Informatika pro moderní fyziky (8) web scraping, procvičení ERb+LaTeX, tvorba vektorové grafiky

František HAVLŮJ

e-mail: haf@ujv.cz

ÚJV Řež
oddělení Reaktorové fyziky a podpory palivového cyklu

akademický rok 2015/2016
23. listopadu 2016

Obsah

Zpracování cizích zdrojů na webu – web scraping

- dosud jsme zpracovávali pouze lokální, hezky formátovaná data
- i leckteré externí služby poskytují pěkná API ve formátech JSON nebo XML
- ale leckdy taky ne a zajímavé informace
- protože jsme chytré horákyně, naučíme se, jak data automaticky získat

Zadání úkolu

- protože po práci si chceme oddychnout a netrápit se přitom náročnou intelektuální činností, přečteme si rádi dobrý komiks
- a protože jsme přiměřeně cyničtí a také si chceme pocvičit angličtinu, přečteme si redmeat
- www.xkcd.com
- .. ale nechceme klikat a nechceme číst na internetu, takže si zhotovíme PDFko se všemi díly najednou
- zvládneme to snadno v LaTeXu, ale potřebujeme postahovat ty obrázky

HTML,

- http://xkcd.com/1000
- zobrazíme zdrojový kód
- HTML: tagy, atributy, `class`, `id`
- naštěstí to první zajímavé umíme vykoukat bez znalostí... stačí regex

Stahování webové stránky

Součást standardní knihovny – `open-uri`

```
require 'open-uri'
f = open(remote_url)
s = f.read
m = s.match(...)
```

- najdu si url obrázku
- někam si ho uložím
- budu chytrý ohledně jména souboru
- dám bacha na příponu!

Stahování dat

```
require 'open-uri'
File.open(local_filename, 'wb') do |f2|
  open(remote_url, 'rb') do |f1|
    f2.write f1.read
  end
end
```

(kdo chce mít lepší život, tak si samozřejmě nadefinuje funkci!)

Vygenerovat PDF

- zase triviální, už to umíme, rychlá akce na deset minut!
- ERb šablona, použít `erb_compiler` (mám z minula)
- seznam souborů vzít z `Dir["comics_*"]`
- všechno známe z minula

Další krok: popisky, nadpis a transkript

- u každého komiksu je popisek – atribut `title` u `img` tagu
- správný selektor je `#comics img`
- nainstalujeme gem `nokogiri`

```
f = open("http://xkcd.com/1000")
s = f.read
doc = Nokogiri::HTML(s)
img = doc.css("#comics img").first
puts img.attributes["title"]
```

- je tam navíc název a transkript
- je chytré to uložit chytře! takže ne 3 pole `titles`, `comments`, `transcripts` ale hezky jedno pole hashů
- zde přijdou ke slovu symboly: je to lepší klíč než řetězec
- `{:title => "...", :comment => "..."}`
- alternativní zápis
- `{title: "...", comment: "..."}`
- a opět poskládám v PDF

Obsah

Zadání dnešní úlohy

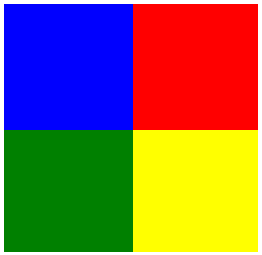
- pro zadanou textovou mapu AZ VR1 potřebuju udělat hezký obrázek
- co druh, to barvička, rozumně zacházet s odstíny (palivo různě modré, R/B/E tyče různě červené, zelené, fialové)

Jak na obrázky

- pěkný formát na tvorbu vektorových obrázku je SVG (Scalable Vector Graphics)
- je to dobrá věc především na internet – všechny prohlížeče ho umí
- stejně jako HTML je postaven na XML

Jednoduchý příklad

```
<svg width="320" height="320" xmlns="http://www.w3.org/2000/svg" version="1.1">
  <rect x="0.0" y="0.0" width="40.0" height="40.0" fill="blue" />
  <rect x="40.0" y="0.0" width="40.0" height="40.0" fill="red" />
  <rect x="0.0" y="40.0" width="40.0" height="40.0" fill="green" />
  <rect x="40.0" y="40.0" width="40.0" height="40.0" fill="yellow" />
</svg>
```



SVG – co a jak

- souřadný systém z levého horního rohu
- je potřeba udat celkovou šířku a výšku
- zatím nám stačí obdélník – tag `rect`
- pozor, je to striktní XML, tedy je nutné `rect` tag uzavřít (!)
- vyzkoušejte – nejdřív jen tak, potom vygenerovat 8x8 mapu (zatím klidně prázdnou)

Další SVG chytrosti

- kromě `rect` se bude hodit také `text`
- jako text se zobrazí obsah příslušného elementu
- opět použiju atributy `x`, `y` (levý dolní roh) a můžu přihodit `text-anchor="middle"`, aby to byl dolní prostředek

Postup

- načtu ze souboru třeba do 2D pole
- budu mít hash s barvičkama
- vykreslím do SVG

A to je vše, přátelé!