如何实现安全提供程序

1. 描述

活字格提供ISecurityProvider接口。通过实现该接口，用户可以与其他系统进行集成。

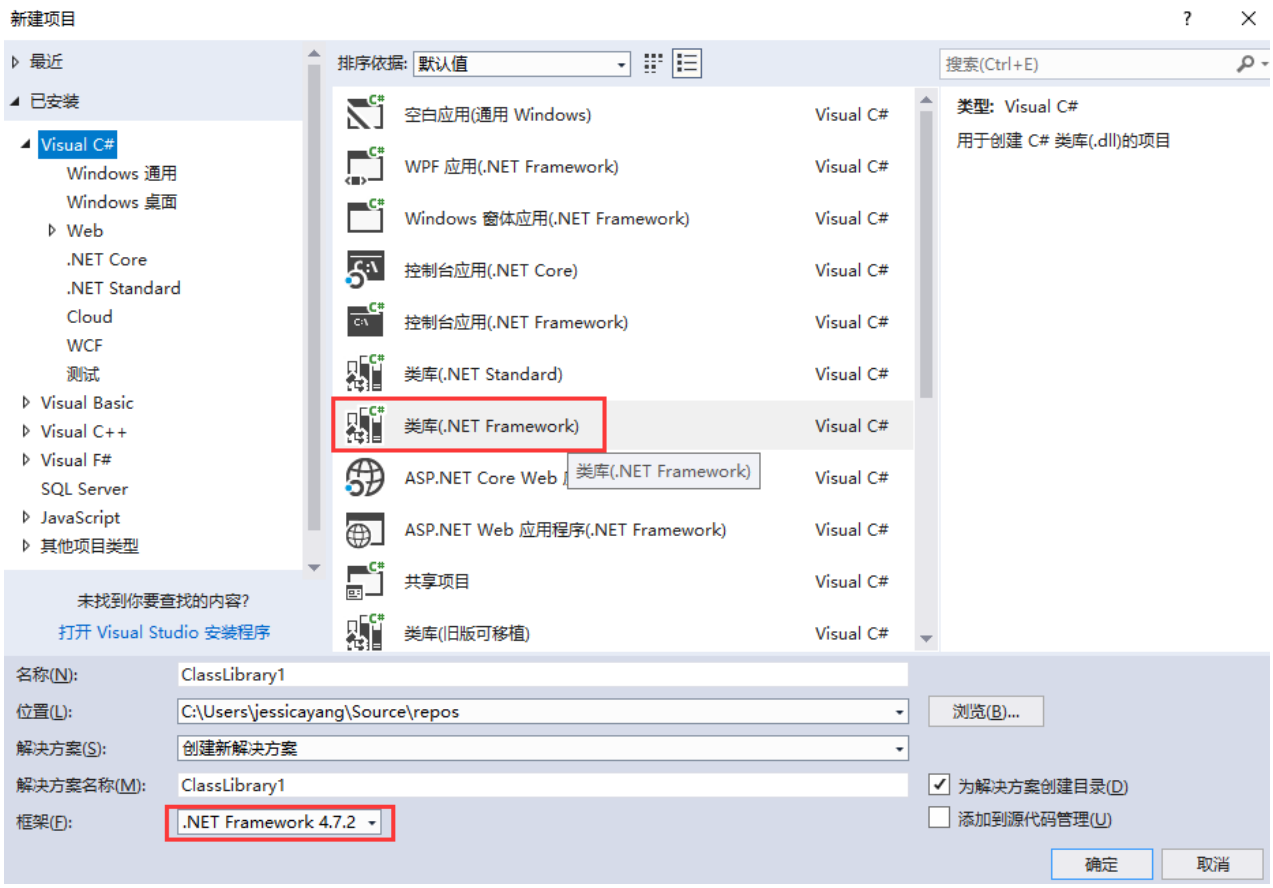
本节介绍如何实现安全提供程序。

2. 实现安全提供程序

操作步骤

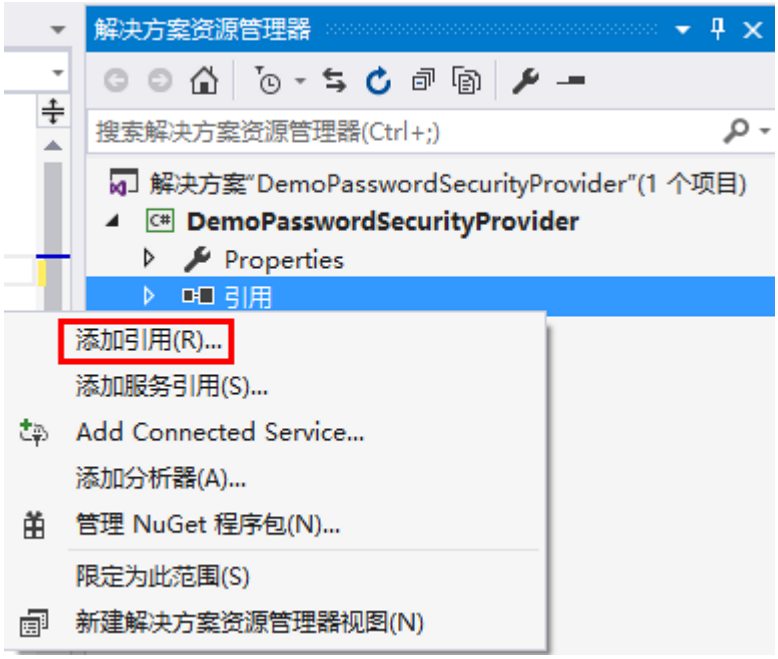
1. 在Visual Studio中创建类库项目。设置项目的Framework框架为.NET Framework 4.7.2。如图1所示。

图1　创建类库项目



2. 在解决方案资源管理器中的“引用”上右击，选择“添加引用”。

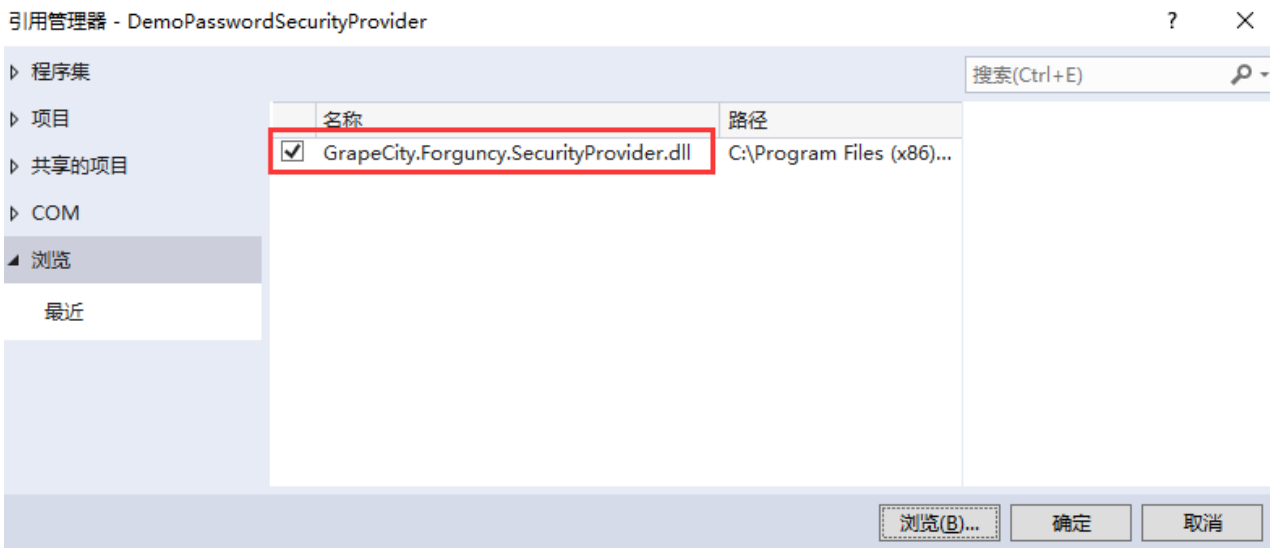
图2　添加引用



3

单击右下角的“浏览”，在活字格的安装目录下找到“GrapeCity.Forguncy.SecurityProvider.dll”文件并将其添加为该工程的引用。

- 如果安装活字格服务端时，安装目录为默认目录，则此文件的路径为“C:\Program Files\ForguncyServer\Website\bin”。
- 如果安装活字格服务端时，安装目录为自定义路径，则此文件的路径为“自定义路径\ForguncyServer\Website\bin”。



4

选中“GrapeCity.Forguncy.SecurityProvider.dll”文件，右击选择“属性”，将“复制到本地”改成“False”。

图3　更改属性

5

在Class1.cs文件中添加代码，示例代码如下：

```
using GrapeCity.Forguncy.SecurityProvider;
using System;
using System.Collections.Generic;
using System.Linq;

namespace DemoPasswordSecurityProvider
{
    public class DemoPasswordSecurityProvider : ISecurityProvider
    {
        List<User> testUsers = new List<User>();
        public DemoPasswordSecurityProvider()
        {
            AddUser("User1", "TestUser1", "User1@User1.com",
"User1Property");
            AddUser("User2", "TestUser2", "User2@User2.com",
"User2Property");
            AddUser("User3", "TestUser3", "User3@User3.com",
"User3Property");
        }

        void AddUser(string userId, string userName, string email,
string customProperty)
        {
            testUsers.Add(new User()
            {
                UserId = userId,
                Email = email,
                FullName = userName
            });

            testUsers.Last().Properties.Add("customProperty",
customProperty);
        }

        public string Name
        {
            get
            {
                return "DemoPasswordSecurityProvider";
            }
```

```
        }

        public AuthenticationType AuthenticationType
        {
            get
            {
                return AuthenticationType.UserNameAndPassword;
            }
        }

        public UserInformationStorageMode UserInformationStorageMode
        {
            get
            {
                return UserInformationStorageMode.InForguncyDatabase;
            }
        }

        public UserInformations UserInformations => throw new
NotImplementedException();

        public bool AllowLogout
        {
            get
            {
                return true;
            }
        }

        private User GetUserInformation(string userId)
        {
            return testUsers.FirstOrDefault(u => u.UserId == userId);
        }

        public User VerifyUser(Dictionary<string, string> properties)
        {
            var userName = properties["userName"];
            var password = properties["password"];

            var user = GetUserInformation(userName);

            if (user != null && password == "123456")
            {
                user.Properties.Add("FGC_Role", "MyRole");
                return user;
            }

            return null;
        }
    }
```

```
}
```

其中，InForguncyDatabase为活字格存储用户信息的一种方式，另一种方式为InMemoryCache。

- InForguncyDatabase
  在这种模式下，当用户首次登录活字格网站时，活字格会将用户信息添加到活字格的用户服务数据库中。管理员需要将角色和组织分配给活字格中的用户。
  如果用户已被添加到活字格，则在原始应用程序中删除用户或用户属性更改，活字格将不会自动同步。

- InMemoryCache
  在这种模式下，活字格站点将在启动时获得所有用户信息，用户信息包括用户名、用户属性、角色和组织。然后将用户信息缓存到站点的内存中。活字格将使用这些信息来设置权限和工作流等。

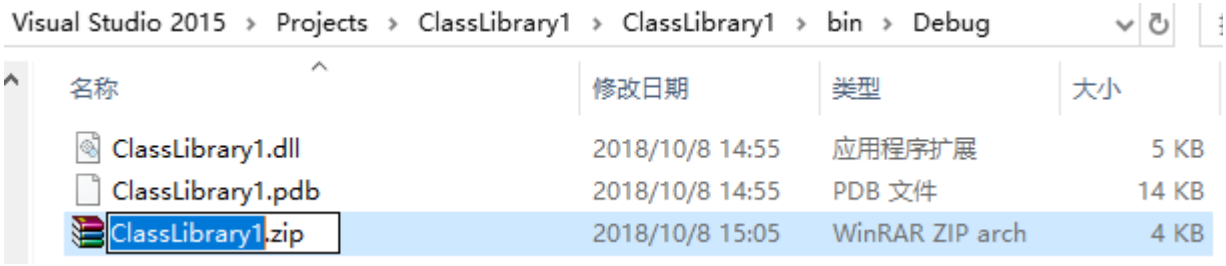
6

代码添加完成后，在名称上右击，选择“生成”或者“重新生成”，进行编译。

7

编译完成后，在文件资源管理器中打开该文件夹，在“bin/Debug”或“bin/Release”文件夹下，将生成的所有文件打包为.zip文件。

图4　打包文件



8

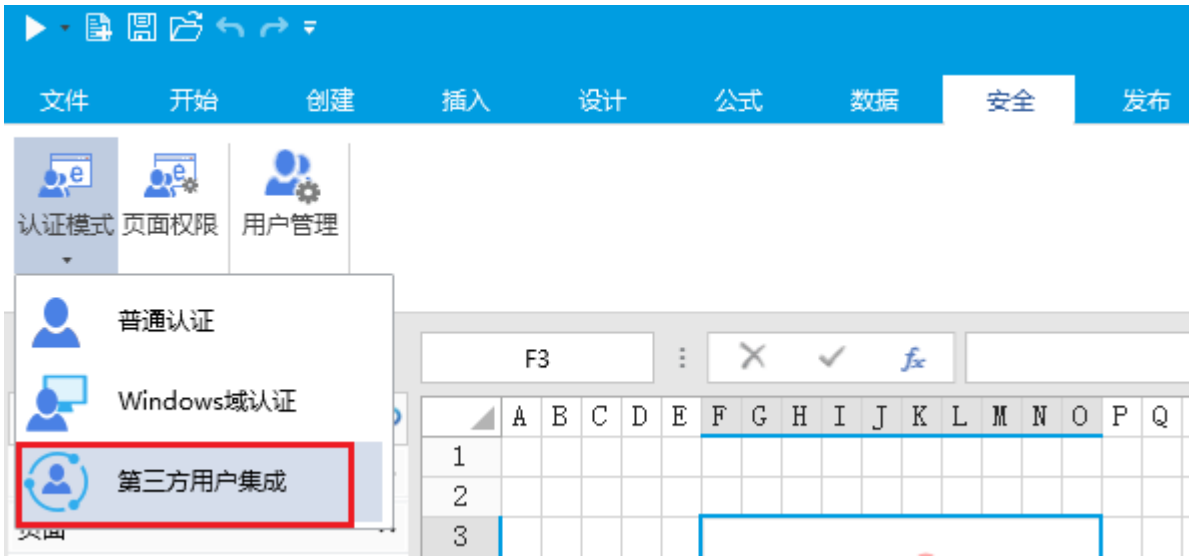
在用户账户管理界面的“第三方”区域，单击“上传”，选择安全提供程序.zip打包文件即可。

图5　上传安全提供程序包

9

在活字格设计器中，选择认证模式为“第三方用户集成”，发布后即可实现自定义的用户安全程序。

图6　选择认证模式


文件
开始
创建
插入
设计
公式
数据
安全
发布
认证模式
页面权限
用户管理
普通认证
Windows域认证
第三方用户集成
F3
A B C D E F G H I J K L M N O P Q
1
2
3

结束

回到顶部