in the DB.

4.4 ISite Interface Reference

The auction sites. Equals on sites must be true iff the two sites have the same Name.

Public Member Functions

- IEnumerable< IUser > ToyGetUsers ()

  Yields all the users of the site. In a realistic example, this method would be more complex, using some sort of pagination

- IEnumerable< ISession > ToyGetSessions ()

  Yields all the sessions of the site. In a realistic example, this method would be more complex, using some sort of pagination
- IEnumerable< IAuction > ToyGetAuctions (bool onlyNotEnded)

  Yields all the (not yet ended) auctions of the site. In a realistic example, this method would be more complex, using some sort of pagination
- ISession? Login (string username, string password)

  Yields the session for the user, new iff no valid session for him/her exists. No user can have two valid sessions on the same site at the same time.
- void CreateUser (string username, string password)

  Add a user of the site.
- void Delete ()

  Disposes of the site and all its associated resources.
- DateTime Now ()

  Returns the current time

Properties

- string Name `[get]`

  The name of the auction site.
- int Timezone `[get]`

  The timezone of the auction site.
- int SessionExpirationInSeconds `[get]`

  The number of seconds needed for the session of an idle user to time out. A positive number
- double MinimumBidIncrement `[get]`

  The minimum amount allowed as increment (from the starting price) for a bid. A positive number

4.4.1 Detailed Description

The auction sites. Equals on sites must be true iff the two sites have the same Name.

4.4.2 Member Function Documentation

4.4.2.1 CreateUser()

```
void CreateUser (
            string username,
            string password )
```

Add a user of the site.

Parameters

username	User login (minimum length=DomainConstraints.MinUserName, maximum=DomainConstraints.MaxUserName chars).
password	User password (minimum length=DomainConstraints.MinUserPassword chars).

Exceptions

AuctionSiteNameAlreadyInUseException	If *username* is already in use for a user of the site.
AuctionSiteArgumentNullException	If *username* or *password* are null.
AuctionSiteArgumentException	If *username* is not null, but its length is (strictly) smaller than DomainConstraints.MinUserName or (strictly) larger than DomainConstraints.MaxUserName, or if *password* is not null, but its length is (strictly) smaller than DomainConstraints.MinUserPassword.

4.4.2.2 Delete()

```
void Delete ( )
```

Disposes of the site and all its associated resources.

4.4.2.3 Login()

```
ISession?  Login (
             string username,
             string password )
```

Yields the session for the user, new iff no valid session for him/her exists. No user can have two valid sessions on the same site at the same time.

Parameters

username	User login.
password	User password.

Returns

The session for the user or null if *username* and *password* do not correspond to a user of the site.

Exceptions

AuctionSiteArgumentNullException	If *username* or *password* are null.
AuctionSiteArgumentException	If *username* is not null, but its length is (strictly) smaller than DomainConstraints.MinUserName or (strictly) larger than DomainConstraints.MaxUserName, or if *password* is not null, but its length is (strictly) smaller than DomainConstraints.MinUserPassword.

4.4.2.4 Now()

```
DateTime Now ( )
```

Returns the current time

Returns

The current time, as provided by the internal IAlarmClock

4.4.2.5 ToyGetAuctions()

```
IEnumerable< IAuction > ToyGetAuctions (
            bool onlyNotEnded )
```

Yields all the (not yet ended) auctions of the site. In a realistic example, this method would be more complex, using some sort of pagination

Parameters

onlyNotEnded	If true, only the auctions not yet ended are taken into account.

Returns

If not *onlyNotEnded* , all the auctions, otherwise only those not yet ended.

4.4.2.6 ToyGetSessions()

```
IEnumerable< ISession > ToyGetSessions ( )
```

Yields all the sessions of the site. In a realistic example, this method would be more complex, using some sort of pagination

Returns

All the sessions of the site.

4.4.2.7 ToyGetUsers()

```
IEnumerable< IUser > ToyGetUsers ( )
```

Yields all the users of the site. In a realistic example, this method would be more complex, using some sort of pagination

Returns

All the users of the site.

4.4.3 Property Documentation

4.4.3.1 MinimumBidIncrement

```
double MinimumBidIncrement  [get]
```

The minimum amount allowed as increment (from the starting price) for a bid. A positive number

4.4.3.2 Name

```
string Name  [get]
```

The name of the auction site.

4.4.3.3 SessionExpirationInSeconds

```
int SessionExpirationInSeconds  [get]
```

The number of seconds needed for the session of an idle user to time out. A positive number

4.4.3.4 Timezone

```
int Timezone  [get]
```

The timezone of the auction site.

4.5 IUser Interface Reference

The users of the auction system. Equals on users must be true iff the two users belong to the same site and have the same Username.

Public Member Functions

- IEnumerable< IAuction > WonAuctions ()

  Yields the auctions won by the user.
- void Delete ()

  Disposes of the user and all its resources. Users cannot be deleted if they are owners or (current) winners of still open auctions. Thus, a call to Delete on a user who is owner or (current) winner of auctions yet to be adjudicated must throw `AuctionSiteInvalidOperationException` Ended owned auctions are disposed of, if any. Ended won auctions are updated, and the information of the winner is removed, if any.

Properties

- string Username [get]

  Gets the unique key used to identify the user of a specific site in the system. It is a string of at least DomainConstraints.MinUserName, and at most DomainConstraints.MaxUserName characters. The same username may be used by different users on different sites.

4.5.1 Detailed Description

The users of the auction system. Equals on users must be true iff the two users belong to the same site and have the same Username.

4.5.2 Member Function Documentation

4.5.2.1 Delete()

```
void Delete ( )
```

Disposes of the user and all its resources. Users cannot be deleted if they are owners or (current) winners of still open auctions. Thus, a call to Delete on a user who is owner or (current) winner of auctions yet to be adjudicated must throw AuctionSiteInvalidOperationException Ended owned auctions are disposed of, if any. Ended won auctions are updated, and the information of the winner is removed, if any.

4.5.2.2 WonAuctions()

```
IEnumerable< IAuction > WonAuctions ( )
```

Yields the auctions won by the user.

Returns

The auctions won by the user (that is, all the ended auctions of this site where this user is the highest bidder).

4.5.3 Property Documentation

4.5.3.1 Username

```
string Username [get]
```

Gets the unique key used to identify the user of a specific site in the system. It is a string of at least DomainConstraints.MinUserName, and at most DomainConstraints.MaxUserName characters. The same username may be used by different users on different sites.

4.6 ISession Interface Reference

The sessions of the users on the site. It may become invalid (expire) if the user is idle for the expiration time set for the site. Equals on sessions must be true iff the two sessions have the same Id.

Public Member Functions

- void Logout ()

  Deletes the session and disposes of all associated resources, if any.
- IAuction CreateAuction (string description, DateTime endsOn, double startingPrice)

  Yields an auction for the described object/service. As a side effect, the expiration time of the session is reset (to the same value as if the session was newly created).

Properties

- string Id `[get]`

  Gets the unique key used to identify the sessions.
- DateTime ValidUntil `[get]`

  Gets the current expiration time of the session.
- IUser User `[get]`

  Gets the user owner of the session.

4.6.1 Detailed Description

The sessions of the users on the site. It may become invalid (expire) if the user is idle for the expiration time set for the site. Equals on sessions must be true iff the two sessions have the same Id.

4.6.2 Member Function Documentation

4.6.2.1 CreateAuction()

```
IAuction CreateAuction (
            string description,
            DateTime endsOn,
            double startingPrice )
```

Yields an auction for the described object/service. As a side effect, the expiration time of the session is reset (to the same value as if the session was newly created).

Parameters

description	The description of the object/service for sale, a non-null and non-empty string.
endsOn	The expiring time of the auction; no bid will be accepted after it.
startingPrice	The starting price of the auction, a non-negative number; the first bid must be greater than or equal to this value.

Returns

Returns the newly created auction, whose Id is an automatically-generated unique identifier.

Exceptions

AuctionSiteInvalidOperationException	The session is not valid (or the corresponding permanent object does not exist anymore).
AuctionSiteArgumentNullException	If *description* is null.
AuctionSiteArgumentException	If *description* is not null but empty.
AuctionSiteArgumentOutOfRangeException	If *startingPrice* is negative
AuctionSiteUnavailableTimeMachineException	If *endsOn* precedes the current time (according to the IAlarmClock of the ISite)

4.6.2.2 Logout()

```
void Logout ( )
```

Deletes the session and disposes of all associated resources, if any.

4.6.3 Property Documentation

4.6.3.1 Id

```
string Id  [get]
```

Gets the unique key used to identify the sessions.

4.6.3.2 User

```
IUser User  [get]
```

Gets the user owner of the session.

4.6.3.3 ValidUntil

```
DateTime ValidUntil  [get]
```

Gets the current expiration time of the session.

4.7 IAuction Interface Reference

The auctions managed by sites. Equals on auctions must be true iff the two auctions belong to the same site and have the same Id.

Public Member Functions

- IUser? CurrentWinner ()

  Returns the user, if any, who has submitted the highest bid so far. In case no bids have been offered yet, it returns null. It may also return null in case of closed auction whose winner has been deleted from the site (after the auction ended).
- double CurrentPrice ()

  Returns the current price, which is the lowest amount needed to best the second highest bid if two or more bids have been offered; otherwise, it coincides with the starting price.
- void Delete ()

  Disposes of the auction and all associated resources, if any.
- bool Bid (ISession session, double offer)

  Makes a bid for this auction on behalf of the session owner; only possible for still open auctions.

Properties

- int Id `[get]`

  Gets the unique key used to identify the auctions.
- IUser Seller `[get]`

  Gets the user who is selling the object/service.
- string Description `[get]`

  Gets the description of the offered object/service.
- DateTime EndsOn `[get]`

  Gets the expiring time of the auction; no bid will be accepted after it.

4.7.1 Detailed Description

The auctions managed by sites. Equals on auctions must be true iff the two auctions belong to the same site and have the same Id.

4.7.2 Member Function Documentation

4.7.2.1 Bid()

```
bool Bid (
            ISession session,
            double offer )
```

Makes a bid for this auction on behalf of the session owner; only possible for still open auctions.

Parameters

session	A valid session. The expiration time of the session is reset to the expiration time of the site for each bid using correct parameters, that is, not throwing an exception, disregarding whether the bid is accepted or not.
offer	The amount offered as bid, that is, the maximum amount the user is willing to pay for the item. This maximum amount must remain confidential.

Returns

True iff the bid is accepted, so that the status of the auction is changed. The bid is rejected, so that the result is false, iff either of the following occurs

- the bidder is (already) the current winner and *offer* is lower than the maximum offer increased by minimumBidIncrement
- the bidder is not the current winner and *offer* is lower than the current price
- the bidder is not the current winner and *offer* is lower than the current price increased by minimumBid↩ Increment AND this is not the first bid

In all other cases, the bid is accepted, the result is true, and the status of the auction is changed as follows:

- if this is the first bid, then the maximum offer is set to *offer* , the current price is not changed (that is, it remains the starting price), and the bidder becomes the current winner;
- if the bidder was already winning this auction, the maximum offer is set to *offer* , current price and current winner are unchanged;
- if this is NOT the first bid, the bidder is NOT the current winner, and *offer* is higher than the current maximum offer, in the following denoted by CMO, then the current price is set to the minimum between *offer* and CMO+minimumBidIncrement, the maximum offer is set to *offer* , and the bidder becomes the current winner;
- if this is NOT the first bid, the bidder is NOT the current winner, and *offer* is NOT higher than the current maximum offer, in the following denoted by CMO, then the current price is set to the minimum between CMO and *offer* +minimumBidIncrement, and the current winner does not change.

Exceptions

AuctionSiteInvalidOperationException	The auction is already closed (or the corresponding permanent object does not exist anymore).
AuctionSiteArgumentOutOfRangeException	If *offer* is negative.
AuctionSiteArgumentNullException	If *session* is null.
AuctionSiteArgumentException	If *session* is not null and, one of the following conditions is true: • the session is not valid anymore; • the logged user is also the Seller of this auction; • the logged user is a user of a site different from the site of the Seller.

4.7.2.2 CurrentPrice()

```
double CurrentPrice ( )
```

Returns the current price, which is the lowest amount needed to best the second highest bid if two or more bids have been offered; otherwise, it coincides with the starting price.

4.7.2.3 CurrentWinner()

```
IUser?  CurrentWinner ( )
```

Returns the user, if any, who has submitted the highest bid so far. In case no bids have been offered yet, it returns null. It may also return null in case of closed auction whose winner has been deleted from the site (after the auction ended).

4.7.2.4 Delete()

```
void Delete ( )
```

Disposes of the auction and all associated resources, if any.

4.7.3 Property Documentation

4.7.3.1 Description

```
string Description  [get]
```

Gets the description of the offered object/service.

4.7.3.2 EndsOn

```
DateTime EndsOn  [get]
```

Gets the expiring time of the auction; no bid will be accepted after it.

4.7.3.3 Id

```
int Id  [get]
```

Gets the unique key used to identify the auctions.

4.7.3.4 Seller

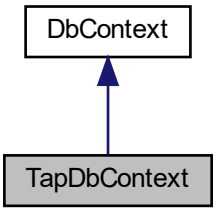
```
IUser Seller [get]
```

Gets the user who is selling the object/service.

4.8 TapDbContext Class Reference

To be extended by the component DbContext, to simplify advanced testing

Inheritance diagram for TapDbContext:



Public Member Functions

- override int SaveChanges ()

  Throws the exception to simulate concurrent violations in the DB if any is memorized in ToBeThrownBySaveChanges
- override void Dispose ()

  Initializes

Protected Member Functions

- TapDbContext ()
- TapDbContext (DbContextOptions options)
- override void OnConfiguring (DbContextOptionsBuilder options)

  Adds logging of DB interactions

Properties

- static bool TapDbContextIsUsed [get]
- static bool OnConfiguringOk = false [get]

  true if extending classes either do not override OnConfiguring, or call the base version in their overriding

4.8.1 Detailed Description

To be extended by the component DbContext, to simplify advanced testing

4.8.2 Constructor & Destructor Documentation

4.8.2.1 TapDbContext() [1/2]

```
TapDbContext ( )  [protected]
```

4.8.2.2 TapDbContext() [2/2]

```
TapDbContext (
            DbContextOptions options )  [protected]
```

4.8.3 Member Function Documentation

4.8.3.1 Dispose()

```
override void Dispose ( )
```

Initializes

`StudentEntities` assuming that `StudentNames2ExpectedEntities` identifies all expected entities

4.8.3.2 OnConfiguring()

```
override void OnConfiguring (
            DbContextOptionsBuilder options )  [protected]
```

Adds logging of DB interactions

Parameters

options	

4.8.3.3 SaveChanges()

```
override int SaveChanges ( )
```

Throws the exception to simulate concurrent violations in the DB if any is memorized in ToBeThrownBySave↩ Changes

Returns

the same as its parent

4.8.4 Property Documentation

4.8.4.1 OnConfiguringOk

```
bool OnConfiguringOk = false  [static], [get]
```

true if extending classes either do not override OnConfiguring, or call the base version in their overriding

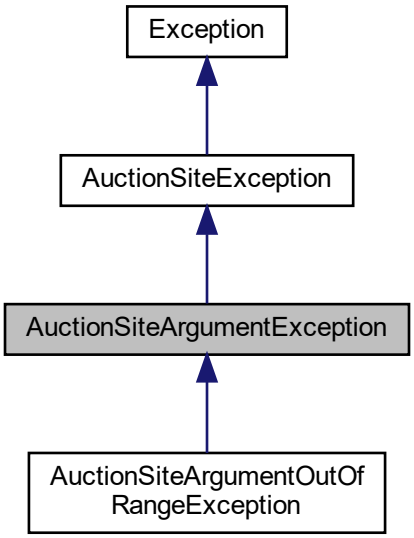
4.8.4.2 TapDbContextIsUsed

```
bool TapDbContextIsUsed  [static], [get]
```

4.9 AuctionSiteArgumentException Class Reference

Thrown when an argument of a method call violates the method preconditions

Inheritance diagram for AuctionSiteArgumentException:



Public Member Functions

- AuctionSiteArgumentException ()
- AuctionSiteArgumentException (string? message)
- AuctionSiteArgumentException (string? message, Exception inner)
- AuctionSiteArgumentException (string? message, string? paramName)
- AuctionSiteArgumentException (string? message, string? paramName, Exception inner)

Protected Member Functions

- AuctionSiteArgumentException (SerializationInfo info, StreamingContext context)

Properties

- virtual ? string ParamName `[get]`

4.9.1 Detailed Description

Thrown when an argument of a method call violates the method preconditions

4.9.2 Constructor & Destructor Documentation

4.9.2.1 AuctionSiteArgumentException() [1/6]

```
AuctionSiteArgumentException ( )
```

4.9.2.2 AuctionSiteArgumentException() [2/6]

```
AuctionSiteArgumentException (
            string?  message )
```

4.9.2.3 AuctionSiteArgumentException() [3/6]

```
AuctionSiteArgumentException (
            string?  message,
            Exception inner )
```

4.9.2.4 AuctionSiteArgumentException() [4/6]

```
AuctionSiteArgumentException (
            string?  message,
            string?  paramName )
```

4.9.2.5 AuctionSiteArgumentException() [5/6]

```
AuctionSiteArgumentException (
            string?  message,
            string?  paramName,
            Exception inner )
```

4.9.2.6 AuctionSiteArgumentException() [6/6]

```
AuctionSiteArgumentException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.9.3 Property Documentation

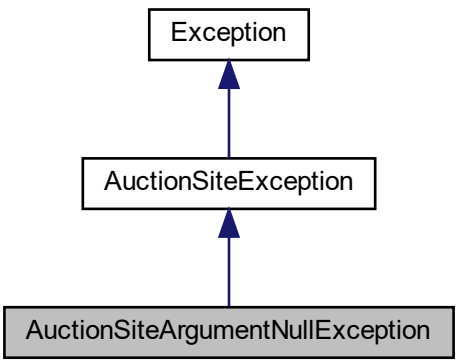
4.9.3.1 ParamName

```
virtual ?  string ParamName  [get]
```

4.10 AuctionSiteArgumentNullException Class Reference

Thrown when an unacceptable null argument is used for a method call

Inheritance diagram for AuctionSiteArgumentNullException:



Public Member Functions

- AuctionSiteArgumentNullException ()
- AuctionSiteArgumentNullException (string? message)
- AuctionSiteArgumentNullException (string? message, Exception inner)

Protected Member Functions

- AuctionSiteArgumentNullException (SerializationInfo info, StreamingContext context)

4.10.1 Detailed Description

Thrown when an unacceptable null argument is used for a method call

4.10.2 Constructor & Destructor Documentation

4.10.2.1 AuctionSiteArgumentNullException() [1/4]

```
AuctionSiteArgumentNullException ( )
```

4.10.2.2 AuctionSiteArgumentNullException() [2/4]

```
AuctionSiteArgumentNullException (
            string?  message )
```

4.10.2.3 AuctionSiteArgumentNullException() [3/4]

```
AuctionSiteArgumentNullException (
            string?  message,
            Exception inner )
```

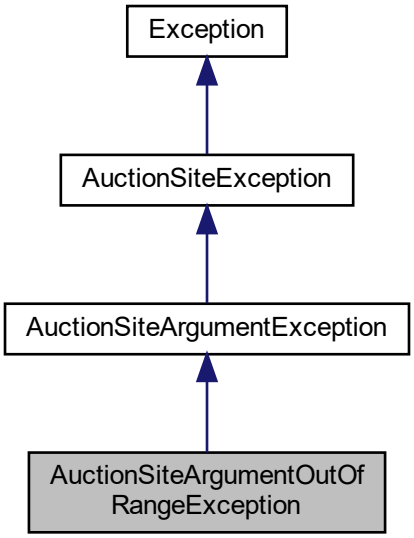
4.10.2.4 AuctionSiteArgumentNullException() [4/4]

```
AuctionSiteArgumentNullException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.11 AuctionSiteArgumentOutOfRangeException Class Reference

Thrown when an argument of a method call does not meet the expected value range for the corresponding parameter

Inheritance diagram for AuctionSiteArgumentOutOfRangeException:



Public Member Functions

- AuctionSiteArgumentOutOfRangeException ()
- AuctionSiteArgumentOutOfRangeException (string? message)
- AuctionSiteArgumentOutOfRangeException (string? message, Exception inner)
- AuctionSiteArgumentOutOfRangeException (string? paramName, string? message)
- AuctionSiteArgumentOutOfRangeException (string? paramName, string? message, Exception inner)
- AuctionSiteArgumentOutOfRangeException (string? paramName, object? value, string? message)

Protected Member Functions

- AuctionSiteArgumentOutOfRangeException (SerializationInfo info, StreamingContext context)

Properties

- virtual ? object ActualValue `[get]`

4.11.1 Detailed Description

Thrown when an argument of a method call does not meet the expected value range for the corresponding parameter

4.11.2 Constructor & Destructor Documentation

4.11.2.1 AuctionSiteArgumentOutOfRangeException() [1/7]

```
AuctionSiteArgumentOutOfRangeException ( )
```

4.11.2.2 AuctionSiteArgumentOutOfRangeException() [2/7]

```
AuctionSiteArgumentOutOfRangeException (
            string?  message )
```

4.11.2.3 AuctionSiteArgumentOutOfRangeException() [3/7]

```
AuctionSiteArgumentOutOfRangeException (
            string?  message,
            Exception inner )
```

4.11.2.4 AuctionSiteArgumentOutOfRangeException() [4/7]

```
AuctionSiteArgumentOutOfRangeException (
            string?  paramName,
            string?  message )
```

4.11.2.5 AuctionSiteArgumentOutOfRangeException() [5/7]

```
AuctionSiteArgumentOutOfRangeException (
            string?  paramName,
            string?  message,
            Exception inner )
```

4.11.2.6 AuctionSiteArgumentOutOfRangeException() [6/7]

```
AuctionSiteArgumentOutOfRangeException (
            string?  paramName,
            object?  value,
            string?  message )
```

4.11.2.7 AuctionSiteArgumentOutOfRangeException() [7/7]

```
AuctionSiteArgumentOutOfRangeException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.11.3 Property Documentation

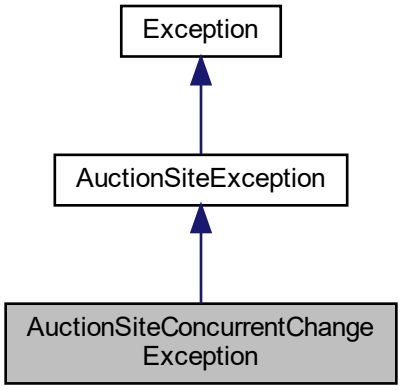
4.11.3.1 ActualValue

```
virtual ?  object ActualValue  [get]
```

4.12 AuctionSiteConcurrentChangeException Class Reference

Thrown to notify an attempt to save an entity that has been modified concurrently in the DB (concurrency management is not required; but, if you want to do it use this exception)

Inheritance diagram for AuctionSiteConcurrentChangeException:



Public Member Functions

- AuctionSiteConcurrentChangeException ()
- AuctionSiteConcurrentChangeException (string? message)
- AuctionSiteConcurrentChangeException (string? message, Exception inner)

Protected Member Functions

- AuctionSiteConcurrentChangeException (SerializationInfo info, StreamingContext context)

4.12.1 Detailed Description

Thrown to notify an attempt to save an entity that has been modified concurrently in the DB (concurrency management is not required; but, if you want to do it use this exception)

4.12.2 Constructor & Destructor Documentation

4.12.2.1 AuctionSiteConcurrentChangeException() [1/4]

```
AuctionSiteConcurrentChangeException ( )
```

4.12.2.2 AuctionSiteConcurrentChangeException() [2/4]

```
AuctionSiteConcurrentChangeException (
            string?  message )
```

4.12.2.3 AuctionSiteConcurrentChangeException() [3/4]

```
AuctionSiteConcurrentChangeException (
            string?  message,
            Exception inner )
```

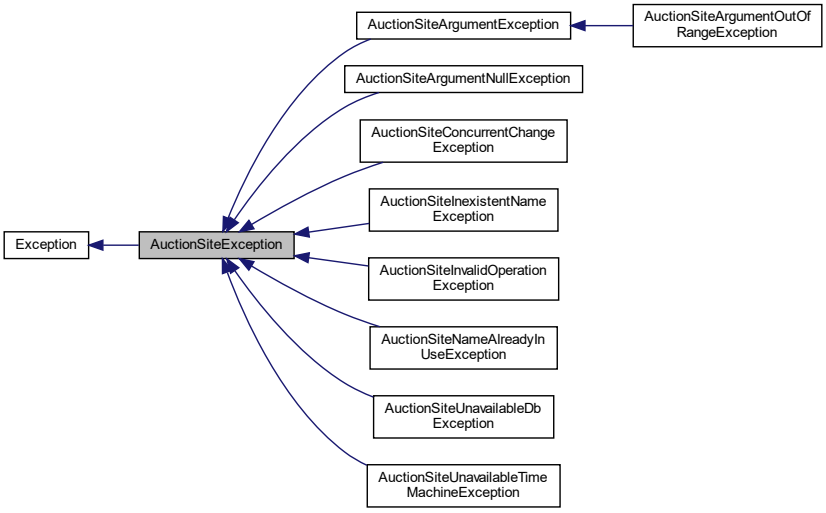
4.12.2.4 AuctionSiteConcurrentChangeException() [4/4]

```
AuctionSiteConcurrentChangeException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.13 AuctionSiteException Class Reference

The interfaces provided by the AuctionSite component.

Inheritance diagram for AuctionSiteException:



Protected Member Functions

- AuctionSiteException ()
- AuctionSiteException (string? message)
- AuctionSiteException (string? message, Exception inner)
- AuctionSiteException (SerializationInfo info, StreamingContext context)

4.13.1 Detailed Description

The interfaces provided by the AuctionSite component.

Only interfaces in this namespace are implemented by the AuctionSite component.

The root for the exception hierarchy of this component. To be used only if no more specific exception is available for the error (you should never need to throw it).

4.13.2 Constructor & Destructor Documentation

4.13.2.1 AuctionSiteException() [1/4]

```
AuctionSiteException ( )  [protected]
```

4.13.2.2 AuctionSiteException() [2/4]

```
AuctionSiteException (
            string?  message )  [protected]
```

4.13.2.3 AuctionSiteException() [3/4]

```
AuctionSiteException (
            string?  message,
            Exception inner )  [protected]
```

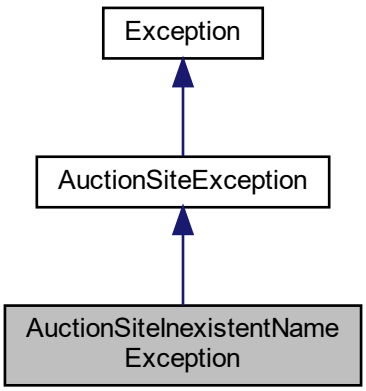
4.13.2.4 AuctionSiteException() [4/4]

```
AuctionSiteException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.14 AuctionSiteInexistentNameException Class Reference

Thrown to notify an attempt to use its name to access an entity that is not available in the database

Inheritance diagram for AuctionSiteInexistentNameException:



Public Member Functions

- AuctionSiteInexistentNameException (string? name)
- AuctionSiteInexistentNameException (string? name, string? message)
- AuctionSiteInexistentNameException (string? name, string? message, Exception inner)
- AuctionSiteInexistentNameException (SerializationInfo info, StreamingContext context)

Properties

- string? Name [get]

Additional Inherited Members

4.14.1 Detailed Description

Thrown to notify an attempt to use its name to access an entity that is not available in the database

4.14.2 Constructor & Destructor Documentation

4.14.2.1 AuctionSiteInexistentNameException() [1/4]

```
AuctionSiteInexistentNameException (
            string?  name )
```

4.14.2.2 AuctionSiteInexistentNameException() [2/4]

```
AuctionSiteInexistentNameException (
            string?  name,
            string?  message )
```

4.14.2.3 AuctionSiteInexistentNameException() [3/4]

```
AuctionSiteInexistentNameException (
            string?  name,
            string?  message,
            Exception inner )
```

4.14.2.4 AuctionSiteInexistentNameException() [4/4]

```
AuctionSiteInexistentNameException (
            SerializationInfo info,
            StreamingContext context )
```

4.14.3 Property Documentation

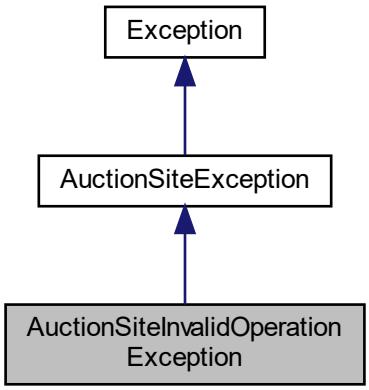
4.14.3.1 Name

```
string?  Name  [get]
```

4.15 AuctionSiteInvalidOperationException Class Reference

Thrown to notify an attempt at invoking an operation on an object in an invalid state (for instance a deleted entity)

Inheritance diagram for AuctionSiteInvalidOperationException:



Public Member Functions

- AuctionSiteInvalidOperationException ()
- AuctionSiteInvalidOperationException (string? message)
- AuctionSiteInvalidOperationException (string? message, Exception inner)

Protected Member Functions

- AuctionSiteInvalidOperationException (SerializationInfo info, StreamingContext context)

4.15.1 Detailed Description

Thrown to notify an attempt at invoking an operation on an object in an invalid state (for instance a deleted entity)

4.15.2 Constructor & Destructor Documentation

4.15.2.1 AuctionSiteInvalidOperationException() [1/4]

```
AuctionSiteInvalidOperationException ( )
```

4.15.2.2 AuctionSiteInvalidOperationException() [2/4]

```
AuctionSiteInvalidOperationException (
            string?  message )
```

4.15.2.3 AuctionSiteInvalidOperationException() [3/4]

```
AuctionSiteInvalidOperationException (
            string?  message,
            Exception inner )
```

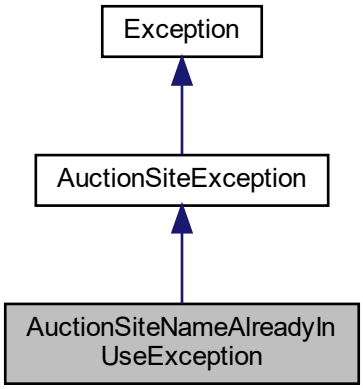
4.15.2.4 AuctionSiteInvalidOperationException() [4/4]

```
AuctionSiteInvalidOperationException (
            SerializationInfo info,
            StreamingContext context )  [protected]
```

4.16 AuctionSiteNameAlreadyInUseException Class Reference

Thrown to notify an attempt to create an entity whose name is already in use. For instance, a site with the same name of another site managed by the same host, or a user with the same name of another user on the same site.

Inheritance diagram for AuctionSiteNameAlreadyInUseException:



Public Member Functions

- AuctionSiteNameAlreadyInUseException (string? name)
- AuctionSiteNameAlreadyInUseException (string? name, string? message)
- AuctionSiteNameAlreadyInUseException (string? name, string? message, Exception inner)
- AuctionSiteNameAlreadyInUseException (SerializationInfo info, StreamingContext context)

Properties

- string? Name `[get]`

Additional Inherited Members

4.16.1 Detailed Description

Thrown to notify an attempt to create an entity whose name is already in use. For instance, a site with the same name of another site managed by the same host, or a user with the same name of another user on the same site.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 AuctionSiteNameAlreadyInUseException() [1/4]

```
AuctionSiteNameAlreadyInUseException (
            string?  name )
```

4.16.2.2 AuctionSiteNameAlreadyInUseException() [2/4]

```
AuctionSiteNameAlreadyInUseException (
            string?  name,
            string?  message )
```

4.16.2.3 AuctionSiteNameAlreadyInUseException() [3/4]

```
AuctionSiteNameAlreadyInUseException (
            string?  name,
            string?  message,
            Exception inner )
```

4.16.2.4 AuctionSiteNameAlreadyInUseException() [4/4]

```
AuctionSiteNameAlreadyInUseException (
            SerializationInfo info,
            StreamingContext context )
```

4.16.3 Property Documentation

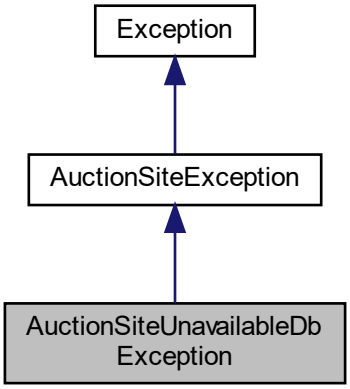
4.16.3.1 Name

```
string?  Name  [get]
```

4.17 AuctionSiteUnavailableDbException Class Reference

Thrown to notify that no connection with the database has been established.

Inheritance diagram for AuctionSiteUnavailableDbException:



Public Member Functions

- AuctionSiteUnavailableDbException ()
- AuctionSiteUnavailableDbException (string? message)
- AuctionSiteUnavailableDbException (string? message, Exception inner)
- AuctionSiteUnavailableDbException (SerializationInfo info, StreamingContext context)

Additional Inherited Members

4.17.1 Detailed Description

Thrown to notify that no connection with the database has been established.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 AuctionSiteUnavailableDbException() [1/4]

```
AuctionSiteUnavailableDbException ( )
```

4.17.2.2 AuctionSiteUnavailableDbException() [2/4]

```
AuctionSiteUnavailableDbException (
            string?  message )
```

4.17.2.3 AuctionSiteUnavailableDbException() [3/4]

```
AuctionSiteUnavailableDbException (
            string?  message,
            Exception inner )
```

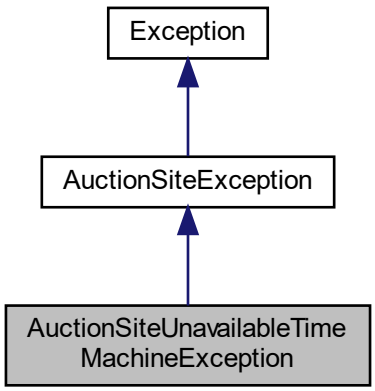
4.17.2.4 AuctionSiteUnavailableDbException() [4/4]

```
AuctionSiteUnavailableDbException (
            SerializationInfo info,
            StreamingContext context )
```

4.18 AuctionSiteUnavailableTimeMachineException Class Reference

Thrown to notify an attempt to create an auction with expiration date in the past.

Inheritance diagram for AuctionSiteUnavailableTimeMachineException:



Public Member Functions

- AuctionSiteUnavailableTimeMachineException ()
- AuctionSiteUnavailableTimeMachineException (string? message)
- AuctionSiteUnavailableTimeMachineException (string? message, Exception inner)
- AuctionSiteUnavailableTimeMachineException (SerializationInfo info, StreamingContext context)

Additional Inherited Members

4.18.1 Detailed Description

Thrown to notify an attempt to create an auction with expiration date in the past.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 AuctionSiteUnavailableTimeMachineException() [1/4]

```
AuctionSiteUnavailableTimeMachineException ( )
```

4.18.2.2 AuctionSiteUnavailableTimeMachineException() [2/4]

```
AuctionSiteUnavailableTimeMachineException (
            string?  message )
```

4.18.2.3 AuctionSiteUnavailableTimeMachineException() [3/4]

```
AuctionSiteUnavailableTimeMachineException (
            string?  message,
            Exception inner )
```

4.18.2.4 AuctionSiteUnavailableTimeMachineException() [4/4]

```
AuctionSiteUnavailableTimeMachineException (
            SerializationInfo info,
            StreamingContext context )
```

Index