Mocking EF DbContext

Erkan_Demirel, 13 Oct 2016

Adapter implemantation to mock DbContext easily

Introduction

If you want to mock `DbContext`, these adapters will help you to test your code easily.

Background

Adapter

Convert the interface of a class into another interface clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces.

Using the Code

I wanted to mock `DbContext`. So I implemented an interface for my context class.

Hide Copy Code

```
public class BrokerContext : DbContext, IBrokerContext
{
.
.
.
}
```

This was fine, but then I couldn't mock `context.Database.ExecuteSqlCommand()`.

So, I implemented an adapter for `Database` property.

But later, I had another problem:

Hide Copy Code

```
using(var myTransaction=context.Database.BeginTransaction())
{
 myTransaction.Commit();
}
```

This time, I couldn't mock `context.Database.BeginTransaction()`.

So I implemented adapter for `DbContextTransaction`.

Here is the last code:

Main Level

Hide Shrink Copy Code

```
 public interface IBrokerContext :IDisposable
 {
     //This is our Interface. We can mock Database with this Interface.
     IMyDatabase Database { get; }
     .
     .
     .
     DbContextConfiguration Configuration { get; }
     int SaveChanges();
     DbSet<TEntity> Set<TEntity>() where TEntity : class;
     DbSet Set(Type entityType);
     Task<int> SaveChangesAsync();
     DbEntityEntry<TEntity> Entry<TEntity>(TEntity entity) where TEntity : class;
     DbEntityEntry Entry(object entity);
 }

public class BrokerContext : DbContext, IBrokerContext
 {
     public BrokerContext()
         : base("Name=BrokerContext")
     {
         //We initialize our Database(IMyDatabase).
         Database = new MyDatabase(base.Database);
         Database.SetInitializer(new CreateDatabaseIfNotExists<BrokerContext>());
     }
     .
     .
     .
     //We are hiding base Class Database.
     public new IMyDatabase Database { get; }
 }
```

Second Level

Hide Shrink Copy Code

```
public interface IMyDatabase
 {
     int ExecuteSqlCommand(string sql, params object[] parameters);
     void SetInitializer<TContext>
     (IDatabaseInitializer<TContext> strategy) where TContext : DbContext;
     //This is our Interface. We can mock BeginTransaction.
     IMyDbContextTransaction BeginTransaction();
 }

 public class MyDatabase : IMyDatabase
 {
     private readonly Database _database;

     public MyDatabase(database)
     {
         _database = database;
     }

     public int ExecuteSqlCommand(string sql, params object[] parameters)
     {
        return _database.ExecuteSqlCommand(sql,parameters);
     }

     public void SetInitializer<TContext>
     (IDatabaseInitializer<TContext> strategy) where TContext : DbContext
     {
         Database.SetInitializer(strategy);
     }

     public IMyDbContextTransaction BeginTransaction()
     {
        //Initialize our DbContextTransaction(MyDbContextTransaction)
        var myDbContextTransaction = new
MyDbContextTransaction(_database.BeginTransaction());
         return myDbContextTransaction;
     }
 }
```

Third Level

Hide Shrink Copy Code

```
 public interface IMyDbContextTransaction:IDisposable
    {
        void Commit();
        void Rollback();
    }

 public class MyDbContextTransaction : IMyDbContextTransaction
{
    private readonly DbContextTransaction _dbContextTransaction;

    public MyDbContextTransaction(DbContextTransaction dbContextTransaction)
    {
        _dbContextTransaction = dbContextTransaction;
    }

    public void Commit()
```

```
    {
        _dbContextTransaction.Commit();
    }

    public void Rollback()
    {
        _dbContextTransaction.Rollback();
    }

    public void Dispose()
    {
        _dbContextTransaction.Dispose();
    }
}
```

I added only methods which I need on adapters. Also, I just implemented adapters if I really needed them, not for all `DbContext`.

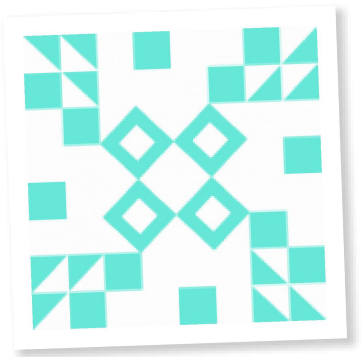
Edit

If you don't want to control all `DbContext.Database`, here is a simpler solution.

License

Share

About the Author



Erkan_Demirel
Software Developer (Senior)
Turkey

No Biography provided

You may also be interested in...

- Mocking EF DbContext and DbContextTransaction with Proxy
- Window Tabs (WndTabs) Add-In for DevStudio
- Testing with mock on Entity Framework 6
- To Heap or not to Heap; That's the Large Object Question?
- SAPrefs - Netscape-like Preferences Dialog
- Introduction to D3DImage

Comments and Discussions

You must Sign In to use this message board.

Search Comments

First Prev Next

Another suggestion
Jan Hansen 2-Mar-16 4:31

Re: Another suggestion
MaDeRkAn 2-Mar-16 23:01

Refresh 1

General News Suggestion Question Bug Answer Joke Praise Rant Admin

Use Ctrl+Left/Right to switch messages, Ctrl+Up/Down to switch threads, Ctrl+Shift+Left/Right to switch pages.

Layout: fixed | fluid