

Локализация и интернационализация программного обеспечения

Инструментарий Современного Программиста

Иван Трепаков

NSU Sys.Pro

American Standard Code for Information Interchange

```
$ ascii -d
```

0 NUL	16 DLE	32	48 0	64 @	80 P	96 `	112 p
1 SOH	17 DC1	33 !	49 1	65 A	81 Q	97 a	113 q
2 STX	18 DC2	34 "	50 2	66 B	82 R	98 b	114 r
3 ETX	19 DC3	35 #	51 3	67 C	83 S	99 c	115 s
4 EOT	20 DC4	36 \$	52 4	68 D	84 T	100 d	116 t
5 ENQ	21 NAK	37 %	53 5	69 E	85 U	101 e	117 u
6 ACK	22 SYN	38 &	54 6	70 F	86 V	102 f	118 v
7 BEL	23 ETB	39 '	55 7	71 G	87 W	103 g	119 w
8 BS	24 CAN	40 (56 8	72 H	88 X	104 h	120 x
9 HT	25 EM	41)	57 9	73 I	89 Y	105 i	121 y
10 LF	26 SUB	42 *	58 :	74 J	90 Z	106 j	122 z
11 VT	27 ESC	43 +	59 ;	75 K	91 [107 k	123 {
12 FF	28 FS	44 ,	60 <	76 L	92 \	108 l	124
13 CR	29 GS	45 -	61 =	77 M	93]	109 m	125 }
14 SO	30 RS	46 .	62 >	78 N	94 ^	110 n	126 ~
15 SI	31 US	47 /	63 ?	79 O	95 _	111 o	127 DEL

ASCII

00:
10:
20: `!"#$%&'()*+,-./`
30: `0123456789:;<=>?`
40: `@ABCDEFGHIJKLMNO`
50: `PQRSTUVWXYZ[\]^_`
60: ``abcdefghijklmno`
70: `pqrstuvwxyz{|}~`

ASCII

00:		80:
10:		90:
20:	!"#\$%&'()*+,-./	a0:
30:	0123456789:;<=>?	b0:
40:	@ABCDEFGHIJKLMNO	c0:
50:	PQRSTUVWXYZ[\]^_	d0:
60:	`abcdefghijklmno	e0:
70:	pqrstuvwxyz{ }~	f0:

Расширения ASCII

ASCII

00:
10:
20: `!"#$%&'()*+,-./`
30: `0123456789:;<=>?`
40: `@ABCDEFGHIJKLMNO`
50: `PQRSTUVWXYZ[\]^_`
60: ``abcdefghijklmno`
70: `pqrstuvwxyz{|}~`

CP866

80: АБВГДЕЖЗИЙКЛМНОП
90: РСТУФХЦЧШЩЪЫЬЭЮЯ
a0: абвгдежзийклмно
b0:
c0:
d0:
e0: рстуфхцчшщъыьэюя
f0: ЁёЄєİïÛű°•·√№¤■

MS-DOS

KOI8-R

-| □ □ □ □ □ □ □ □
▒ ▓ • √ ≈ ≤ ≥] ° ² · ÷
= || ꝑ ё ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ
|| ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ ꝑ
юабцдефгхийклмно
пярстужввызшэщчъ
ЮАБЦДЕФГХИЙКЛМНО
ПЯРСТУЖВВЫЗШЭЩЧЪ

Unix

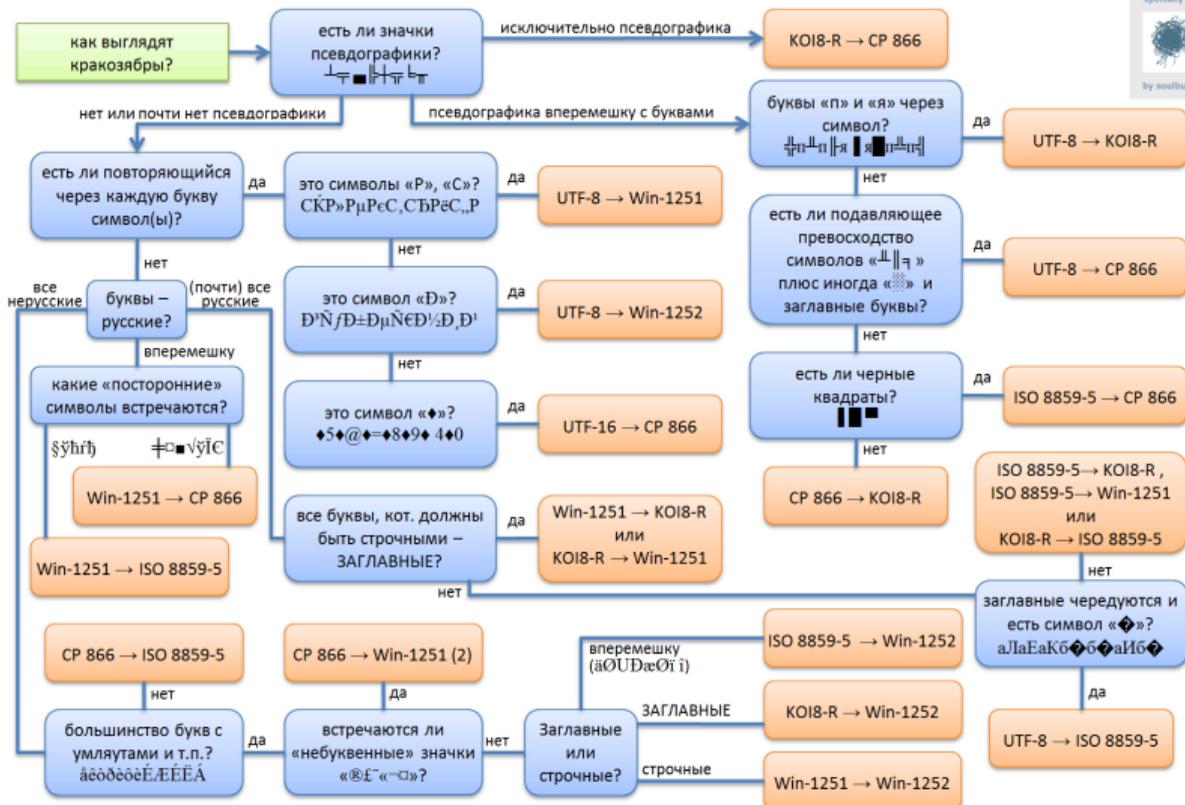
CP1251

ђѓ,ѓ,„…†‡€%&‹›ќџџ
ђ' ' ' ' ' • — ™ ђ › њ ѓ џ џ
ЎўЈџГ|§ЁЄ«¬-®Ї
°±Іігµ¶•ё№є»јSsї
АБВГДЕЖЗИЙКЛМНОП
РСТУФХЦЧШЩЪЫЬЭЮЯ
абвгдежзийклмно
рстуфхцчшщъыьэюя

Windows

оПХБЕР ЛХП!

ОПХБЕР ЛХП!

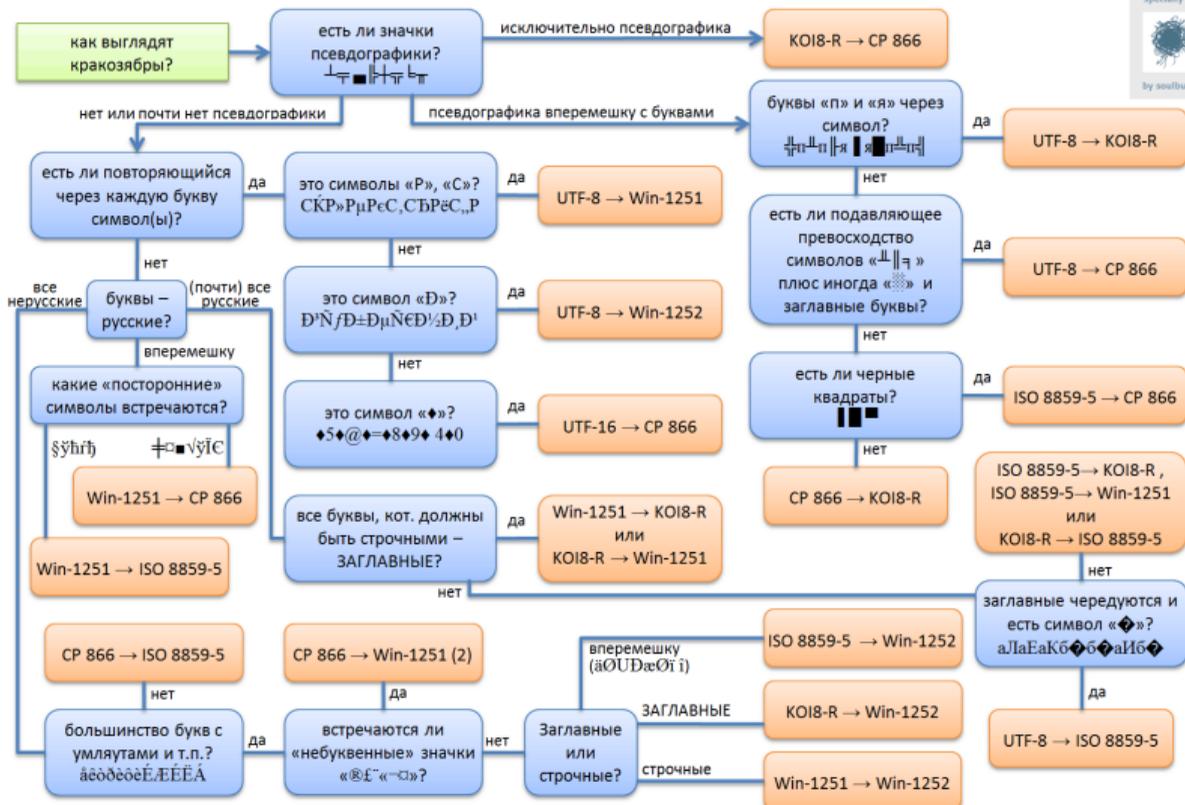


Крокозябры

оПХБЕР ЛХП!

```
$ echo "оПХБЕР ЛХП!" \  
| iconv -t cp1251 \  
| iconv -f koi8-r  
Ноуаеп куо!
```

```
$ echo "оПХБЕР ЛХП!" \  
| iconv -t koi8-r \  
| iconv -f cp1251  
Привет мир!
```



Крокозябры

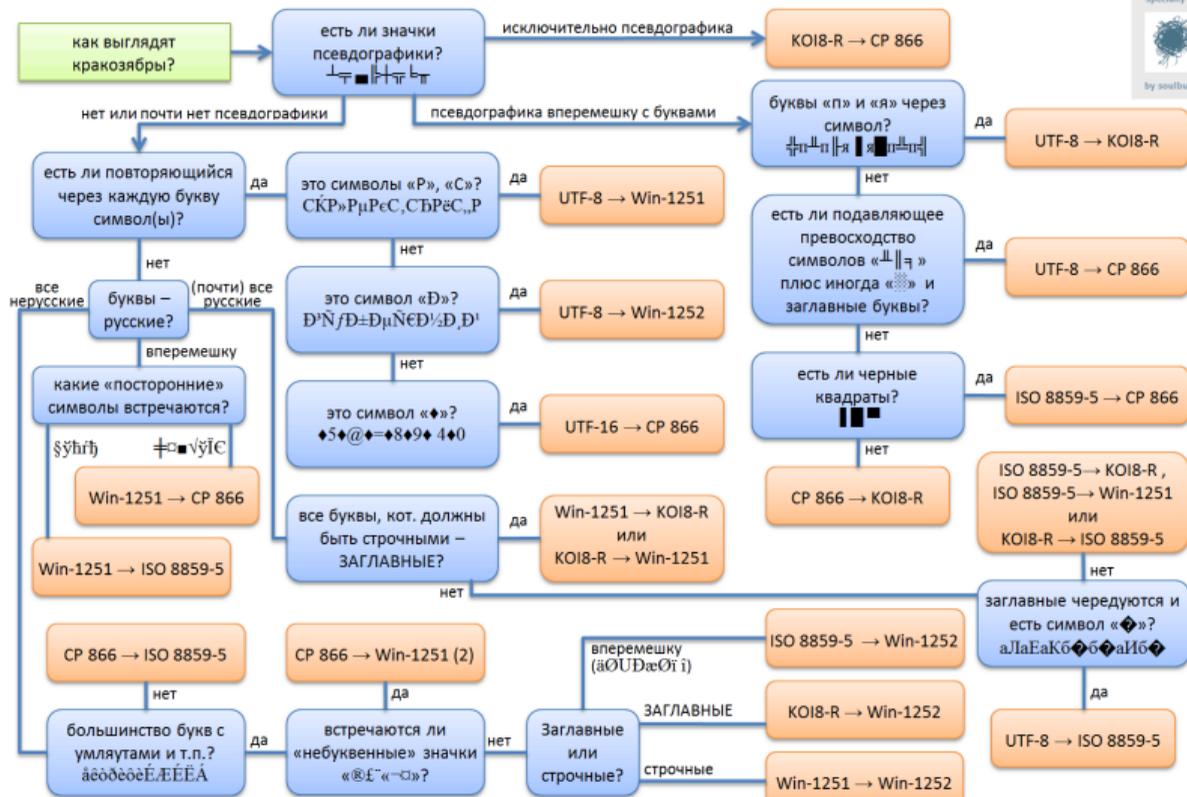
оПХБЕР ЛХП!

```
$ echo "оПХБЕР ЛХП!" \  
| iconv -t cp1251 \  
| iconv -f koi8-r  
Ноуап кую!
```

```
$ echo "оПХБЕР ЛХП!" \  
| iconv -t koi8-r \  
| iconv -f cp1251  
Привет мир!
```

Инструменты

- iconv
- encs
- ucharDET



Unicode Standard

- Универсальное кодирование символов
- Количество различных кодов 1 114 112
- **Unicode 16.0** (2024 год) использует лишь 154 998

Цели

- **Универсальность** Содержит все возможные символы современных и древних языков, технических текстов, диакритику и emoji
- **Эффективность** *Plain text* кодирование в одном из трех стандартных форматов: **UTF-32**, **UTF-16**, **UTF-8**
- **Однозначность** Каждый код *однозначно* соответствует единственному символу



Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Hello 🌍!

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `uniname` (часть пакета `uniutils`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Hello 🌍!

U+0048	H	LATIN CAPITAL LETTER H
U+0065	e	LATIN SMALL LETTER E
U+006C	l	LATIN SMALL LETTER L
U+006C	l	LATIN SMALL LETTER L
U+006F	o	LATIN SMALL LETTER O
U+0020		SPACE
U+1F30E	🌍	EARTH GLOBE AMERICAS
U+0021	!	EXCLAMATION MARK

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `uname` (часть пакета `uniutils`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Hello 🌍!

```
$ printf 'Hello \U1F30E!' | uname
U+0048  H   LATIN CAPITAL LETTER H
U+0065  e   LATIN SMALL LETTER E
U+006C  l   LATIN SMALL LETTER L
U+006C  l   LATIN SMALL LETTER L
U+006F  o   LATIN SMALL LETTER O
U+0020           SPACE
U+1F30E 🌍   EARTH GLOBE AMERICAS
U+0021  !   EXCLAMATION MARK
```

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Привет 🌍!

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `uniname` (часть пакета `uniutils`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Привет 🌍!

U+041F	П	CYRILLIC CAPITAL LETTER PE
U+0440	р	CYRILLIC SMALL LETTER ER
U+0438	и	CYRILLIC SMALL LETTER I
U+0432	в	CYRILLIC SMALL LETTER VE
U+0435	е	CYRILLIC SMALL LETTER IE
U+0442	т	CYRILLIC SMALL LETTER TE
U+0020		SPACE
U+1F30E		EARTH GLOBE AMERICAS
U+0021	!	EXCLAMATION MARK

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '☺' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Привет 🌍!

```
$ printf 'Привет \U1F30E!' | unicode
U+041F П CYRILLIC CAPITAL LETTER PE
U+0440 р CYRILLIC SMALL LETTER ER
U+0438 и CYRILLIC SMALL LETTER I
U+0432 в CYRILLIC SMALL LETTER VE
U+0435 е CYRILLIC SMALL LETTER IE
U+0442 т CYRILLIC SMALL LETTER TE
U+0020 SPACE
U+1F30E 🌍 EARTH GLOBE AMERICAS
U+0021 ! EXCLAMATION MARK
```

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Йо-йо

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '○' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = 'o' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Йо-йо

U+0439	Й	CYRILLIC CAPITAL LETTER SHORT I
U+043E	о	CYRILLIC SMALL LETTER O
U+002D	-	HYPHEN-MINUS
U+0438	и	CYRILLIC SMALL LETTER I
U+0306	◌̆	COMBINING BREVE
U+043E	о	CYRILLIC SMALL LETTER O

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `uname` (часть пакета `uniutils`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = 'o' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Йо-йо

```
$ printf 'Йо-и\u0306о' | uname
```

U+0439	Й	CYRILLIC CAPITAL LETTER SHORT I
U+043E	о	CYRILLIC SMALL LETTER O
U+002D	-	HYPHEN-MINUS
U+0438	и	CYRILLIC SMALL LETTER I
U+0306	◌̆	COMBINING BREVE
U+043E	о	CYRILLIC SMALL LETTER O

Code points



- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '○' if U.category(c)[0] == 'M' else ''
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

Code points



- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '♂' if U.category(c)[0] == 'M' else '♀'
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```

U+1F468	♂	MAN
U+200D		ZERO WIDTH JOINER
U+1F469	♀	WOMAN
U+200D		ZERO WIDTH JOINER
U+1F467	♀	GIRL

Code points

- Кодировать “абстрактные символы”
- Целые числа от 0 до $10FFFF_{16}$
- Стандартно записываются в hex с префиксом U+
- Имеют уникальные имена

Инструменты

- `unicode` (часть пакета `unicodetools`)
- Модуль `unicodedata` в Python

```
import sys, unicodedata as U

for i, c in enumerate(sys.stdin.read()):
    s = '♂' if U.category(c)[0] == 'M' else '♀'
    print('U+{0:04X}\t{1}{2}\t{3}'
          .format(ord(c), s, c, U.name(c)))
```



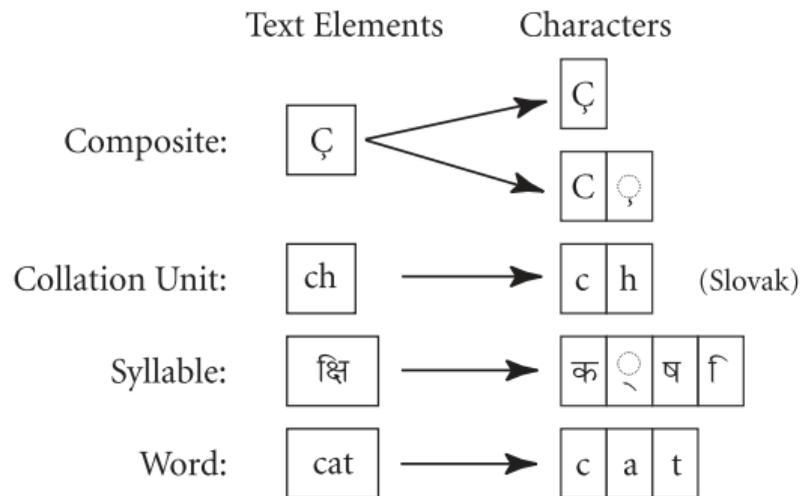
```
$ printf '\U1F468\u200D\U1F469\u200D\U1F467' \
| unicode
```

```
U+1F468 🧑 MAN
U+200D ZERO WIDTH JOINER
U+1F469 🧑 WOMAN
U+200D ZERO WIDTH JOINER
U+1F467 🧒 GIRL
```

Ключевые моменты

Ключевые моменты

- Текст разбивается на символы



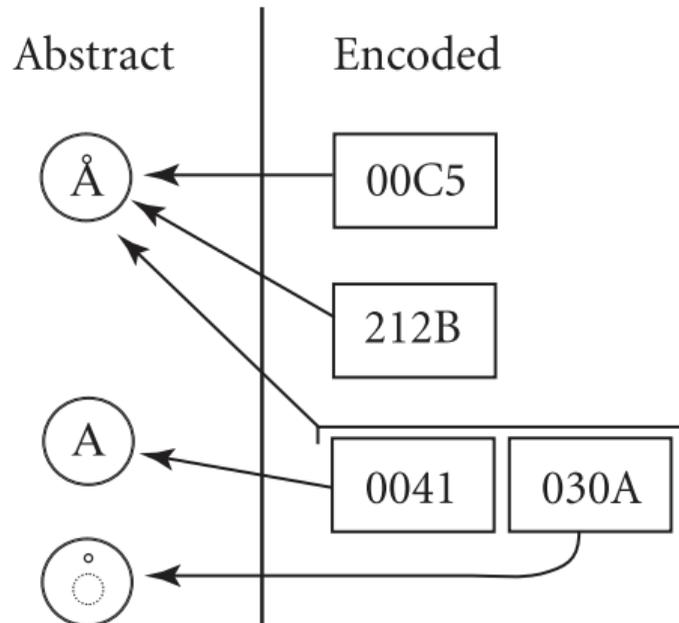
Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы

Glyphs	Unicode Characters
A Ā Á Â Ã Ä Å Æ Ā	U+0041 LATIN CAPITAL LETTER A
a ā á â ã ä å æ	U+0061 LATIN SMALL LETTER A
п n ū	U+043F CYRILLIC SMALL LETTER PE
ه هـ هـ هـ	U+0647 ARABIC LETTER HEH
fi fi	U+0066 LATIN SMALL LETTER F + U+0069 LATIN SMALL LETTER I

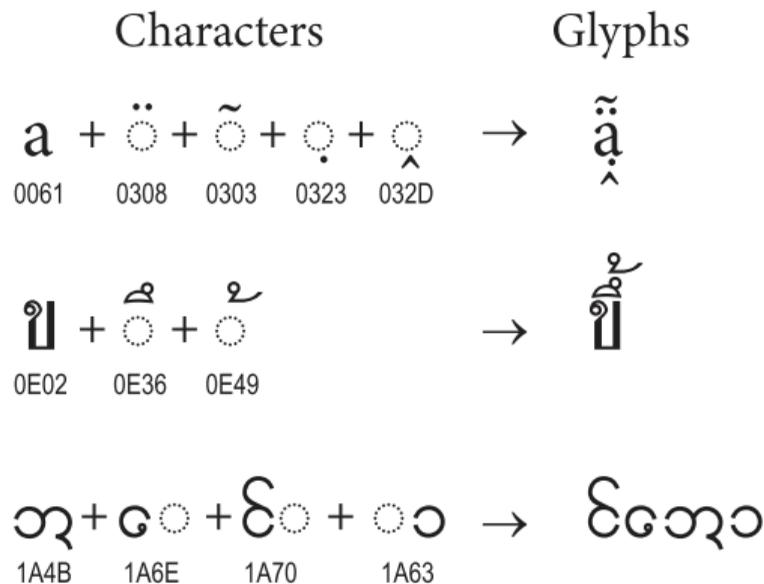
Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы
- Символы кодируются последовательностью code point'ов
 - Code point целое число от 0 до $10FFFF_{16}$ соответствующее некоторому абстрактному символу



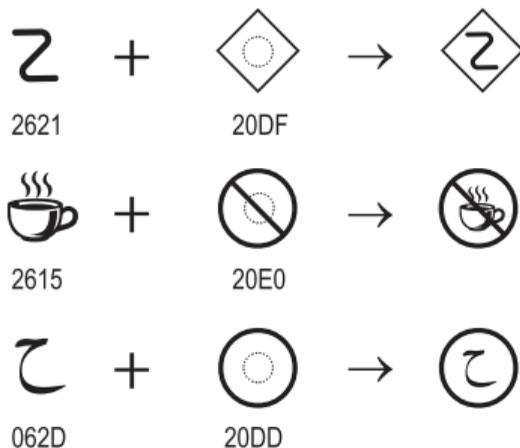
Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы
- Символы кодируются последовательностью code point'ов
 - Code point целое число от 0 до $10FFFF_{16}$ соответствующее некоторому абстрактному символу
- Символы могут комбинироваться



Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы
- Символы кодируются последовательностью code point'ов
 - Code point целое число от 0 до $10FFFF_{16}$ соответствующее некоторому абстрактному символу
- Символы могут комбинироваться



Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы
- Символы кодируются последовательностью code point'ов
 - Code point целое число от 0 до $10FFFF_{16}$ соответствующее некоторому абстрактному символу
- Символы могут комбинироваться
- Правила поиска, сортировки, эквивалентности и других процессов описаны в стандарте

$$\textcircled{1} \quad \underset{0042}{\text{B}} + \underset{00C4}{\text{Ä}} \equiv \underset{0042}{\text{B}} + \underset{0041}{\text{A}} + \underset{0308}{\text{ö}}$$

$$\textcircled{2} \quad \underset{01C7}{\text{LJ}} + \underset{0041}{\text{A}} \approx \underset{004C}{\text{L}} + \underset{004A}{\text{J}} + \underset{0041}{\text{A}}$$

$$\textcircled{3} \quad \underset{0032}{\text{2}} + \underset{00BC}{\frac{1}{4}} \approx \underset{0032}{\text{2}} + \underset{0031}{\text{1}} + \underset{2044}{\text{/}} + \underset{0034}{\text{4}}$$

Ключевые моменты

- Текст разбивается на символы
 - Символы \neq глифы
- Символы кодируются последовательностью code point'ов
 - Code point целое число от 0 до $10FFFF_{16}$ соответствующее некоторому абстрактному символу
- Символы могут комбинироваться
- Правила поиска, сортировки, эквивалентности и других процессов описаны в стандарте
- Три стандартных формата кодирования code point'ов: UTF-32, UTF-16, UTF-8
 - Code unit Минимальная единица кодирования в формате

A 00000041	Ω 000003A9	語 00008A9E	𐄀 00010384
---------------	---------------	---------------	---------------

UTF-32

A 0041	Ω 03A9	語 8A9E	𐄀 D800 DF84
-----------	-----------	-----------	------------------

UTF-16

A 41	Ω CE A9	語 E8 AA 9E	𐄀 F0 90 8E 84
---------	--------------	-------------------	------------------------

UTF-8

Unicode

Unicode Transformation Format

- **Code unit** Минимальная единица кодирования в формате
- **Scalar value** Кодированное значение code point'a в диапазонах U+0000..U+D7FF и U+E000..U+10FFFF
- **Символ \neq code point \neq scalar value \neq code unit**

A 00000041	Ω 000003A9	語 00008A9E	𐄀 00010384
---------------	---------------	---------------	---------------

UTF-32

A 0041	Ω 03A9	語 8A9E	𐄀 D800 DF84
-----------	-----------	-----------	------------------

UTF-16

A 41	Ω CE A9	語 E8 AA 9E	𐄀 F0 90 8E 84
---------	--------------	-------------------	------------------------

UTF-8

UTF-32

Code unit 32-битное значение

- Каждый скаляр кодируется одним 32-битным значением
- Кодировка фиксированной длины
- Большой размер
- Формат строк в Python 3

A 00000041	Ω 000003A9	語 00008A9E	卍 00010384
---------------	---------------	---------------	---------------

UTF-32

A 0041	Ω 03A9	語 8A9E	卍 D800 DF84
-----------	-----------	-----------	------------------

UTF-16

A 41	Ω CE A9	語 E8 AA 9E	卍 F0 90 8E 84
---------	--------------	-------------------	------------------------

UTF-8

UTF-16

Code unit 16-битное значение

- U+0000..U+D7FF и U+E000..U+FFFF кодируются одним 16-битным значением
- U+1000..U+10FFFF кодируются двумя 16-битными значениями в диапазоне U+D800..U+DFFF, называемыми **суррогатными парами**
- Символы большинства языков кодируются одним code unit'ом
- Кодировка переменной длины
- Формат строк в Python 2, Java и Windows API

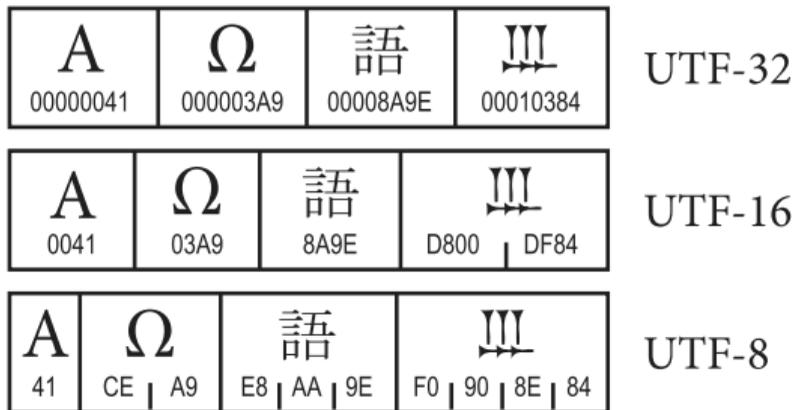


Table 3-5. UTF-16 Bit Distribution

Scalar Value	UTF-16
xxxxxxxxxxxxxxxx	xxxxxxxxxxxxxxxx
000uuuuuxxxxxxxxxxxxxxxx	110110wwwwwwxxxxxx 11011xxxxxxxxxxxx

Note: www = uuuu - 1

UTF-8

Code unit 8-битное значение

- Расширение ASCII
- По-байтовая кодировка
- **Компактный размер**
- **Кодировка переменной длины**
- Стандартный формат текста в Web (HTML, CSS, XML) и Unix системах

A 00000041	Ω 000003A9	語 00008A9E	Ⅲ 00010384	UTF-32
A 0041	Ω 03A9	語 8A9E	Ⅲ D800 DF84	UTF-16
A 41	Ω CE A9	語 E8 AA 9E	Ⅲ F0 90 8E 84	UTF-8

Table 3-6. UTF-8 Bit Distribution

Scalar Value	First Byte	Second Byte	Third Byte	Fourth Byte
00000000 0xxxxxxx	0xxxxxxx			
00000yyy yxxxxxxx	110yyyyy	10xxxxxx		
zzzzyyyy yxxxxxxx	1110zzzz	10yyyyyy	10xxxxxx	
000uuuuu zzzzyyyy yxxxxxxx	11110uuu	10uuzzzz	10yyyyyy	10xxxxxx

Byte order

- Многобайтовые кодировки требуют определения порядка байтов при записи code unit'a
- *Big-endian* — байты пишутся начиная с самого старшего
- *Little-endian* — байты пишутся начиная с самого младшего
- **Byte order mark (BOM)** U+FEFF в соответствующем формате

<table border="1"><tr><td>A</td><td>Ω</td><td>語</td><td>Ⅲ</td></tr><tr><td>00 00 00 41</td><td>00 00 03 A9</td><td>00 00 8A 9E</td><td>00 01 03 84</td></tr></table>	A	Ω	語	Ⅲ	00 00 00 41	00 00 03 A9	00 00 8A 9E	00 01 03 84	UTF-32BE
A	Ω	語	Ⅲ						
00 00 00 41	00 00 03 A9	00 00 8A 9E	00 01 03 84						
<table border="1"><tr><td>A</td><td>Ω</td><td>語</td><td>Ⅲ</td></tr><tr><td>41 00 00 00</td><td>A9 03 00 00</td><td>9E 8A 00 00</td><td>84 03 01 00</td></tr></table>	A	Ω	語	Ⅲ	41 00 00 00	A9 03 00 00	9E 8A 00 00	84 03 01 00	UTF-32LE
A	Ω	語	Ⅲ						
41 00 00 00	A9 03 00 00	9E 8A 00 00	84 03 01 00						
<table border="1"><tr><td>A</td><td>Ω</td><td>語</td><td>Ⅲ</td></tr><tr><td>00 41</td><td>03 A9</td><td>8A 9E</td><td>D8 00 DF 84</td></tr></table>	A	Ω	語	Ⅲ	00 41	03 A9	8A 9E	D8 00 DF 84	UTF-16BE
A	Ω	語	Ⅲ						
00 41	03 A9	8A 9E	D8 00 DF 84						
<table border="1"><tr><td>A</td><td>Ω</td><td>語</td><td>Ⅲ</td></tr><tr><td>41 00</td><td>A9 03</td><td>9E 8A</td><td>00 D8 84 DF</td></tr></table>	A	Ω	語	Ⅲ	41 00	A9 03	9E 8A	00 D8 84 DF	UTF-16LE
A	Ω	語	Ⅲ						
41 00	A9 03	9E 8A	00 D8 84 DF						
<table border="1"><tr><td>A</td><td>Ω</td><td>語</td><td>Ⅲ</td></tr><tr><td>41</td><td>CE A9</td><td>E8 AA 9E</td><td>F0 90 8E 84</td></tr></table>	A	Ω	語	Ⅲ	41	CE A9	E8 AA 9E	F0 90 8E 84	UTF-8
A	Ω	語	Ⅲ						
41	CE A9	E8 AA 9E	F0 90 8E 84						

Переменные окружения

- `LANG` Локаль по умолчанию
- `LC_COLLATE` Поиск и сравнение строк
- `LC_CTYPE` Диапазоны символов (алфавит, числа, верхний/нижний регистры)
- `LC_TIME` Формат даты и времени
- `LC_NUMERIC` Формат чисел
- `LC_MESSAGES` Язык сообщений системы и утилит
- `LC_ADDRESS` Формат адреса и локации
- ...
- `LC_ALL` Переопределяет все вышеперечисленные переменные

Переменные окружения

- **LANG** Локаль по умолчанию
- **LC_COLLATE** Поиск и сравнение строк
- **LC_CTYPE** Диапазоны символов (алфавит, числа, верхний/нижний регистры)
- **LC_TIME** Формат даты и времени
- **LC_NUMERIC** Формат чисел
- **LC_MESSAGES** Язык сообщений системы и утилит
- **LC_ADDRESS** Формат адреса и локации
- ...
- **LC_ALL** Переопределяет все вышеперечисленные переменные

Unix locale

language[_territory][.codeset][@modifier]

- **en_US** — американский английский
- **ru_RU.UTF-8** — русский (Россия)
- **fr_CA.ISO8859-1** — канадский французский
- ...

Примеры

```
$ export LANG=ru_RU.UTF-8
$ echo "windows" | \
  awk '{print toupper($0)}'
WINDOWS
$ export LANG=tr_TR.UTF-8
$ echo "windows" | \
  awk '{print toupper($0)}'
WINDOWS
```

Примеры

```
$ export LANG=ru_RU.UTF-8
$ echo "windows" | \
  awk '{print toupper($0)}'
WINDOWS
$ export LANG=tr_TR.UTF-8
$ echo "windows" | \
  awk '{print toupper($0)}'
WINDOWS
```

```
$ locale
LANG="en_US.UTF-8"
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="ru_RU.UTF-8"
LC_TIME="ru_RU.UTF-8"
...
LC_ALL=
$ locale -k LC_TIME | grep 'd_t_fmt'
d_t_fmt="%a %d %b %Y %T"
$ date
C6 26 окт 2024 00:02:42 +07
$ LC_TIME=en_US date
Sat Oct 26 12:02:42 AM +07 2024
$ LC_ALL=C date
Sat Oct 26 00:02:42 +07 2024
```

Q&A