不同语言编译时、运行时的编码方式

三种字符的编码

选择字母、希腊字母、中文三种字符，查询其编码如下：

z U+007A

编码	hex	dec (bytes)	dec	binary
UTF-8	7A	122	122	01111010
UTF-16BE	00 7A	0 122	122	00000000 01111010
UTF-16LE	7A 00	122 0	31232	01111010 00000000
UTF-32BE	00 00 00 7A	0 0 0 122	122	00000000 00000000 00000000 01111010
UTF-32LE	7A 00 00 00	122 0 0 0	2046820352	01111010 00000000 00000000 00000000

Ω U+03A9

编码	hex	dec (bytes)	dec	binary
UTF-8	CE A9	206 169	52905	11001110 10101001
UTF-16BE	03 A9	3 169	937	00000011 10101001
UTF-16LE	A9 03	169 3	43267	10101001 00000011
UTF-32BE	00 00 03 A9	0 0 3 169	937	00000000 00000000 00000011 10101001
UTF-32LE	A9 03 00 00	169 3 0 0	2835546112	10101001 00000011 00000000 00000000

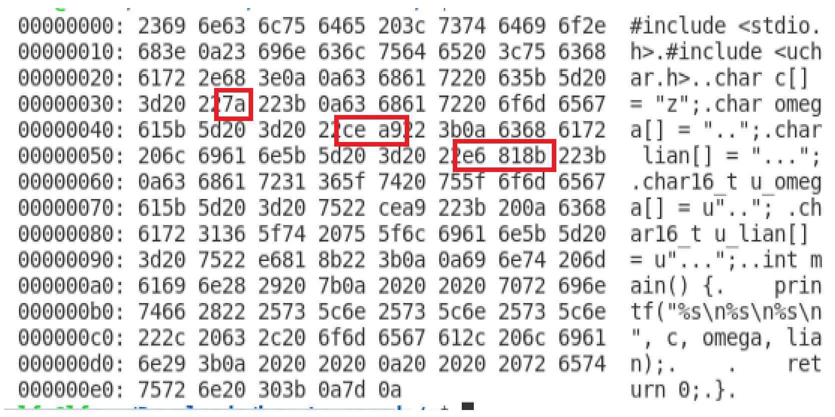
恋 U+604B

编码	hex	dec (bytes)	dec	binary
UTF-8	E6 81 8B	230 129 139	15106443	11100110 10000001 10001011
UTF-16BE	60 4B	96 75	24651	01100000 01001011
UTF-16LE	4B 60	75 96	19296	01001011 01100000
UTF-32BE	00 00 60 4B	0 0 96 75	24651	00000000 00000000 01100000 01001011
UTF-32LE	4B 60 00 00	75 96 0 0	1264582656	01001011 01100000 00000000 00000000

C语言

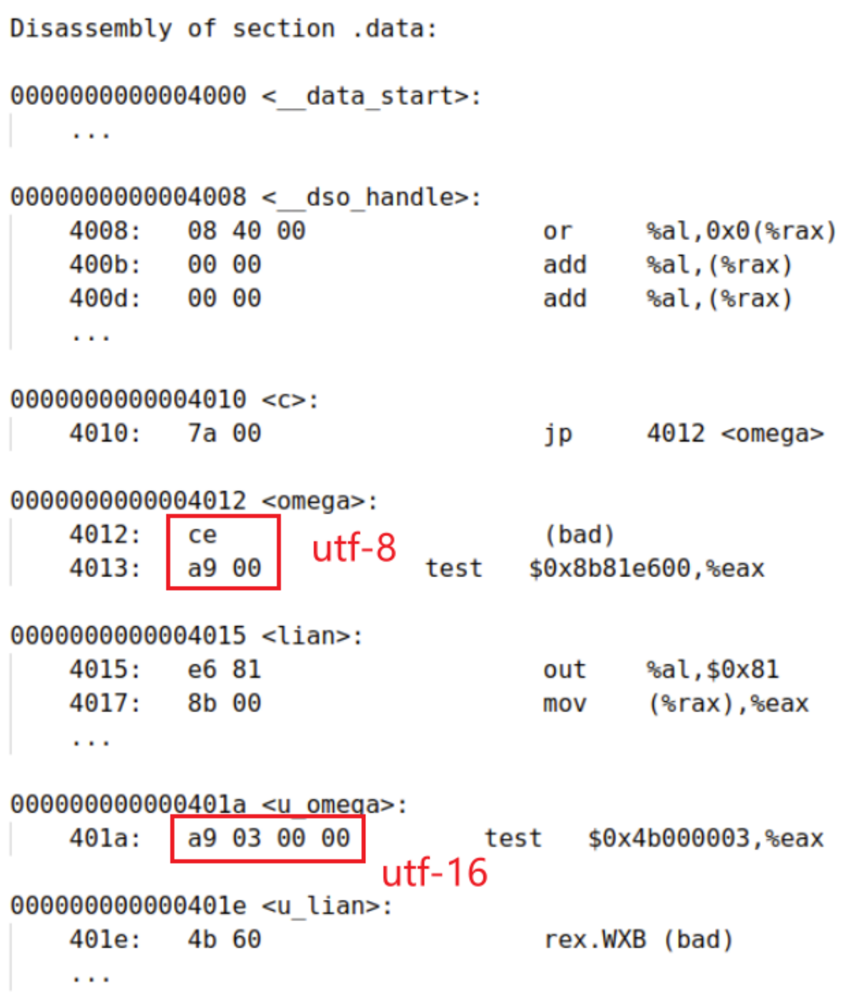
对源代码做十六进制转储，得到其编码方式为utf-8：

```c
#include <stdio.h>
#include <uchar.h>

char c[] = "z";
char omega[] = "Ω";
char lian[] = "恋";
char16_t u_omega[] = u"Ω";
char16_t u_lian[] = u"恋";

int main() {
    printf("%s\n%s\n%s\n", c, omega, lian);

    return 0;
}
```



```
00000000: 2369 6e63 6c75 6465 203c 7374 6469 6f2e  #include <stdio.
00000010: 683e 0a23 696e 636c 7564 6520 3c75 6368  h>.#include <uch
00000020: 6172 2e68 3e0a 0a63 6861 7220 635b 5d20  ar.h>..char c[]
00000030: 3d20 227a 223b 0a63 6861 7220 6f6d 6567  = "z";.char omeg
00000040: 615b 5d20 3d20 22ce a922 3b0a 6368 6172  a[] = "..";.char
00000050: 206c 6961 6e5b 5d20 3d20 22e6 818b 223b   lian[] = "...";
00000060: 0a63 6861 7231 365f 7420 755f 6f6d 6567  .char16_t u_omeg
00000070: 615b 5d20 3d20 7522 cea9 223b 200a 6368  a[] = u"..."; .ch
00000080: 6172 3136 5f74 2075 5f6c 6961 6e5b 5d20  ar16_t u_lian[]
00000090: 3d20 7522 e681 8b22 3b0a 0a69 6e74 206d  = u"...";..int m
000000a0: 6169 6e28 2920 7b0a 2020 2020 7072 696e  ain() {.    prin
000000b0: 7466 2822 2573 5c6e 2573 5c6e 2573 5c6e  tf("%s\n%s\n%s\n
000000c0: 222c 2063 2c20 6f6d 6567 612c 206c 6961  ", c, omega, lia
000000d0: 6e29 3b0a 2020 2020 0a20 2020 2072 6574  n);.    .    ret
000000e0: 7572 6e20 303b 0a7d 0a                   urn 0;.}.
```

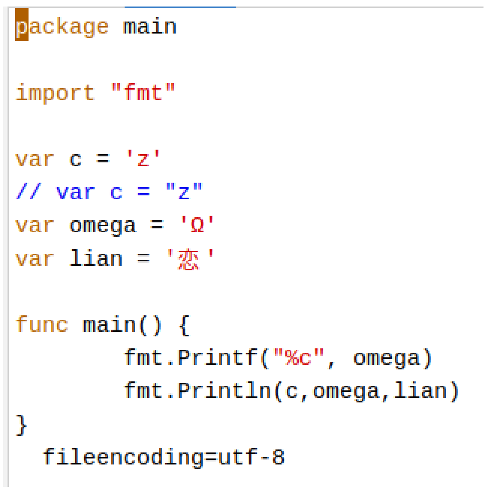
对编译文件做反汇编，得到其默认编码格式是utf-8，可指定为其他格式，如utf-16：



```
Disassembly of section .data:

0000000000004000 <__data_start>:
    ...

0000000000004008 <__dso_handle>:
    4008:   08 40 00                or     %al,0x0(%rax)
    400b:   00 00                   add    %al,(%rax)
    400d:   00 00                   add    %al,(%rax)
    ...

0000000000004010 <c>:
    4010:   7a 00                   jp     4012 <omega>

0000000000004012 <omega>:
    4012:   ce                      (bad)
    4013:   a9 00               test   $0x8b81e600,%eax

0000000000004015 <lian>:
    4015:   e6 81                   out    %al,$0x81
    4017:   8b 00                   mov    (%rax),%eax
    ...

000000000000401a <u_omega>:
    401a:   a9 03 00 00         test   $0x4b000003,%eax

000000000000401e <u_lian>:
    401e:   4b 60                   rex.WXB (bad)
    ...
```

Go

源代码文件编码方式为utf–8：



```
package main

import "fmt"

var c = 'z'
// var c = "z"
var omega = 'Ω'
var lian = '恋'

func main() {
        fmt.Printf("%c", omega)
        fmt.Println(c,omega,lian)
}
  fileencoding=utf-8
```

对编译文件做反汇编，得到其编码格式是utf-16：

```
000000000053f024 <main.c>:
  53f024:    7a 00                   jp     53f026 <main.c+0x2>
    ...

000000000053f028 <main.lian>:
  53f028:    4b 60                   rex.WXB (bad)
    ...

000000000053f02c <main.omega>:
  53f02c:    a9 03 00 00         test   $0x8000003,%eax
```

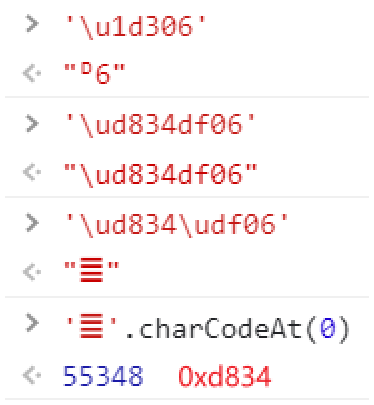
JavaScript

JavaScript使用的编码方式是UCS-2，是UTF-16的子集。
两个字节以内，UCS-2与UTF-16一致；多于两个字节时，UCS-2会将原字符视作由两个字符组成的字符串。
比如，字符'`𝌆`'编码如下：

𝌆 U+1D306

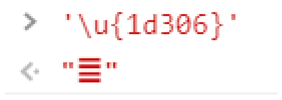
编码	hex	dec (bytes)	dec	binary
UTF-8	F0 9D 8C 86	240 157 140 134	4036856966	11110000 10011101 10001100 10000110
UTF-16BE	D8 34 DF 06	216 52 223 6	3627343622	11011000 00110100 11011111 00000110
UTF-16LE	34 D8 06 DF	52 216 6 223	886572767	00110100 11011000 00000110 11011111
UTF-32BE	00 01 D3 06	0 1 211 6	119558	00000000 00000001 11010011 00000110
UTF-32LE	06 D3 01 00	6 211 1 0	114491648	00000110 11010011 00000001 00000000

在JavaScript中，无法通过码点`1d306`或utf-16编码`d834df06`得到该字符，因为JS将其视作字符`d834`和`df06`构成的字符串：



```
> '\u1d306'
<· "ᴰ6"
> '\ud834df06'
<· "\ud834df06"
> '\ud834\udf06'
<· "𝌆"
> '𝌆'.charCodeAt(0)
<· 55348   0xd834
```

新版本ES6大幅增强了对Unicode的支持，通过大括号能够直接表示4字节码点的字符，新增了专门处理4字节码点的函数等等：



```
> '\u{1d306}'
<· "𝌆"
```

参考：http://www.ruanyifeng.com/blog/2014/12/unicode.html