Informatika pro moderní fyziky (9)
ERb šablony. Regulární výrazy.

František HAVLŮJ

e-mail: haf@ujv.cz

ÚJV Řež
oddělení Reaktorové fyziky a podpory palivového cyklu

akademický rok 2022/2023
30. listopadu 2022

Obsah

- sestavení dokumentu v LaTeXu – výroba PDF v praxi
- použití ERb šablon (úvod)

Obsah

Úkoly

- najít všechna celá čísla: jaké je nejmenší a největší číslo a kolik jich je
- najít všechny emailové adresy: vypište je do souboru seskupené podle domény
- vlnka: nedělitelné mezery v textu před k, s, v, z (výsledek vypište do souboru)
- nahradit desetinné tečky čárkami (výsledek vypište do souboru)

Jak na to: regulární výrazy (regexy)

- můžu definovat chytrou ‘masku’
- nejjednodušší: /text/ - text
- skupiny písmen: /1[abc]/ - 1a, 1b ...
- speciální skupiny: `/\\da/` - 1a, 2a, 3a ...
- opakování: +, *
- vytažení části výrazu do tzv. capture group – stačí uzávorkovat
- hračka a vysvětlení: rubular.com

Regexy v Ruby

- `String#match` vrací MatchData objekt, m[0] je ten samotný řetězec, m[i] jsou skupiny
- `String#gsub` nahrazuje; můžu použít s blokem, nicméně tam nemám capture groups (ale můžu použít `Regexp.last_match`)
- pokud použiju formu bez bloku, mám capture groups v \\1 a \\2
- `String#scan` vrací pole všech výskytů

Obsah

Co se nám nelíbí na generování dokumentu

- je to hrozně roztahané uvnitř zdrojáku
- je to dost nepřehledné
- nevidíme strukturu texového dokumentu
- představte si složitější dokument...

ERb (Embedded Ruby)

- lepší šablona - “aktivní text”
- používá se například ve webových aplikacích
- hodí se ale i na generování latexových dokumentů, resp. všude, kde nám nesejde na whitespace
- poměrně jednoduchá syntaxe, zvládne skoro všechno

Základní syntaxe ERb (1)

Jakýkoli Ruby příkaz, přiřazení, výpočet ...

```
<% a = b + 5 %>
<% list = ary * ", " %>
```

Základní syntaxe ERb (2)

Pokud chci něco vložit, stačí přidat rovnítko

```
<%= a %>
<%= ary[1] %>
<%= b + 5 %>
```

Základní syntaxe ERb (3)

Radost je možnost použít bloky a tedy i iterátory apod. v propojení s vkládaným textem:

```
<% (1..5).each do |i| %>
Number <%= i %>
<% end %>
<% ary.each do |x| %>
Array contains <%= x %>
<% end %>
```

Důležité upozornění

- oddělení modelu a view
- přestože lze provádět zpracování dat a výpočty přímo v ERb, je to nejvíc nejhorší nápad
- je chytré si všechno připravit v modelu (tj. v Ruby skriptu, kterým data chystáme)
- a kód ve view (tj. v ERb šabloně) omezit na naprosté minimum

Jak ze šablony udělat výsledek

Příklad překladu ERb

```
require "erb_compiler"

erb(template, filename, {:x => 1, :y => 2})
```

třetí parametr je hash, který nám vlastně definuje proměnné dostupné uvnitř šablony při překladu

Ukázka v latexu

```
\subsection{Koncentrace kyseliny borité}

<% cycles.each do |i| %>

\subsubsection{Kampaň <%= i %>}

\begin{center}
  \includegraphics[width=0.8\textwidth]{bc_<%= "%02d" % i %>_bc.eps}
\end{center}

<% end %>
```

Dva úkoly na začátek

1. pomocí ERb vyrobte v LaTeXu malou násobilku do tabulky
2. s jedinou šablonou vyrobte 3 dokumenty s násobilkou 7x7, 9x9, 11x11 (musíte parametrizovat)
3. (minule jsme dělali čistě textovou)

Obsah

Úkol na dnešek

- přepracovat náš krásný report z JE Třeskoprsky s použitím ERb
- tabulky nicméně zajisté předělávat nebudeme – použijeme hotové

Co je podstatné

- nezapomenout, kdy jsem v Ruby, kdy v ERb, kdy v latexu a vždy dodržet zadanou syntaxi
- mít na paměti, jak dostat data ze skriptu do šablony
- potřebuju mít načtený seznam všech kampaní do nějakého pole a to poslat do šablony

Obsah

Zadání dnešní úlohy

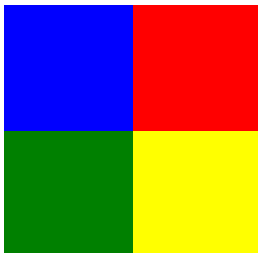
- pro zadanou textovou mapu AZ VR1 potřebuju udělat hezký obrázek
- co druh, to barvička, rozumně zacházet s odstíny (palivo různě modré, R/B/E tyče různě červené, zelené, fialové)

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!