عنوان: **آشنایی با آزمایش واحد (unit testing) در دات نت، قسمت 6**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۱ ۰۹:۱۵:۲۸
آدرس: www.dotnettips.info
برچسب‌ها: Unit testing

ادامه آشنایی با NUnit

فرض کنید یک RSS reader نوشته‌اید که فیدهای فارسی و انگلیسی را دریافت می‌کند. به صورت پیش فرض هم مشخص نیست که کدام فید اطلاعات فارسی را ارائه خواهد داد و کدامیک انگلیسی. تشخیص محتوای فارسی و از راست به چپ نشان دادن خودکار مطالب آن‌ها به عهده‌ی برنامه نویس است. بهترین روش برای تشخیص این نوع الگوها، استفاده از regular expressions است.
برای مثال الگوی تشخیص اینکه آیا متن ما حاوی حروف انگلیسی است یا خیر به صورت زیر است:

```
[a-zA-Z]
```

که بیان بصری آن به این شکل است.
در مورد تشخیص وجود حروف فارسی در یک عبارت، یکی از دو الگوی زیر بکار می‌رود:

```
[\u0600-\u06FF]
[ا-یءئ]
```

در مورد اینکه بازه یونیکد فارسی استاندارد از کجا شروع می‌شود می‌توان به مقاله‌ی آقای حاج‌لو مراجعه نمود (به صورت خلاصه، بازه مصوب عربی یونیکد، همان بازه یونیکد فارسی نیز می‌باشد . یا به بیان بهتر، بازه‌ی فارسی، جزئی از بازه‌ای است که عربی نام گرفته است). البته بازه‌ی مصوب دیگری هم در مورد ایران باستان وجود دارد به نام old Persian که مورد استفاده‌ی روزمره‌ای ندارد!

کلاس زیر را در مورد استفاده از این الگوها تهیه کرده‌ایم:

```
using System.Text.RegularExpressions;

namespace sample
{
   public static class CDetectFarsi
   {
       public static bool ContainsFarsiData(this string txt)
       {
           return !string.IsNullOrEmpty(txt) &&
                   Regex.IsMatch(txt, "[ا-یءئ]");
       }

       public static bool ContainsFarsi(this string txt)
       {
           return !string.IsNullOrEmpty(txt) &&
                   Regex.IsMatch(txt, @"[\u0600-\u06FF]");
       }
   }
}
```

همانطور که ملاحظه می‌کنید در اینجا از extension methods سی شارپ 3 جهت توسعه کلاس پایه string استفاده شد.
اکنون می‌خواهیم بررسی کنیم آیا این الگوها مقصود ما را برآورده می‌سازند یا خیر.

```
using NUnit.Framework;
using sample;

namespace TestLibrary
{
   [TestFixture]
   public class TestFarsiClass
   {
       /*****************************************************************************/
       [Test]
       public void TestContainsFarsi1()
       {
           Assert.IsTrue("وحید".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi2()
       {
           Assert.IsTrue("گردان".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi3()
       {
           Assert.IsTrue("سپیدTest".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi4()
       {
           Assert.IsTrue("123456بررسی".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi5()
       {
           Assert.IsFalse("Book".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi6()
       {
           Assert.IsTrue("\۳۸۷".ContainsFarsi());
       }

       [Test]
       public void TestContainsFarsi7()
       {
           Assert.IsFalse("Здравствуйте!".ContainsFarsi()); //Russian hello!
       }


       /*****************************************************************************/
       [Test]
       public void TestContainsFarsiData1()
       {
           Assert.IsTrue("وحید".ContainsFarsiData());
       }

       [Test]
       public void TestContainsFarsiData2()
       {
           Assert.IsTrue("گردان".ContainsFarsiData());
       }

       [Test]
       public void TestContainsFarsiData3()
       {
           Assert.IsTrue("سپیدTest".ContainsFarsiData());
       }

       [Test]
       public void TestContainsFarsiData4()
       {
           Assert.IsTrue("123456بررسی".ContainsFarsiData());
       }

       [Test]
       public void TestContainsFarsiData5()
       {
           Assert.IsFalse("Book".ContainsFarsiData());
```

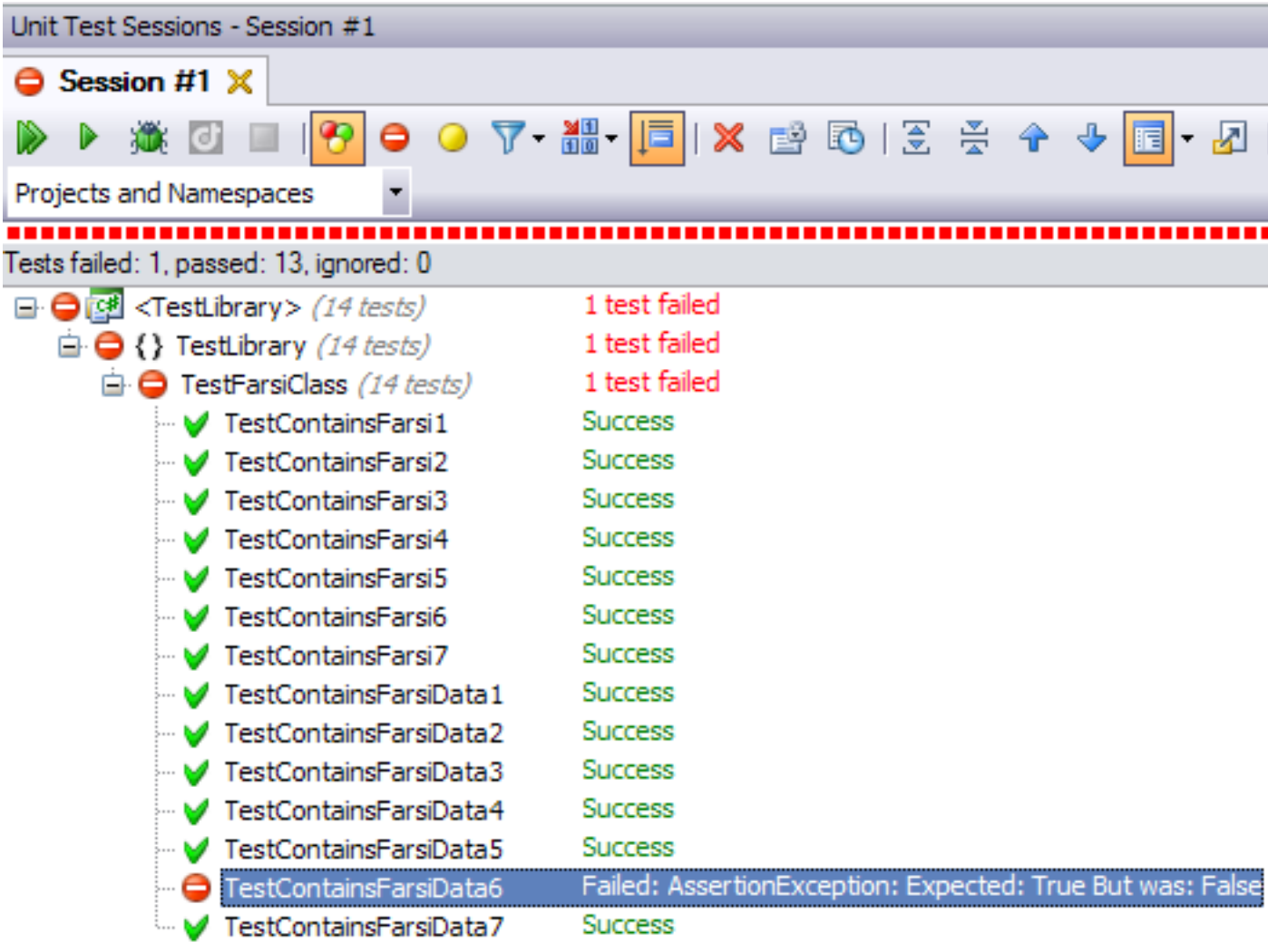
```
        }

        [Test]
        public void TestContainsFarsiData6()
        {
            Assert.IsTrue("\۳۸۷".ContainsFarsiData());
        }

        [Test]
        public void TestContainsFarsiData7()
        {
            Assert.IsFalse("Здравствуйте!".ContainsFarsiData()); //Russian hello!
        }
    }
}
```

در کلاس فوق هر دو متد را با آزمایش‌های واحد مختلفی بررسی کرده‌ایم، انواع و اقسام حروف فارسی، ترکیبی از فارسی و انگلیسی، ترکیبی از فارسی و اعداد انگلیسی، عبارت کاملا انگلیسی، عدد کاملا فارسی و یک عبارت روسی! (در یک کلاس عمومی با متدهای عمومی بدون آرگومان از نوع void)

کلاس CDetectFarsi در برنامه اصلی قرار دارد و کلاس TestFarsiClass در یک پروژه class library دیگر قرار گرفته است (در این مورد و جدا سازی آزمایش‌ها از پروژه اصلی در قسمت‌های قبل بحث شد)

همچنین به ازای هر عبارت Assert یک متد ایجاد گردید تا شکست یکی، سبب اختلال در بررسی سایر موارد نشود.

نتیجه اجرای این آزمایش واحد با استفاده از امکانات مجتمع افزونه ReSharper به صورت زیر است:



منهای یک مورد، سایر آزمایشات ما با موفقیت انجام شده‌اند. موردی که با شکست مواجه شده، بررسی اعداد کاملا فارسی است که البته در الگوی دوم لحاظ نشده است و انتظار هم نمی‌رود که آن‌را به این شکل پشتیبانی کند.
برای اینکه در حین اجرای آزمایشات بعدی این متد در نظر گرفته نشود، می‌توان ویژگی Test آن‌را به صورت زیر تغییر داد:

```
[Test,Ignore]
```

نکته: مرسوم شده است که نام متدهای آزمایش واحد به صورت زیر تعریف شوند (با Test شروع شوند، در ادامه نام متدی که بررسی می‌شود ذکر گردد و در آخر ویژگی مورد بررسی عنوان شود):

```
Test[MethodToBeTested][SomeAttribute]
```

ادامه دارد...

نظرات خوانندگان

نویسنده: مسعود
تاریخ: ۱۳۸۷/۱۰/۲۱ ۲۰:۵۷:۰۰

واقعا جالب !

کیف کردم ، تا حالا یه چند باری خواستم برم سراغ این Unit Testing ولی اصلا وقت نشد.

ممنون استاد