



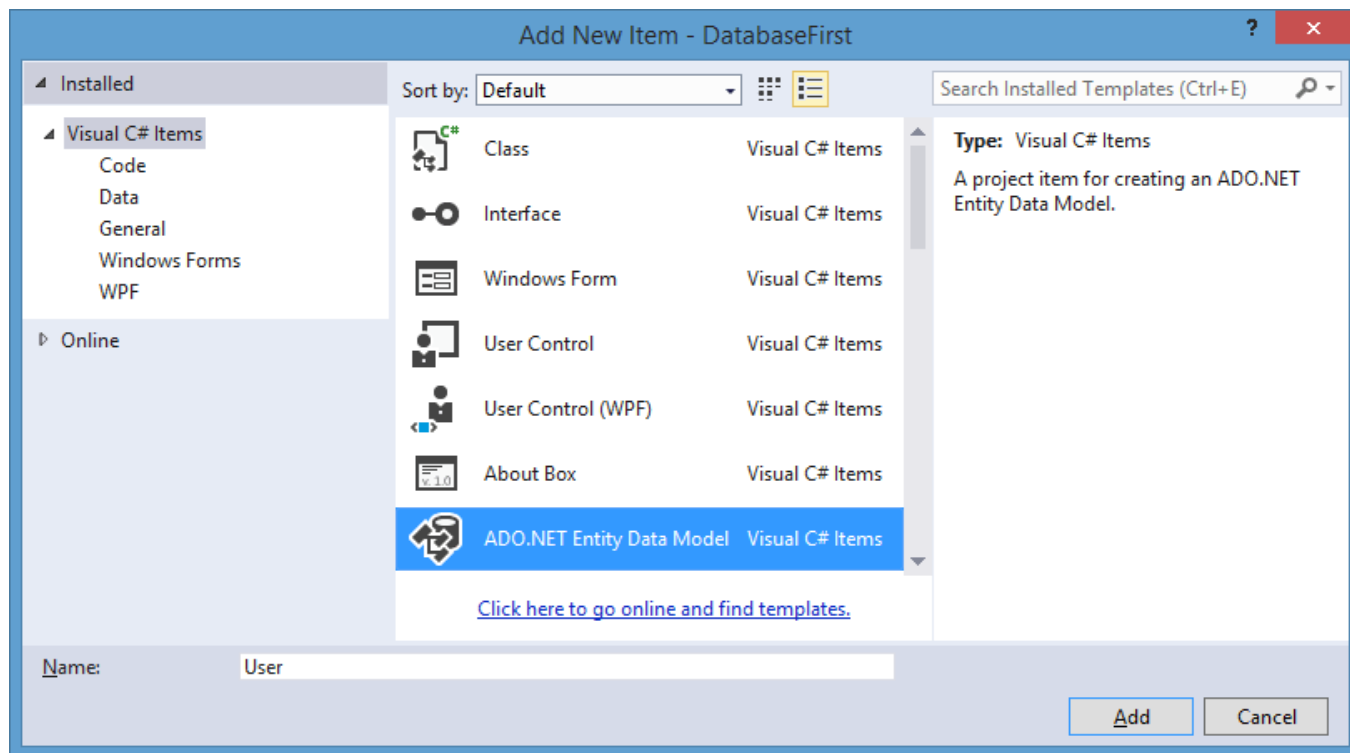
Автоматизация Code First

Последнее обновление: 31.10.2015

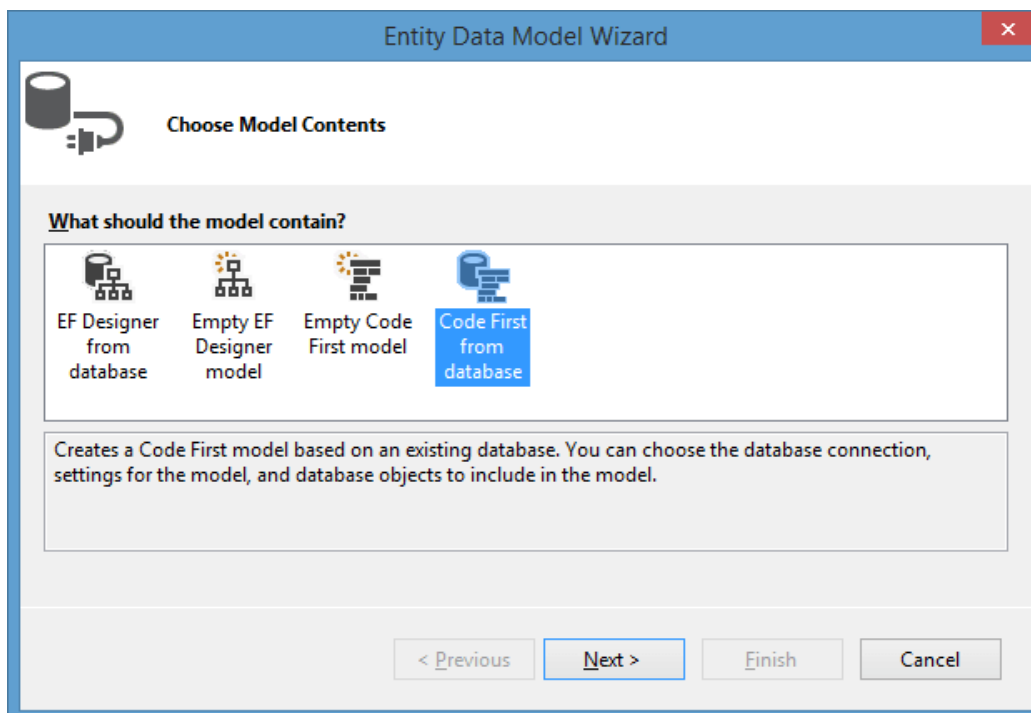


Вручную создавать классы по уже готовой бд со всеми полями и связями между собой довольно утомительно, особенно если таблиц в БД очень много. В обновленных версиях Visual Studio 2013 с пакетами обновления SP3 мы можем автоматизировать этот процесс.

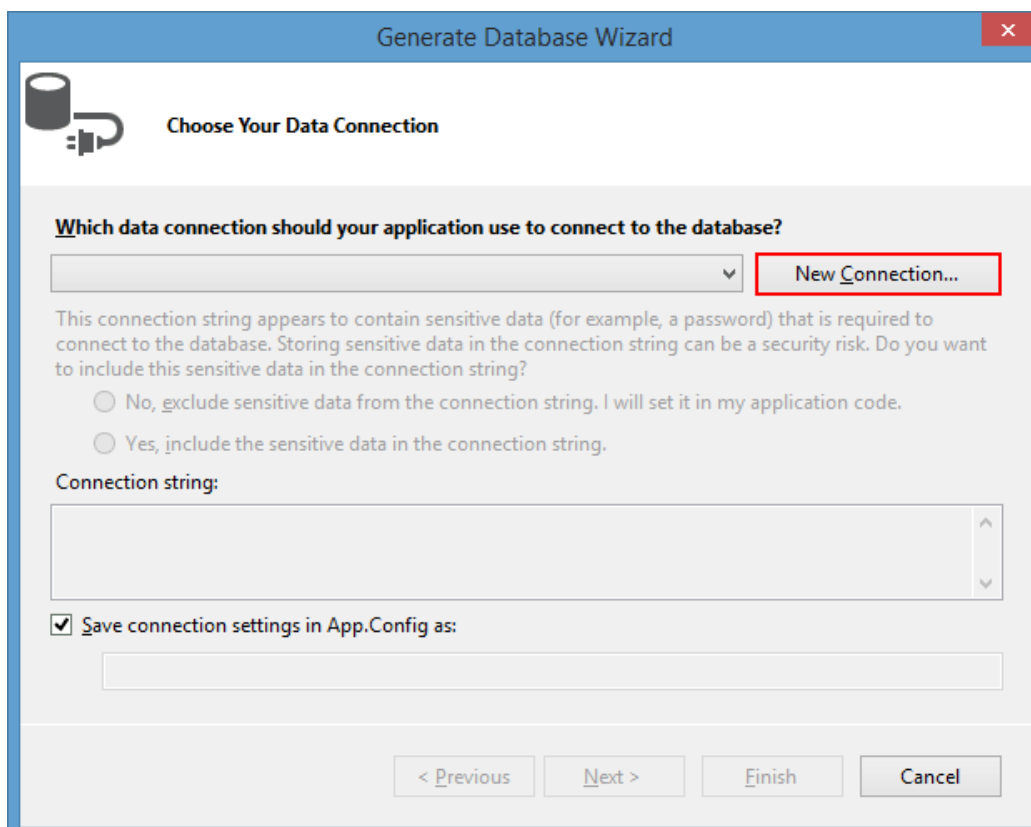
Для этого добавим в проект новый элемент **ADO.NET Entity Data Model**:



Нажмем ОК и нам откроется мастер создания модели. Здесь нам надо выбрать пункт **Code First from database**:



Далее на следующем шаге настройки модели надо будет установить подключение к имеющейся базе данных.



Нажмем на кнопку **New Connection** и в следующем окне настроек подключения выберем сервер и базу данных, с которой мы хотим работать:

Connection Properties

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
(localdb)\v11.0 Refresh

Log on to the server

☒ Use Windows Authentication
☐ Use SQL Server Authentication

User name:
Password:
☐ Save my password

Advanced...

Connect to a database

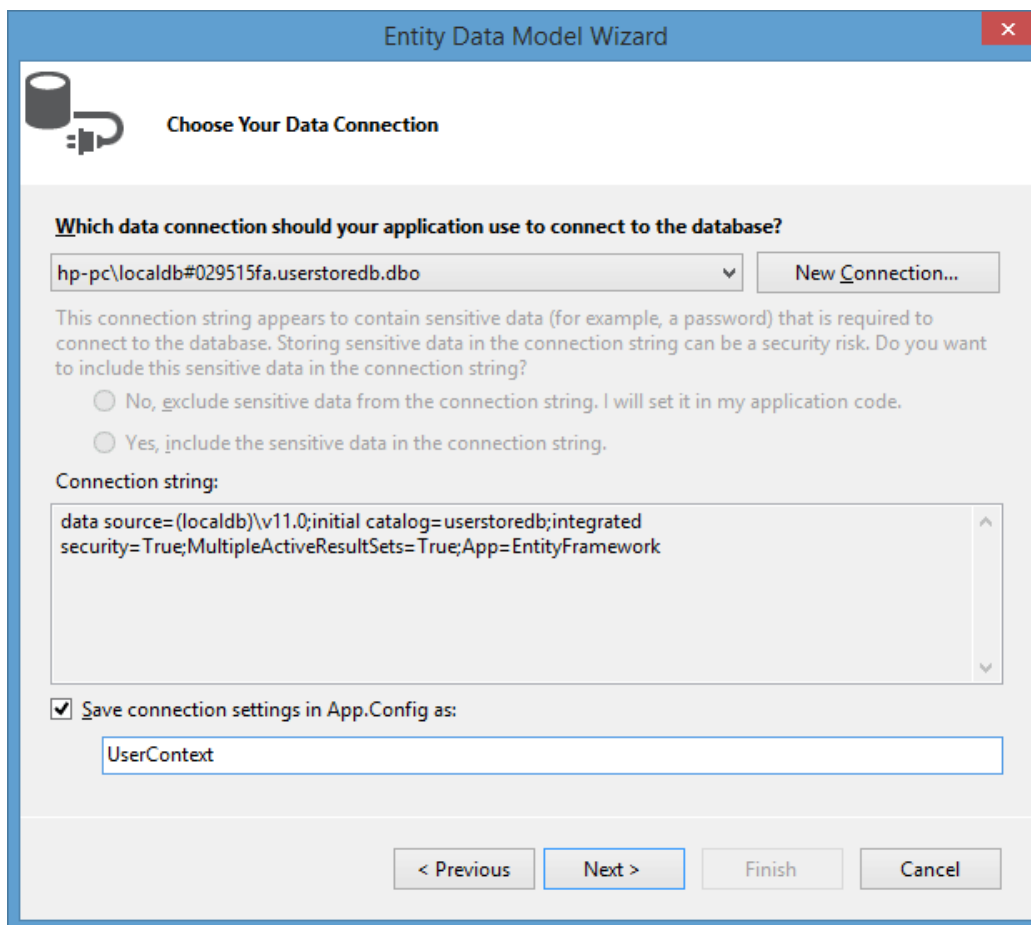
☒ Select or enter a database name:
userstoredb

☐ Attach a database file:
 Browse...

Logical name:

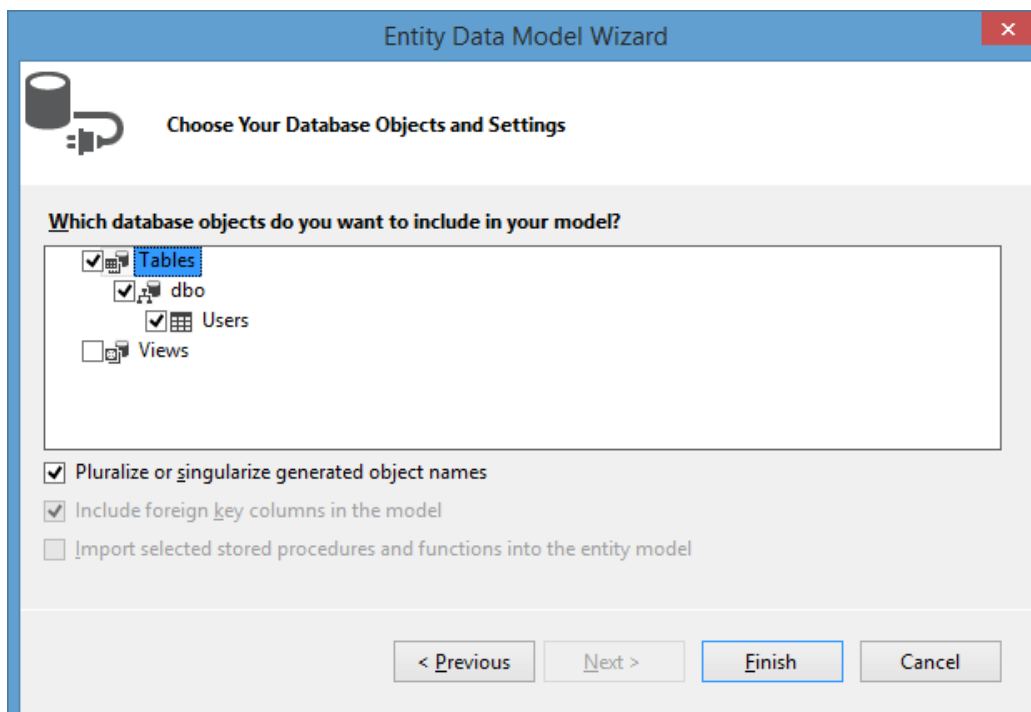
Test Connection OK Cancel

После этого в окне мастера настройки модели появится выбранное подключение. И также здесь мы можем установить название подключения, которое будет использоваться в файле конфигурации App.config. Изменим его, например, на UserContext:



The screenshot shows the 'Entity Data Model Wizard' window, specifically the 'Choose Your Data Connection' step. The title bar reads 'Entity Data Model Wizard'. The main heading is 'Choose Your Data Connection'. Below this, a question asks: 'Which data connection should your application use to connect to the database?'. A dropdown menu shows 'hp-pc\localdb#029515fa.userstoredb.dbo', and a 'New Connection...' button is to its right. A warning message states: 'This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?'. Two radio buttons are provided: 'No, exclude sensitive data from the connection string. I will set it in my application code.' (selected) and 'Yes, include the sensitive data in the connection string.'. Below this, the 'Connection string:' is displayed in a text box: 'data source=(localdb)\v11.0;initial catalog=userstoredb;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework'. A checkbox 'Save connection settings in App.Config as:' is checked, with a text box below it containing 'UserContext'. At the bottom are buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

Нажмем Next, и на следующем шаге нам будет предложено выбрать те таблицы из бд, по которым нам надо создать модели:



The screenshot shows the 'Entity Data Model Wizard' window, specifically the 'Choose Your Database Objects and Settings' step. The title bar reads 'Entity Data Model Wizard'. The main heading is 'Choose Your Database Objects and Settings'. Below this, a question asks: 'Which database objects do you want to include in your model?'. A list of objects is shown with checkboxes: 'Tables' (checked), 'dbo' (checked), 'Users' (checked), and 'Views' (unchecked). Below the list, three checkboxes are present: 'Pluralize or singularize generated object names' (checked), 'Include foreign key columns in the model' (checked), and 'Import selected stored procedures and functions into the entity model' (unchecked). At the bottom are buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

И затем нажмем Finish. После этого будут сгенерированы классы моделей. Например, в моем случае по единственной таблице в бд будет сгенерирован следующий класс:

```
namespace AutoCodeSecond
{
    using System;
    using System.Collections.Generic;
    using System.ComponentModel.DataAnnotations;
```

```
using System.ComponentModel.DataAnnotations.Schema;
using System.Data.Entity.Spatial;

public partial class User
{
    public int Id { get; set; }

    [Required]
    [StringLength(50)]
    public string Name { get; set; }

    public int Age { get; set; }
}
```

И также надо отметить, что в файле *App.config* появилось определение подключения:

```
<connectionStrings>
  <add name="UserContext" connectionString="data source=(localdb)\v11.0;initial catalog=userstoredb;
    integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

Для полноценной работы нам осталось добавить класс контекста данных:

```
using System;
using System.Collections.Generic;
using System.Data.Entity;

namespace AutoCodeSecond
{
    class UserContext : DbContext
    {
        public UserContext():base("UserContext")
        { }
        public DbSet<User> Users { get; set; }
    }
}
```

И теперь мы можем взаимодействовать с базой данных:

```
using(UserContext db = new UserContext())
{
    foreach (User u in db.Users)
        Console.WriteLine("{0}.{1} - {2}", u.Id, u.Name, u.Age);
}
```

[Назад](#) [Содержание](#) [Вперед](#)

