Log in / create account

Fluent NHibernate wiki

Fluent configuration

From Fluent NHibernate

Contents

Fluent NHibernate provides an API for completely configuring NHibernate for use with your application, all within code. The API is broken down into five main methods, three of which are required.

```
Fluently.Configure()
  .Database(/* your database settings */)
  .Mappings(/* your mappings */)
  .ExposeConfiguration(/* alter Configuration */) // optional
  .BuildSessionFactory();
```

You can combine these methods in various ways to setup your application.

1. Fluently.Configure starts the configuration process
2. Database is where you specify your database configuration using the database configuration api
3. Mappings is where you supply which mappings you're using
4. ExposeConfiguration is optional, but allows you to alter the raw Configuration object
5. BuildSessionFactory is the final call, and it creates the NHibernate SessionFactory instance from your configuration.

The API is fairly small, and relatively discoverable, so it's easy to configure it in a way that will work with your application; however, here are some common usages:

Exclusively fluent

If you're in the situation where your application is exclusively using fluent mappings, then this is the configuration for you.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
    m.FluentMappings
      .AddFromAssemblyOf<YourEntity>())
  .BuildSessionFactory();
```

This setup uses the SQLite database configuration, but you can substitute that with your own; it then adds any fluent mappings from the assembly that contains YourEntity.

Automappings

If you're using only auto mappings, then this config is for you.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
    m.AutoMappings.Add(
      // your automapping setup here
      AutoMap.AssemblyOf<YourEntity>(type => type.Namspace.EndsWith("Entities"))))
  .BuildSessionFactory();
```

Replace the code inside AutoMappings.Add with your auto mapping configuration.

Mixed fluent mappings and auto mappings

If you're using a combination of standard fluent mappings and auto mappings, then this example should show you how to get started.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
  {
    m.FluentMappings
      .AddFromAssemblyOf<YourEntity>();

    m.AutoMappings.Add(
      // your automapping setup here
      AutoMap.AssemblyOf<YourEntity>(type => type.Namspace.EndsWith("Entities")));
  })
  .BuildSessionFactory();
```

You can see that this is a combination of the two previous examples, the Mappings method can accept multiple kinds of mappings.

HBM mappings

You've not yet got around to using Fluent NHibernate fully, but you are configuring your database with it; this configuration will let you configure your database and add your traditional hbm mappings.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
    m.HbmMappings
      .AddFromAssemblyOf<YourEntity>())
  .BuildSessionFactory();
```

The HbmMappings property allows you to add HBM XML mappings in a few different ways, this example adds everything from an assembly which defines YourEntity; however, you can add from an assembly instance, or just add single types.

Mixed HBM and fluent mappings

You're migrating your entities to Fluent NHibernate but haven't quite got them all across yet - this is for you.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
  {
    m.HbmMappings
      .AddFromAssemblyOf<YourEntity>();

    m.FluentMappings
      .AddFromAssemblyOf<YourEntity>();
  })
  .BuildSessionFactory();
```

The whole shebang: fluent, auto, and hbm mappings

You're a crazy fool and map a bit of everything, then this is how you'd configure it.

```
var sessionFactory = Fluently.Configure()
  .Database(SQLiteConfiguration.Standard.InMemory)
  .Mappings(m =>
  {
    m.HbmMappings
      .AddFromAssemblyOf<YourEntity>();

    m.FluentMappings
      .AddFromAssemblyOf<YourEntity>();

    m.AutoMappings.Add(
      // your automapping setup here
      AutoMap.AssemblyOf<YourEntity>(type => type.Namspace.EndsWith("Entities")));
  })
  .BuildSessionFactory();
```

Exporting mappings

In the Mappings call, you can do the following:

```
.Mappings(m =>
{
  m.FluentMappings
    .AddFromAssemblyOf<YourEntity>()
    .ExportTo(@"C:\your\export\path");

  m.AutoMappings
    .Add(/* ... */)
    .ExportTo(@"C:\your\export\path");
})
```

That will export all of your fluent and automapped mappings in hbm.xml format to whatever location you specify.

Retrieved from "http://wiki.fluentnhibernate.org/Fluent_configuration"

Page Discussion View source History

Search