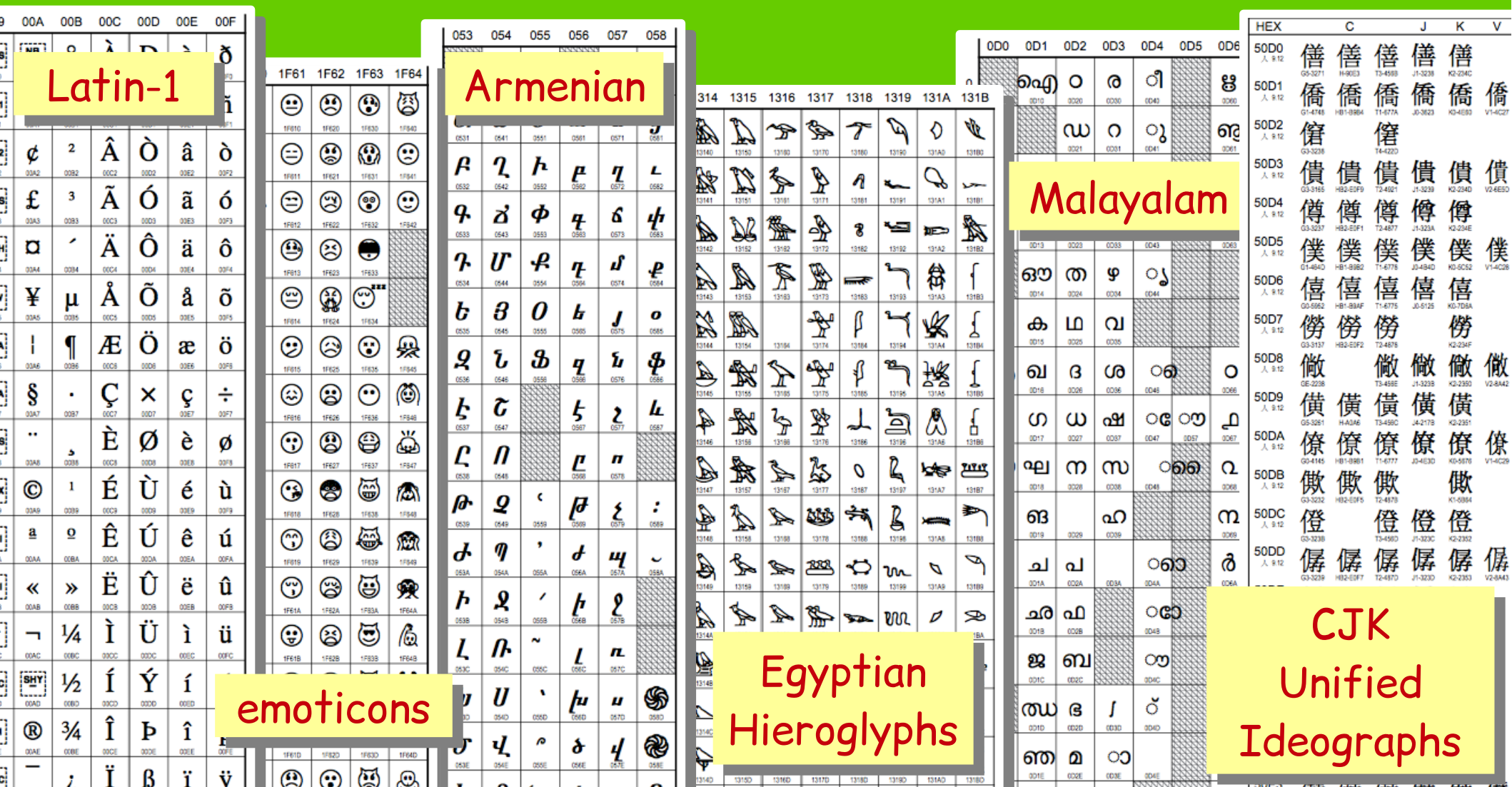
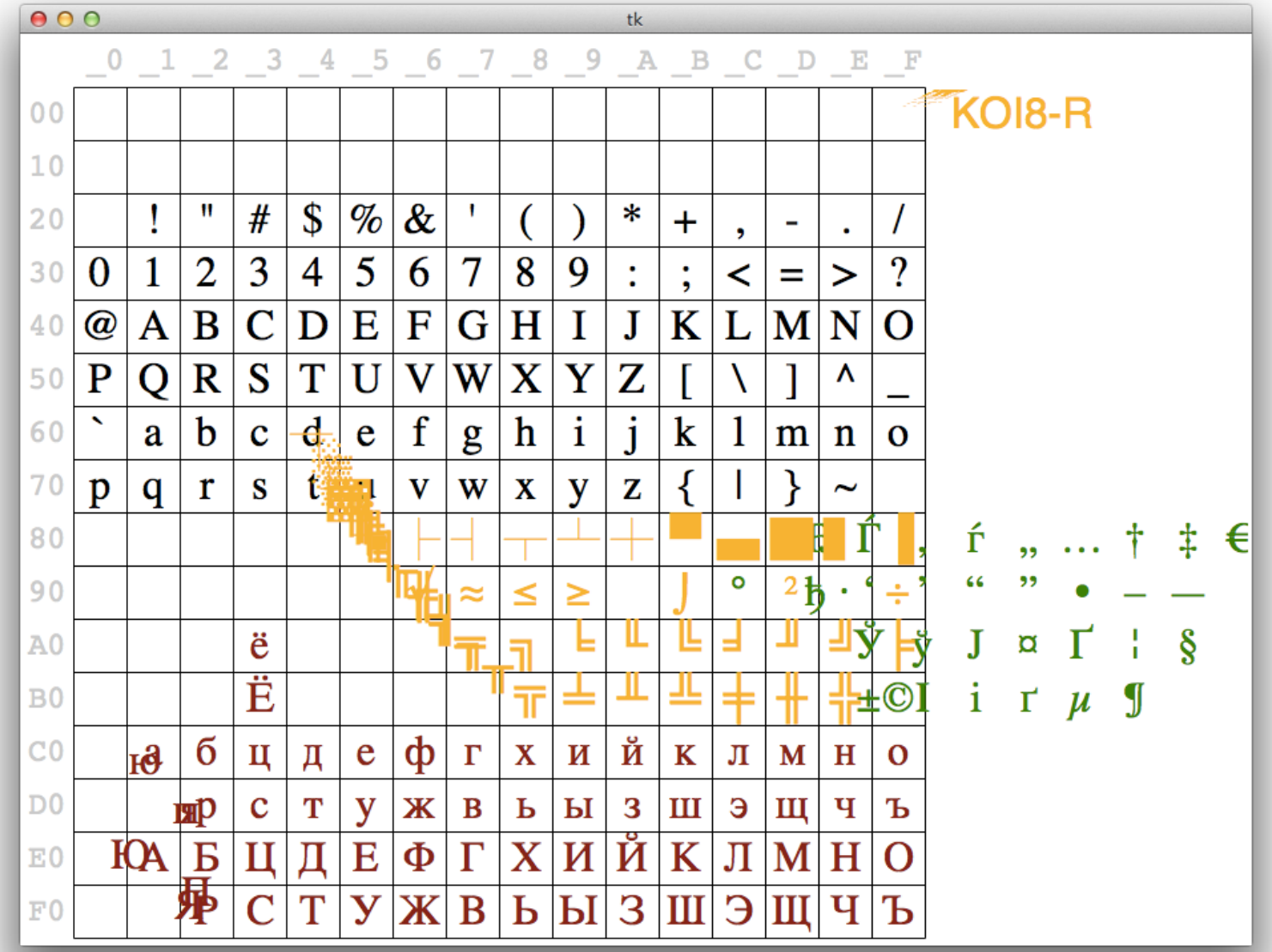
Unicode solutions in Python 2 and 3

Unicode solutions in Python 2 and 3



The single-byte codepage ballet



Video: https://www.youtube.com/watch?v=J4qioAacrYo
Source code: http://bit.ly/1Oqt0MZ

PythonPro

Why Unicode

- Too many incompatible byte encodings
- Separate concepts:
  - character identity: one **code point** for each abstract character
    - `U+0041 → LATIN CAPITAL LETTER A`
    - `U+096C → DEVANAGARI DIGIT SIX`
  - binary representation: multiple **encodings**
    - `U+0041 → 0x41 | 0x41 0x00`
    - `U+096C → 0xE0 0xA5 0xAC | 0x6C 0x09`

UTF-8

UTF-16LE

PythonPro

A sample of encodings

char.	code point	ascii	latin1	cp1252	cp437	gb2312	utf-8	utf-16le
A	U+0041	41	41	41	41	41	41	41 00
¿	U+00BF	*	BF	BF	A8	*	C2 BF	BF 00
Ã	U+00C3	*	C3	C3	*	*	C3 83	C3 00
á	U+00E1	*	E1	E1	A0	A8 A2	C3 A1	E1 00
Ω	U+03A9	*	*	*	EA	A6 B8	CE A9	A9 03
ۿ	U+06BF	*	*	*	*	*	DA BF	BF 06
“	U+201C	*	*	93	*	A1 B0	E2 80 9C	1C 20
€	U+20AC	*	*	80	*	*	E2 82 AC	AC 20
┌	U+250C	*	*	*	DA	A9 B0	E2 94 8C	0C 25
气	U+6C14	*	*	*	*	C6 F8	E6 B0 94	14 6C
氣	U+6C23	*	*	*	*	*	E6 B0 A3	23 6C
𝄞	U+1D11E	*	*	*	*	*	F0 9D 84 9E	34 D8 1E DD

Figure 4-1 of Fluent Python

PythonPro

.encode() vs .decode()

- “Humans use text. Computers speak bytes.”
  - Esther Nam and Travis Fischer in *Character encoding and Unicode in Python (Pycon US 2014)*

- Use .encode() to convert **human** text to **bytes**
- Use .decode() to convert **bytes** to **human** text

2.7 gotcha:
the methods
.encode() and .decode()
exist in **str** and **unicode**

PythonPro

Data types for text and bytes

	Python 2.7	Python 3.4
Human text	**unicode** u'café', u'caf\xe9'	**str** 'café', u'café'
Bytes (imutável)	**str** 'café', 'caf\xe9', b'café'...	**bytes** b'caf\xc3\xa9'
Bytes (mutável)	**bytearray** bytearray(b'caf\xc3\xa9')	**bytearray** bytearray(b'caf\xc3\xa9')

Best practice

The Unicode sandwich

bytes → str Decode bytes on input,

100% str process text only,

str → bytes encode text on output.

Figure 4-2 of Fluent Python

PythonPro

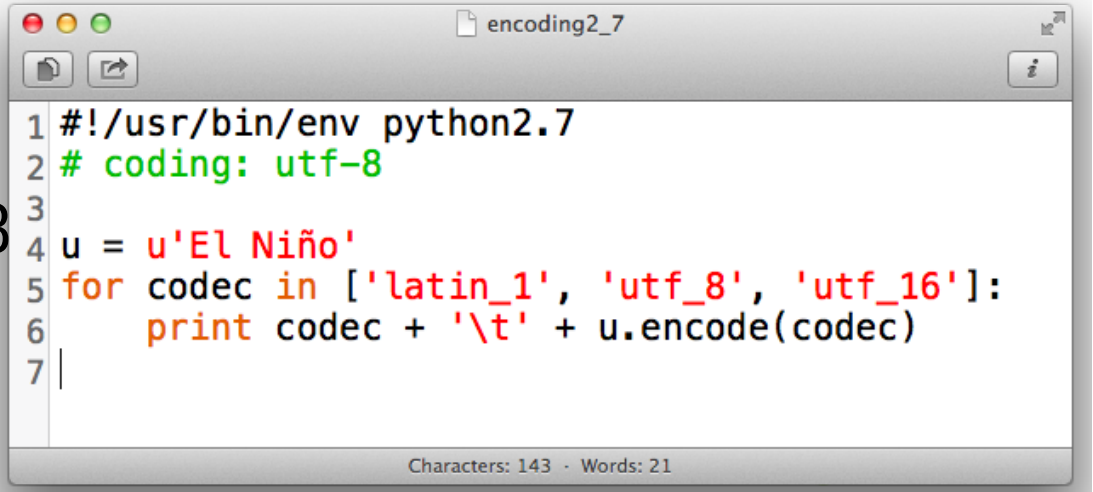
Coping with Unicode Errors

- SyntaxError
  - A .py file is loaded with contents in an unexpected encoding
- UnicodeDecodeError
  - A binary sequence is contains bytes that are not valid in the expected encoding
- UnicodeEncodeError
  - A Unicode string contains codepoints that have no representation in the desired encoding

Coping with SyntaxError

- A .py file is loaded with contents in an unexpected encoding
  - The source file encoding is not the default, and no # encoding comment was found.
  - The source file encoding is not the one declared in the # encoding comment
- Defaults:
  - Python 2.7 == ASCII
  - == UTF-8

2.7 gotcha: default source encoding is ASCII



Unicode database

5. bash

```
$ python3 numerics_demo.py
U+0031    1     re_dig  isdig   isnum     1.00   DIGIT ONE
U+00bc    ¼     -       -       isnum     0.25   VULGAR FRACTION ONE QUARTER
U+00b2    ²     -       isdig   isnum     2.00   SUPERSCRIPT TWO
U+0969    ३     re_dig  isdig   isnum     3.00   DEVANAGARI DIGIT THREE
U+136b    ፫     -       isdig   isnum     3.00   ETHIOPIC DIGIT THREE
U+216b    XII   -       -       isnum    12.00   ROMAN NUMERAL TWELVE
U+2466    ⑦     -       isdig   isnum     7.00   CIRCLED DIGIT SEVEN
U+2480    ⒀     -       -       isnum    13.00   PARENTHESIZED NUMBER THIRTEEN
U+3285    ㊅     -       -       isnum     6.00   CIRCLED IDEOGRAPH SIX
$
```

PythonPro

Unicode database

5. bash

```
$ python3 numerics_demo.py
U+0031    1     re_dig  isdig   isnum     1.00   DIGIT ONE
U+00bc    ¼     -       -       isnum     0.25   VULGAR FRACTION ONE QUARTER
U+00b2    ²     -       isdig   isnum     2.00   SUPERSCRIPT TWO
U+0969    ३     re_dig  isdig   isnum     3.00   DEVANAGARI DIGIT THREE
U+136b    ፫     -       isdig   isnum     3.00   ETHIOPIC DIGIT THREE
U+216b    XII   -       -       isnum    12.00   ROMAN NUMERAL TWELVE
U+2466    ⑦     -       isdig   isnum     7.00   CIRCLED DIGIT SEVEN
U+2480    ⒀     -       -       isnum    13.00   PARENTHESIZED NUMBER THIRTEEN
U+3285    ㊅     -       -       isnum     6.00   CIRCLED IDEOGRAPH SIX
$
```

numerics_demo.py — Edited

```
1  import unicodedata
2  import re
3
4  re_digit = re.compile(r'\d')
5
6  sample = '1\xbc\xb2\u0969\u136b\u216b\u2466\u2480\u3285'
7
8  for char in sample:
9      print('U+%04x' % ord(char),                          # <A>
10           char.center(6),                                # <B>
11           're_dig' if re_digit.match(char) else '-',     # <C>
12           'isdig' if char.isdigit() else '-',            # <D>
13           'isnum' if char.isnumeric() else '-',          # <E>
14           format(unicodedata.numeric(char), '5.2f'),     # <F>
15           unicodedata.name(char),                        # <G>
16           sep='\t')
17
```

Characters: 578 · Words: 57