网站平台由ASP.NET 5.0迁移到.NET 6.0

Published on Nov 16, 2021 UTC by Wang Ye <https://wangye.org>

本网站自从3月上线以来已经有8个月时间，期间也有各种bug，也勉强修复了，最初网站运行在.NET 5.0平台上，后来一直关注着.NET 6.0的进展，在微软正式Release的时候立马开始适配新的6.0平台，期间也遇到过各种坑，留有这篇文章仅做记载。

1 安装Visual Studio 2022

因为Visual Studio 2019不再支持NET 6.0 SDK，如果想继续在VS平台下开发，必须升级到Visual Studio 2022，对于个人来说我们选择Community版本就可以了，安装完成后选择用最新的VS2022打开你的项目。

2 升级NuGet Package到最新版本

由于我是等到.NET 6.0正式版出来的时候升级的，大部分的第三方包（Package）已经适配了.NET 6.0，可以直接升级，但仍然建议各位查阅项目官方的文档，特别是breaking changes以及用户提交的issues，确认没有引入新的bug或者不兼容的情况存在。

3 存在的问题及解决办法

3.1 Npgsql 6.0的Breaking changes

这个地方和旧版本兼容性问题较多，具体可以参考官方文档，我首先遇到的问题就是带时间域的时间戳和不带时间域的时间戳转换问题，新版本的Npgsql严格了这两种转换，如果不明确指定日期格式就会运行报异常，具体如下：

```
System.InvalidCastException: 'Cannot write DateTime with
Kind=Unspecified to PostgreSQL type 'timestamp with time zone', only
UTC is supported. Note that it's not possible to mix DateTimes with
```

```
different Kinds in an array/range. See the
Npgsql.EnableLegacyTimestampBehavior AppContext switch to enable
legacy behavior.'
```

对于PostgreSQL数据库来说UTC时间戳以`timestamptz`类型表示，对于.NET来说UTC时间戳以带`Kind=Utc`的`DateTime`类型表示，值得注意的是`DateTime`这种类型在CLR类型中既可以表示UTC时间戳也可以表示非UTC时间戳，这就存在.NET的DateTime类型到PostgreSQL的`timestamp`和`timestamptz`转换的混淆，我们必须在EF Core中做出配置来决定如何转换，旧版本的Npgsql对于这种转换往往是隐式的，这往往会带来一些不可预料的程序bug或者安全问题，现在新版本的Npgsql对此要求显式指派时间戳或者日期类型。

解决方法有两个，通过在迁移文件里加入代码`migrationBuilder.Sql("SET TimeZone='UTC'");`并重新执行迁移程序，或者告诉Npgsql采用旧的兼容模式，具体通过在配置Npgsql服务前加上如下代码：

```
public void ConfigureServices(IServiceCollection services)
{
    // ... 省略其他代码
    AppContext.SetSwitch("Npgsql.EnableLegacyTimestampBehavior",
true);
    AppContext.SetSwitch("Npgsql.DisableDateTimeInfinityConversions",
true);
    // ... 省略其他代码
}
```

关于这个问题的讨论除了可以参考官方文档外还可以看这个贴子[《.NET 6 RC1 and issues with datetime fields #2000》]。

除了日期问题，还遇到一个Npgsql的类型转换异常，具体报错如下：

```
Unable to cast object of type
'Npgsql.EntityFrameworkCore.PostgreSQL.Storage.Internal.Mapping.Npgsql
ByteArrayTypeMapping' to type
```

```
'Npgsql.EntityFrameworkCore.PostgreSQL.Storage.Internal.Mapping.NpgsqlArrayTypeMapping'.
```

这个异常导致的原因主要是Linq写法错用了int数组，比如包含id为1、2、3的记录条数，之前是直接用数组`int[]`，然后 调用数组`Contains`方法，这样在Npgsql翻译成SQL语句的时候容易报这类异常，将数组改成`List`类型即可解决。

3.2 System.Drawing.Common不再跨平台

这个问题非常令我感觉意外，毕竟在.NET 5.0和之前.NET Core版本，`System.Drawing.Common`一直作为独立的跨平台package存在的，但是到.NET 6.0时代，这个package则变成了Windows Only，不太清楚微软葫芦里卖的什么药，具体报异常内容如下：

```
System.PlatformNotSupportedException : System.Drawing.Common is not supported on non-Windows platforms. See https://aka.ms/systemdrawingnonwindows for more information.
```

微软在文档《System.Drawing.Common only supported on Windows》也指出了这个Breaking changes以及change的原因，大概是这个package主要还是为Windows设计的，跨平台不是重点，支撑其跨平台的native库libgdiplus包含大量未经测试的代码以及引用了第三方库，使其维护变得困难，因此微软团队不得不放弃对其跨平台支持。

好在这个问题可以通过引入第三方package来解决，这里推荐SixLabors.ImageSharp，API和`System.Drawing.Common`类似，修改成本不高，本站正是通过此package完成升级的。

3.3 第三方验证码无法读取系统字体

本站验证码使用了由Edi Wang开发的Edi.Captcha.AspNetCore，此验证码package升级到.NET 6.0后也采用了SixLabors.ImageSharp依赖，但是作者在生成验证码字体时使用了`SystemFonts.CreateFont`API直接硬编码了Arial字体，如

果程序运行在Windows环境下倒没有问题，毕竟Windows都带Arial字体，但是运行在Linux下尤其是Docker下，就会报`SixLabors.Fonts.Exceptions.FontFamilyNotFoundException`异常，从而验证码无法显示，本着尽量不魔改第三方package的精神，解决此问题最简单直接的办法就是安装Arial字体，两种方式，一是直接把Arial字体打包进Docker镜像；二是参考[《Install fonts in Linux container for ASP.NET Core》]问答直接安装ttf-mscorefonts-installer的Debian包（Debian环境），对于Dockerfile来说，根据实际情况增加下面的配置：

```
RUN sed -i'.bak' 's/$/ contrib/' /etc/apt/sources.list && \
        apt-get update && \
        apt-get install -y ttf-mscorefonts-installer fontconfig && \
        fc-cache -f -v
```

顺利解决问题，目前所有.NET 5.0迁移到6.0的问题已经全部解决，网站运行平稳，如果后续有其他问题我再补充，另外再推荐一篇文章[《.NET 6, A guide for the high impact Breaking Changes》]。

External References (Links)

Visual Studio 2022
https://visualstudio.microsoft.com/launch/
官方文档
https://www.npgsql.org/doc/release-notes/6.0.html#breaking-changes
《.NET 6 RC1 and issues with datetime fields #2000》
https://github.com/npgsql/efcore.pg/issues/2000
《System.Drawing.Common only supported on Windows》
https://docs.microsoft.com/en-us/dotnet/core/compatibility/core-libraries/6.0/system-drawing-common-windows-only
SixLabors.ImageSharp
https://github.com/SixLabors/ImageSharp
Edi Wang

https://edi.wang/

Edi.Captcha.AspNetCore

https://github.com/EdiWang/Edi.Captcha.AspNetCore

《Install fonts in Linux container for ASP.NET Core》

https://stackoverflow.com/questions/60934639/install-fonts-in-linux-container-for-asp-net-core

《.NET 6, A guide for the high impact Breaking Changes》

https://blog.georgekosmidis.net/2021/11/14/net-6-a-guide-for-the-high-impact-breaking-changes/