**理论：byte与各种编码**

在开始前，先了解byte与各个编码之间的关系

**Byte**

byte是8位2进制，所有的数据都是由byte组成，如：

图片是一连串的byte

ASCII字符”A”是一个byte

可以说byte是我们的原始数据

**编码**

如下问题：

这是2个byte“01100010 00010001”，它代表了什么？

答案：

如果没有指明编码，那他什么都不代表

在ASCII编码规则中指明：“01100010”代表”b”，”00010001”代表控制字符VT

所有你可以认为这2个byte是b+VT

在Unicode编码中指明“01100010 00010001”代表汉字“我”

所有你也可以认为这2个byte是我

编码是约定好的，物理在Linux还是Window，其编码的byte都是相同的

**ASCII编码**

ASCII属于字符编码

ASCII以一个byte进行编码，从0-127（即00000000-01111111），分别代表不同的字符，

如97（01100001）代表a

**Unicode字符集**

Unicode字符集对全世界的字符进行了编号，如25105（十进制）代表汉字“我”

ASCII字符的位置在Unicode字符集还是一样（即0-127与ASCII编码相同）

**Unicode编码（UTF-16）**

标准的Unicode编码称为UTF-16

Unicode编码直接以Unicode字符的编号进行编码

Unicode编码以2个byte进行编码，如“01100010 00010001”（25105）代表汉字“我”

2个byte最多只能编码6万多个字符，但世界的字符可不止6万个，因此Unicode预留了一些编码范围（以11011开头的编码），如编码 \ud801（16进制）其不代表任何字符，其需要以后面的编码组成一个编码，如 \ud801\udccf 组合在一起，代表一个字符

**UTF-8编码**

UTF-8基于Unicode字符集

UTF一个字符编码使用的byte不固定，1-8个byte不等

一个英文UTF-8字符占1byte，一个汉字占3byte

UTF-8部分编码规则如下，其中xxxx就是Unicode字符的编号：

1字节 0xxxxxxx （编号小于127（即111 1111）使用一个字节，）

2字节 110xxxxx 10xxxxxx

3字节 1110xxxx 10xxxxxx 10xxxxxx

**UTF-32编码**

UTF-32基于Unicode字符集，直接以Unicode字符的编号进行编码

UTF-32使用固定4个byte对Unicode字符集进行编码

**Base64**

问题：

11110111这个byte我像以字符的形式保存，但这个byte在ASCII，Unicode，UTF-8中均找不到对应的字符编码，怎么办？

Base64是一种转换规则，其将3个byte分成4个组，每组6位，然后将这4个6位对照Base64编码表，如：010011对应T，以T的ASCII码对应的byte进行存储

Base64可以将任意的byte转成的可显示ASCII，这有利与我们对一些非字符编码的数据转成ASCII字符进行保存或传输

**image/jpeg**

image/jpeg是jpg图片的编码格式，其保存的是一系列的byte

如何在两个服务器之间传输jpg图片？

我们读取图片（即读取一系列的byte），然后将这些byte发给对方，并告诉他这是image/jpeg编码格式（即这是一张jpg图片），对方服务器可以原封不动的将这些byte保存到一个文件中，并将文件的后缀改为 .jpg即可

**byte，Rune，char，string的关系**

.net 采用unicode（utf-16）编码，也就是说 char，Rune，string 都是针对utf-16编码的，他们的数据byte就是utf-16编码

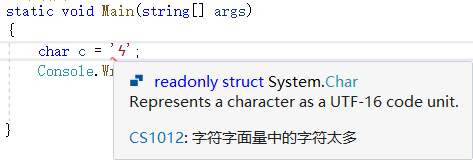
**char**

char由2个byte组成

char c = '我';

Console.WriteLine($"{(c & 0xFF00) >> 8}|{c & 0x00FF}");

对于需要4个byte的字符，无法赋值给char，如下



**Rune**

Rune代表utf16的一个字符，如下代码

// 正确

Rune r = new Rune('a');

Rune r2 = new Rune('\ud801', '\udccf');

// 报错，utf16 编码中 \ud801 不表示任何字符

Rune r3 = new Rune('\ud801');

**string**

string是由char组成的，当然也可以任务string是由Rune组成的，其Length为包含char的个数，如下代码解释他们的关系

string s = "\ud801\udccf";

// 访问string的char，s由2个char组成

foreach (char c in s) {

    Console.WriteLine((int)c);

}

// 访问 string 的 Rune，s由1个Rune组成

foreach (Rune r in s.EnumerateRunes()) {

    Console.WriteLine(r);

}

**编码转换**

System.Text.Encoding类包含各种编码的转换方法，如下示例utf8

// 将string转为utf8

byte[] utf8b = System.Text.Encoding.UTF8.GetBytes(s);

// 将utf8转为string

string s2 = System.Text.Encoding.UTF8.GetString(utf8b);

// 将string转为utf16，实际上string的byte就是utf16，所以这里只是提交string的byte而已

byte[] utf16b = System.Text.Encoding.Unicode.GetBytes(s);

// 将utf16转为string

string s3 = System.Text.Encoding.Unicode.GetString(utf16b);

注：除utf16外其他编码都只能使用byte保存，Encoding.UTF8.GetString(utf8b) 返回的不是utf8的字符串，而是将utf8的byte转为utf16的byte，再利用utf16 byte生成字符串