MTAT.03.311
Loomuliku keele töötlus *Python*is
(6 EAP)

Karl-Oskar Masing

08.09.2015

Tehniline eeltöötlus

Kodeering

Kodeering - ajalugu

- Arvuti talletab informatsiooni bittidena
- Vajadus informatsiooni kuvamiseks ja vahetamiseks
- ASCII (1968)
  - American Standard Code for Information Interchange
  - Seab sümboli arvudele 0...127 (7 bitised arvud)

Kodeering - ajalugu

- 8-bitiste arvutite tulekuga 80ndatel jäi viimane bit vabaks
- Sündisid kohalikud dialektid, mis seadsid 8ndale bitile oma krõnksud
- Ajaga tekkisid *de facto* standardid, kuidas arve 128-255 kujutada (*extended ASCII*)
  - ISO-8859-1 (latin1)
  - Windows-1252 (rahvasuus ANSI, vale termin)

Kodeering - ajalugu

- 255 sümbolit vähe
  - mitmekeelsed tekstid
  - idamaised keeled
- 16-bitise Unicode’i sünd (1987)
  - eesmärgiks ühendada kõik kirjalikud keeled
  - aluseks latin1
    - 255 esimest bitijada ühtivad
  - 65 536 sümbolit

Unicode

- Töötab abstraktsetel koodipunktidel (code point)
- Koodipunktile on seatud vastavusse sümbol
- Võib mõelda kui sümboli tasemel mõistest
- Sümboli graafilise kujutuse defineerib font ja GUI rakendused

Unicode

- “õ”-le vastava koodipunkti tähis
  - U+00F5
- Unicode’i sõne koosneb koodipunktide jadast

  >>> u”ilus”
- Koodipunktide jada tuleb mälus kujutada baitidena

Unicode

- Reeglite kogumikku, mille abil Unicode'i sõne konverteeritakse baitide jadaks, nimetatakse kodeeringuks
- Python'i vaikekodeering on ASCII
- Terminali kodeeringud saab käsuga

```
>>> sys.stdout.encoding
>>> sys.stdin.encoding
```

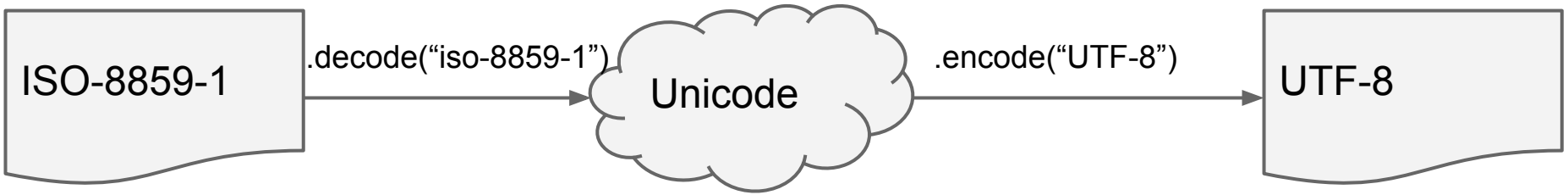
Unicode

- Kodeeringust Unicode’i sõneks saame vastava kodeeringu abil dekodeerides

- Unicode’i sõnest kodeeringuks saame vastava kodeeringu abil kodeerides

Praktika

- Kodeeringust A kodeeringusse B saame läbi Unicode'i



Õige kodeeringuga dekodeerimine töötab alati, sest Unicode on "kõigi tähestike ülemhulk"

Ei tööta, kui kodeering ei toeta vastavat sümbolit. Unicode'is on "kõik maailma sümbolid"

Praktika - dekodeerimine ei tööta alati

```
>>> sys.stdin.encoding
'UTF-8'
>>> "jääääär"
'j\xc3\xa4\xc3\xa4\xc3\xa4\xc3\xa4r'
>>> unicode("jääääär")  # kasutab vaikedekodeeringut ega suuda UTF-i dekodeerida
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in position 1: ordinal not in range(128)
```

Praktika - kodeerimine ei tööta alati

```
>>> sys.stdin.encoding
'UTF-8'
>>> "jääääär".decode("utf-8")
u'j\xe4\xe4\xe4\xe4r'
>>> "jääääär".decode("utf-8").encode("ascii")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
UnicodeEncodeError: 'ascii' codec can't encode characters in position 1-4:
ordinal not in range(128)
```

Praktika - vahel töötab ka

```
>>> "jahimees".decode("utf-8")
u'jahimees'
>>> "jahimees".decode("utf-8").encode("ascii")
'jahimees'
>>> "jahimees".decode("utf-8").encode("ascii").decode("utf-8")
u'jahimees'
```

Praktika - failidega töötamine

```
>>> import codecs
>>> with codecs.open(“utf8.fail”,’r’,encoding=”utf-8”) as fin:
…       fin.readline()  # tagastab utf-8’ga dekodeeritud rea

>>> with codecs.open(“utf8.fail”,’w’,encoding=”utf-8”) as fout:
…       fin.write(“tere”)  # kirjutab faili utf-8 kodeeringus
```

Praktika - hea tava

- Mitte-ingliskeelsete dokumentidega töötades alati asjatada maksimaalselt Unicode’i tasemel
  - vastasel korral tahab mõni funktsioon varem või hiljem kodeerida-dekodeerida ASCII-d

Mooduli kodeering

- Pythoni .py failis saab defineerida kodeeringu, mida interpretaator ASCII asemel kasutab
  - saab kasutada täpitähti kommentaarides (mittesoovitatav)

```
# -*- coding: utf-8 -*-
from collections import defaultdict  # impordin sõnaraamatu
```

Indekseerimine

Pipeline’id

Tüütud andmeformaadid

XML

- Võimaldab talletada
  - märgendatud teksti (struktureeritud)
  - informatsiooni kaota, eeldusel, et suudame pärast parsida
- Pythonil on töötamiseks standardteek *xml*

XML

```
import xml.etree.ElementTree as ET


tree = ET.parse(“my_xml.xml”)
root = tree.getroot()
first_div_text = tree.find
(“{xmlns_domain_address}div”).text
```

PDF

- Pythonil puudub vaike PDF-i parsija
- Andmete kättesaamine ning kvaliteet on ebausaldusväärsed
  - mõistlikum püüda andmeid kätte saada PDF-i eelsest formaadist

PyPDF2

```
from PyPDF2 import PdfFileReader

with open(file_name,’r’) as fin:
        reader = PdfFileReader(fin)
        for page_num in range(reader.getNumPages()):
                text = reader.getPage(page_num).extractText()
                do_something_with(text)
```

Excel

Zip

DOCX

Andmete teisendamine

Zippimine