Go语言 标准库 unicode/utf8包

Constants常量

```
const (
    RuneError = '\uFFFD'     // 错误的Rune或"Unicode replacement character"
    RuneSelf  = 0x80         // 低于RunSelf的字符用代表单字节的同一值表示
    MaxRune   = '\U0010FFFF' // 最大的合法unicode码值
    UTFMax    = 4            // 最大的utf-8编码的unicode字符的长度
)
```

示例：

```
func main() {
    var RuneError rune = '\uFFFD'
    var RuneSelf rune = 0x80
    var MaxRune rune = '\U0010FFFF'
    var UTFMax rune = 4
    fmt.Println(RuneError) //65533
    fmt.Println(RuneSelf)  //128
    fmt.Println(MaxRune)   //1114111
    fmt.Println(UTFMax)    //4
}
```

DecodeRune函数

函数解码p开始位置的第一个utf-8编码的码值，返回该码值和编码的字节数。
如果编码不合法，会返回(RuneError, 1)。该返回值在正确的utf-8编码情况下是不可能返回的。
如果一个utf-8编码序列格式不正确，或者编码的码值超出utf-8合法码值的范围，或者不是该码值的最短编码，该编码序列即是不合法的。函数不会执行其他的验证。

```
func DecodeRune(p []byte) (r rune, size int)
```

示例：

```
func main() {
    m := []byte{'a', 'b'}
    a, b := utf8.DecodeRune(m)
    fmt.Println(a) //91
    fmt.Println(b) //1
}
```

DecodeRuneInString函数

函数类似DecodeRune但输入参数是字符串。

```
func DecodeRuneInString(s string) (r rune, size int)
```

示例：

```
func main() {
    s := "世界"
    a, b := utf8.DecodeRuneInString(s)
    fmt.Println(s[0])     //228
    fmt.Println(a)        //19990
    fmt.Println(b)        //3
    fmt.Printf("%b\n", a) //100111000010110
}
```

```
func main() {
    str := "Hello, 世界"
    for len(str) > 0 {
        r, size := utf8.DecodeRuneInString(str)
        fmt.Printf("%c %v\n", r, size)
        str = str[size:]
    }
    /*
        H 1
        e 1
        l 1
        l 1
        o 1
        , 1
          1
        世 3
        界 3
    */
}
```

ValidRune

判断r是否可以编码为合法的utf-8序列。

```
func ValidRune(r rune) bool
```

示例：

```
func main() {
    var a rune = '包子'
    b := utf8.ValidRune(a)
    fmt.Println(b) //true
}
```

RuneLen

返回r编码后的字节数。如果r不是一个合法的可编码为utf-8序列的值，会返回-1。

```
func RuneLen(r rune) int
```

RuneStart

报告字节b是否可以作为某个rune编码后的第一个字节。第二个即之后的字节总是将左端两个字位设为10。

```
func RuneStart(b byte) bool
```

FullRune

报告切片p是否以一个码值的完整utf-8编码开始。不合法的编码因为会被转换为宽度1的错误码值而被视为完整的。

```
func FullRune(p []byte) bool
```

示例：

```
func main() {
    m := []byte{'a', 'b'}
    a := utf8.FullRune(m)
    fmt.Println(a) //true
}
```

FullRuneInString

函数类似FullRune但输入参数是字符串。

```
func FullRuneInString(s string) bool
```

RuneCount

返回p中的utf-8编码的码值的个数。错误或者不完整的编码会被视为宽度1字节的单个码值。

```
func RuneCount(p []byte) int
```

RuneCountInString

函数类似RuneCount但输入参数是一个字符串。

```
func RuneCountInString(s string) (n int)
```

示例：

```
func main() {
    a := utf8.RuneCountInString("周杰伦")
    fmt.Println(a) //3
}
```

Valid

返回切片p是否包含完整且合法的utf-8编码序列。

```
func Valid(p []byte) bool
```

ValidString

报告s是否包含完整且合法的utf-8编码序列。

```
func ValidString(s string) bool
```

EncodeRune

EncodeRune将r的utf-8编码序列写入p（p必须有足够的长度），并返回写入的字节数。

```
func EncodeRune(p []byte, r rune) int
```

DecodeLastRune

函数解码p中最后一个utf-8编码序列，返回该码值和编码序列的长度。

```
func DecodeLastRune(p []byte) (r rune, size int)
```

DecodeLastRuneInString

函数类似DecodeLastRune但输入参数是字符串。

```
func DecodeLastRuneInString(s string) (r rune, size int)
```