# Audit library

This utility is used to track changes on entities or any kind of object.

# How to use this library

Consider following Sample Product Class and we want to track the changes to object product.



In order to track the changes, instance of type IAuditTrail needs to be created. Any changes to object that happens inside the scope of instance will be tracked.

AuditTrailFactory is used to create audit trail instance.



AuditTrailFactory’s Create Method needs following parameters to create an instance.

* Event Type: Type of event we are trying to track. E.g ProductUpdate
* Getter of Target Object we are trying to track.
* Instance of IAuditDataAdapter which will be used to save the audit event.
* Instance of IAuditEventUser which stores information about the user making these changes.

Once audit trail instance is created it will start tracking the changes to be object in IAuditEvent. In this example I am changing the price of product from 100 to 20.

We can also use Comment method of IAuditTrail to add comments to event.

Invoking Save method will save the AuditEvent using IAuditDataAdapter instance. In this example I am saving the event in Json file.

See below the content of the files for this Audit Event.



Custom data adapter can be created for persisting the event. E.g. Data Adapter for storing data to SQL server database. All data adapters should implement IAuditDataAdapter interface. This interface has following methods that needs to be implemented by DataAdapters.



There is also an Abstract Class AuditDataAdapter that implements Serialize method which serializes the object in to JToken. It leaves implementation of InsertEvent abstract for child class.

Please look at sample test case in code to learn more about using this utility.

# Code

## Solution

Audit Solution has two projects

* Audit class library with AuditTrail Implementation
* AuditTest unit test project with sample unit test to save audit trail.

## Interfaces

### IAuditTrail

This interface has following signature. Instance of type IAuditTrail can be created using AuditTrailFactory’s Create method. Implementation of this is described in detail below.



### IAuditDataAdapter

This adapter will be used to save the AuditEvent.



### IAuditEvent

This interface is used to define an Event that we want to save in our Audit Trail.



### IAuditTargetEntity

This is used to define the entity that needs to be tracked.



### IAuditEventUser

This is used to define the user who created the event



## Implementation

### AuditTrail

This class implements IAuditTrail Interface.

It has a constructor to create instance from options passed to it.



AuditOptions provides following instances

* Data Adapter that will be used to save the audit trail.
* Getter of target object that needs to be tracked
* New instance of AuditEvent.
* TargetEntity that will store the state of objects before and after change.

To track object changes AuditTrail saves original serialized object in TargetEntity.Old property during the creation of instance. At time of save it saves changed serialized object in TargetEntity.New property.



Since IAuditTrail is inheriting IDisposible AuditTrail also provides implementation of Dispose method. Here it ends the event and serializes the changed target object and saves in TargetEntity.New property



### AuditTrailFactory

This is a factory that is used to create the instance of AuditTrail . It has a create method that takes following parameters

* Event Type: Type of event we are trying to track. E.g ProductUpdate
* Getter of Target Object we are trying to track.
* Instance of IAuditDataAdapter which will be used to save the audit event.
* Instance of IAuditEventUser which stores information about the user making these changes



### FileDataAdapter

This adapter implements IAuditDataAdapter Interface. This provides implementation of saving the audit trail into json file.



# Future Implementation

If given more time I would like to create SQL server Database adapter that will save audit trail to database. Also there is a scope of enabling more options like saving the audit trail when scope ends without implicitly calling save method.