

# Локализация и интернационализация программного обеспечения

## Инструментарий Современного Программиста

Иван Трепаков

NSU Sys.Pro

# ASCII

## 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

# Расширения ASCII

## ASCII

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

# Расширения ASCII

## ASCII

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

# Расширения ASCII

## ASCII

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

## CP866

80: АБВГДЕЖЗИЙКЛМНОП  
90: РСТУФХЦЧШЩЪЫЬЭЮЯ  
a0: абвгдежзийклмноп  
b0:  | | | | | | | | | | | | | | | |  
c0:  | | | | | | | | | | | | | | | |  
d0:  | | | | | | | | | | | | | | | |  
e0: рстуфхцчшщъыьэюя  
f0: ЁёЄєӢӓӮӹ••√№■




*MS-DOS*


# Расширения ASCII

ASCII

CP866

KOI8-R

00:	80: АБВГДЕЖЗИЙКЛМНОП
10:	90: РСТУФХЦЧШЩЪЫЬЭЮЯ
20: <code>!"#\$%&amp;'()*+,-./</code>	a0: абвгдежзийклмноп
30: <code>0123456789:;&lt;=&gt;?</code>	b0: 
40: <code>@ABCDEFGHIJKLMNO</code>	c0: 
50: <code>PQRSTUVWXYZ[\]^_</code>	d0: 
60: <code>`abcdefghijklmno</code>	e0: рстуфхцчшщъыьэюя
70: <code>pqrstuvwxyz{ }~</code>	f0: ЁёЄєӢӓӮӹ°•√№¤■

- | □ □ □ □ □ □ □ □  
 ( ■ • √ ≈ ≤ ≥ ∫ ° ² ÷  
= || ё ё ё ё ё ё ё ё  
|| | Ё | | | | | | | | ©  
ю аб ц де ф г х и й к л м н о  
п я р с т у ж в ь ы з ш э щ ч ь  
Ю А Б Ц Д Е Ф Г Х И Й К Л М Н О  
П Я Р С Т У Ж В Ы Ъ Ы З Ш Э Щ Ч Ъ

MS-DOS

Unix

## Расширения ASCII

ASCII		CP866		KOI8-R		CP1251	
00:		80:	АБВГДЕЖЗИЙКЛМНОП	- ▯▮▨▧▩▪▫▬▭	Ѓѓ,´,´´´´†‡€%љ<њќћџ		
10:		90:	РСТУФХЦЧШЩЪЫЬЭЮЯ	░░░░▦▧■√≈≤≥ ]°².÷	ђ‘’””•—™љ>њќћџ		
20:	!"#\$%&'()*+,-./	a0:	абвгдежзийклмноп	=  ғёгґтҗтҕллллҝҜ	ӱӳҝҐ!\$Ё©Є«¬-®İ		
30:	0123456789:;<=>?	b0:	░░░░ ↑↑↑↑↑↑↑↑↑↑	↑↑↑↑↑↑↑↑↑↑↑↑↑↑⊙	°±ııǣμ¶·ё№є»јѕѕї		
40:	@ABCDEFGHIJKLMNO	c0:	▬▯▯▯▯▯▯▯▯▯▯▯	юабцдефгхийклмно	АБВГДЕЖЗИЙКЛМНОП		
50:	PQRSTUVWXYZ[\]^_`	d0:	▬▯▯▯▯▯▯▯▯▯▯▯	пярстужвыызшэщчъ	РСТУФХЦЧШЩЪЫЬЭЮЯ		
60:	`abcdefghijklmno	e0:	рстуфхцчшщъыьэюя	ЮАБЦДЕФГХИЙКЛМНО	абвгдежзийклмноп		
70:	pqrstuvwxyz{ }~	f0:	ЁёЄєİіıŸŸ°••√№є■	ПЯРСТУЖВЫЫЗШЭЩЧЪ	рстуфхцчшщъыьэюя		

MS-DOS

# Unix

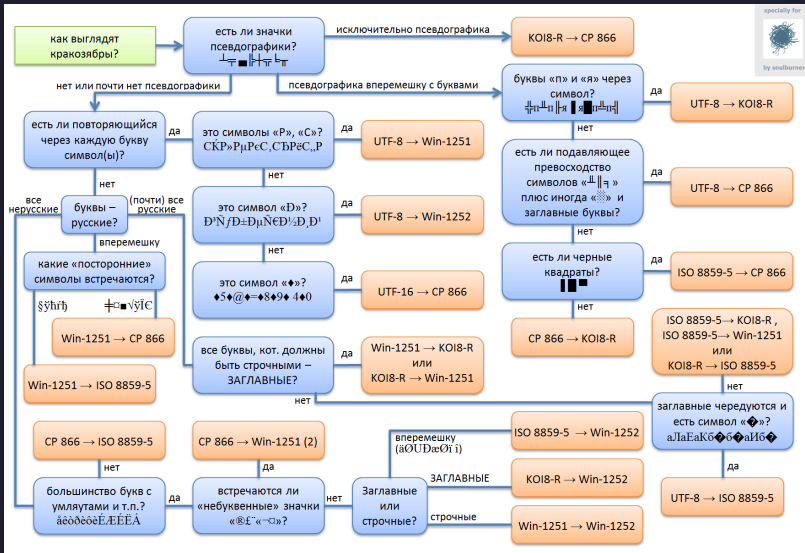
## Windows





оПХБЕР ЛХП!

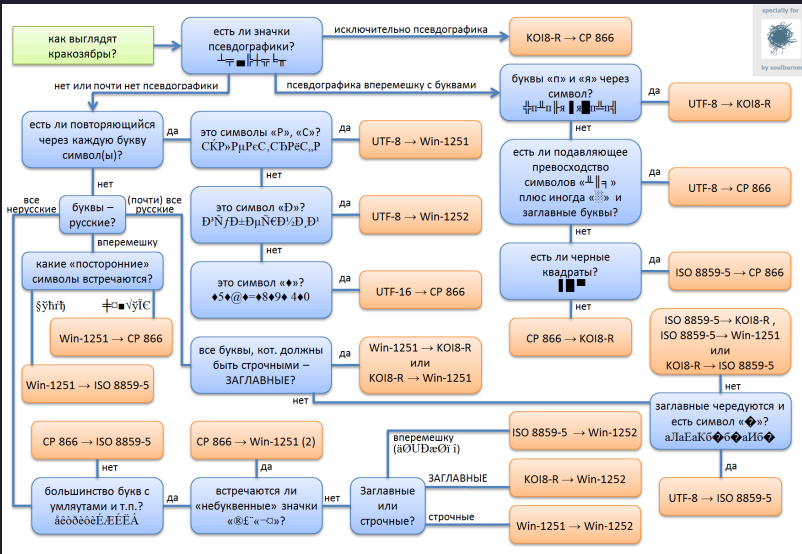
ОПХБЕР ЛХП!



# Крокозябры

ОПХБЕР ЛХП!

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

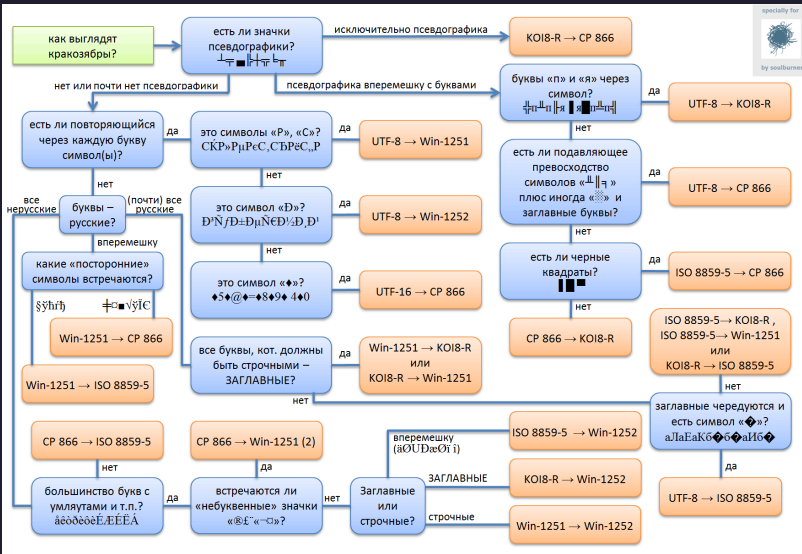


# Крокозябры

ОПХБЕР ЛХП!

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

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



# Крокозябры

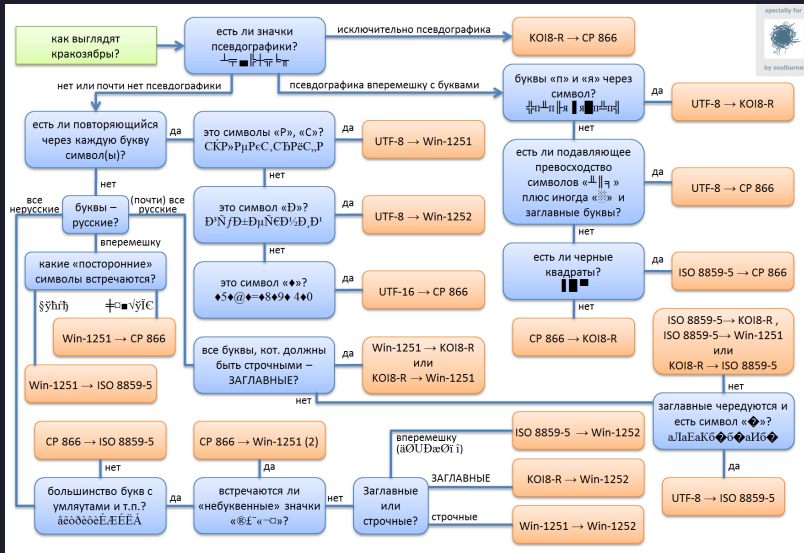
оПХБЕР ЛХП!

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

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

## Инструменты

- iconv
- encs
- ucharset



# Unicode

## 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



# Unicode

## Code points

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

# Unicode

## Code points

- Кодируют “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)))
```



# Unicode

## Code points

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

Hello 🌍!

## 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)))
```

# Unicode

## Code points

- Кодируют “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

# Unicode

## Code points

- Кодируют “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

# Unicode

## Code points

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

Привет 🌍!

## 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)))
```

# Unicode

## Code points

- Кодируют “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

# Unicode

## Code points

- Кодируют “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)))
```

Привет 🌍!

```
$ printf 'Привет \U1F30E!' | uname
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

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

Йо-йо

## 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)))
```

# Unicode

## Code points

- Кодируют “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)))
```

## 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



# Unicode

## Code points

- Кодируют “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)))
```

## 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

# Unicode

## Code points



- Кодируют “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)))
```

# Unicode

## Code points






- Кодируют “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)))
```

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

# Unicode

## Code points

- Кодируют “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)))
```



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

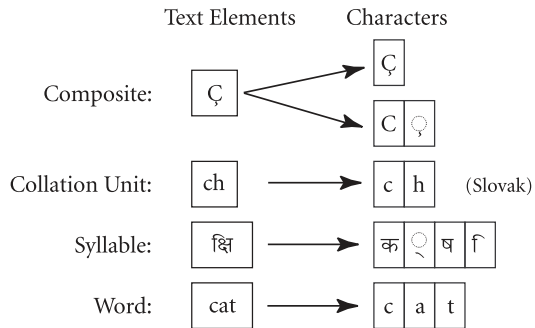
U+1F468		MAN
U+200D		ZERO WIDTH JOINER
U+1F469		WOMAN
U+200D		ZERO WIDTH JOINER
U+1F467		GIRL

## Ключевые моменты

# Unicode

## Ключевые моменты

- Текст разбивается на символы



# Unicode

## Ключевые моменты

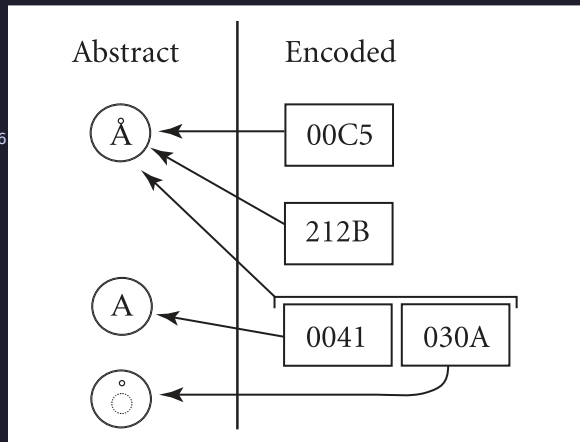
- Текст разбивается на символы
  - Символы  $\neq$  глифы

Glyphs	Unicode Characters
A Ā A A A A A A A	U+0041 LATIN CAPITAL LETTER A
a ā a a a a a a	U+0061 LATIN SMALL LETTER A
п n ū	U+043F CYRILLIC SMALL LETTER PE
ه ه ه ه	U+0647 ARABIC LETTER HEH
fi fi	U+0066 LATIN SMALL LETTER F + U+0069 LATIN SMALL LETTER I

# Unicode

## 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

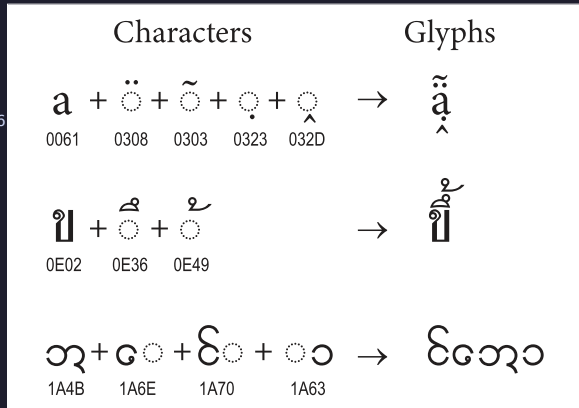




# Unicode

## Ключевые моменты

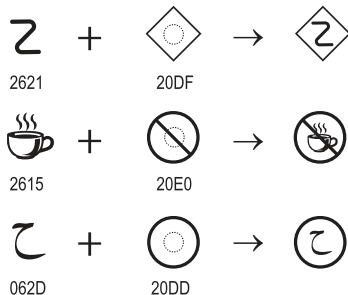
- Текст разбивается на символы
  - Символы  $\neq$  глифы
- Символы кодируются последовательностью code point'ов
  - Code point целое число от 0 до  $10FFFF_{16}$  соответствующее некоторому абстрактному символу
- Символы могут комбинироваться



# Unicode

## 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



# Unicode

## 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

$$\begin{array}{l} \textcircled{1} \quad \underset{0042}{B} + \underset{00C4}{\ddot{A}} \equiv \underset{0042}{B} + \underset{0041}{A} + \underset{0308}{\ddot{\circ}} \\ \textcircled{2} \quad \underset{01C7}{LJ} + \underset{0041}{A} \approx \underset{004C}{L} + \underset{004A}{J} + \underset{0041}{A} \\ \textcircled{3} \quad \underset{0032}{2} + \underset{00BC}{\frac{1}{4}} \approx \underset{0032}{2} + \underset{0031}{1} + \underset{2044}{/} + \underset{0034}{4} \end{array}$$

# Unicode

## 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

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

# Unicode

## UTF-32

*Code unit 32-битное значение*

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

UTF-32

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

UTF-16

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

UTF-8

# Unicode

## UTF-32

Code unit 32-битное значение

## UTF-16

Code unit 16-битное значение

Table 3-5. UTF-16 Bit Distribution

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

Note: www = uuuu - 1

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

UTF-32

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

UTF-16

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

UTF-8

# Unicode

## UTF-32

*Code unit 32-битное значение*

## UTF-16

*Code unit 16-битное значение*

**Table 3-5. UTF-16 Bit Distribution**

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

Note: www = uuuu - 1

## UTF-8

*Code unit 8-битное значение*

**Table 3-6. UTF-8 Bit Distribution**

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

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

UTF-32

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

UTF-16

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

UTF-8

# Unicode

## UTF-32

*Code unit 32-битное значение*

## UTF-16

*Code unit 16-битное значение*

**Table 3-5.** UTF-16 Bit Distribution

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

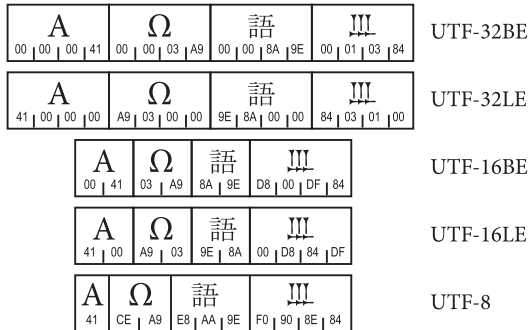
Note: www = uuuu - 1

## UTF-8

*Code unit 8-битное значение*

**Table 3-6.** UTF-8 Bit Distribution

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







## Переменные окружения

- LANG Локаль по умолчанию
- LC\_COLLATE Поиск и сравнение строк
- LC\_STYPE 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

# Локали

## 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