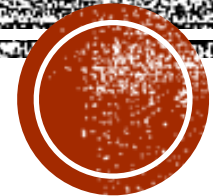


CONSTRUCTING SEARCHES

Introduction to Regular Expressions

(MENA Version)



MAXIM ROMANOV

To start: Regular Expressions Practicum File
on *GitHub*: [https://github.com/
maximromanov/re_tutorial](https://github.com/maximromanov/re_tutorial)

- **On Windows or Mac**

- Install Sublime Text
- Open the practicum file in the editor



What are Regular Expressions?

- very small language for describing textual patterns
- not a programming language, yet a part of each one
- incredibly powerful tool for find/replace operations
- old (1950s-60s)
- arcane art
- ubiquitous



Why Use Regular Expressions?

To search:

- all spelling variations of the same word:
 - Col. Qadhdhafi or Kaddafi? Isfahan or Işbahān?
- words of specific morphological patterns:
 - *[root]er, [root]ed, [root]ing [root]s*: all derivatives from the same root
- entities that may be referred to differently:
 - references to Austria?
 - references to education in biographies

To search and replace:

- reformat “dirty”/inconsistent data

To tag:

- make texts navigable and more readable
- tag information relevant to your research

•

and many other uses...



The Basics

- a **regular expression** is a pattern enclosed within delimiters
 - delimiters will differ depending on a programming language or software that you use; you may also not see them at all
 - most text editors that support RE do not display delimiters (*EditPad Pro*, *Sublime Text*, *TextMate*)
- most characters match themselves
- there are also special characters

Example:

- `/Hamburg/` is a regular expression that matches “Hamburg”
 - *slash* is the delimiter enclosing the expression
 - “Hamburg” is the pattern



/at/

- Matches strings with “a” followed by “t”.

at

hat

that

atlas

aft

Athens



/at/

- Matches strings with “a” followed by “t”.

at

hat

that

atlas

aft

Athens



Characters & Special Characters

- most characters match themselves
- matching is case sensitive
- special characters: `()^${}[]\|. +?*`
- to match a special character in your text, you need to “escape it”, i.e. precede it with “\” in your pattern:
 - `/Bağdād [sic]/`
does not match “**Bağdād [sic]**”
 - `/Bağdād \[sic\]/`
matches “**Bağdād [sic]**”



Character Classes: `[]`

- Characters within `[]` are choices for a single-character match.
- Think of a type of ***or***.
- Order within `[]` is unimportant.
- `/x[01]/` matches `>>> "x0"` and `"x1"`.
- `/[10][23]/` matches `>>>`
`>>> "02", "03", "12" and "13"`.
- Initial `^` negates the class:
 - `/[^45]/` matches any character except 4 or 5.



`/[ch]at/`

- Matches strings with “c” or “h”, followed by “a”, followed by “t”.

chat

cat

fat

phat



`/[ch]at/`

- Matches strings with “c” or “h”, followed by “a”, followed by “t”.

that

at

chat

cat

fat

phat



Ranges (within classes)

- Ranges define sets of characters within a class.
 - `/[1-9]/` matches any non-zero digit
 - `/[a-zA-Z]/` matches any letter of the English alphabet
 - `/[12][0-9]/` matches numbers between 10 and 29



Ranges shortcuts

Shortcut	Name	Equivalent Class
<code>\d</code>	digit	<code>[0-9]</code>
<code>\D</code>	not digit	<code>[^0-9]</code>
<code>\w</code>	word	<code>[a-zA-Z0-9_]</code> (<i>actually more!</i>)
<code>\W</code>	not word	<code>[^a-zA-Z0-9_]</code>
<code>\s</code>	space	<code>[\t\n\r\f\v]</code>
<code>\S</code>	not space	<code>[^\t\n\r\f\v]</code>
<code>.</code>	everything	<code>[^\n]</code> (depends on mode)



`/\d\d\d[-]\d\d\d\d/`

- Matches strings with: 501-1234 234 1252

- Three digits
- Space or dash
- Four digits

652.2648 713-342-7452

PE6-5000 653-6464x256



/\d\d\d[-]\d\d\d\d/

- Matches strings with:

- Three digits
- Space or dash
- Four digits

501-1234

234 1252

652.2648

713-342-7452

PE6-5000

653-6464x256



Repeaters

- Symbols indicating that the preceding element of the pattern can repeat.
- **/runs?/** matches *runs* or *run*
- **/1\d*/** matches any number beginning with “1”.

Repeater	Count
?	zero or one
+	one or more
*	zero or more
{ <i>n</i> }	exactly <i>n</i>
{ <i>n</i> , <i>m</i> }	between <i>n</i> and <i>m</i> times
{ , <i>m</i> }	no more than <i>m</i> times
{ <i>n</i> , }	at least <i>n</i> times



Repeaters

Strings:

1: "at" 2: "art"
3: "arrrrrt" 4: "aft"

Patterns:

A: /ar?t/ B: /a[fr]?t/
C: /ar*t/ D: /ar+t/
E: /a.*t/ F: /a.+t/

Repeater

Count

?

zero or one

+

one or more

*

zero or more

{ *n* }

exactly *n*

{ *n* , *m* }

between *n* and *m*
times

{ , *m* }

no more than *m*
times

{ *n* , }

at least *n* times



Repeaters

1: `"at"` 2: `"art"`
3: `"arrrrt"` 4: `"aft"`

- `/ar?t/` matches "at" and "art" but not "arrrt".
- `/a[fr]?t/` matches "at", "art", and "aft".
- `/ar*t/` matches "at", "art", and "arrrrt"
- `/ar+t/` matches "art" and "arrrt" but not "at".
- `/a.*t/` matches anything with an 'a' eventually followed by a 't'.



Lab: Intro (in the practicum file)

Repeater	Count	Shortcut	Name
?	zero or one	\d	digit
+	one or more	\D	not digit
*	zero or more	\w	word
{ <i>n</i> }	exactly <i>n</i> times	\W	not word
{ <i>n</i> , <i>m</i> }	between <i>n</i> and <i>m</i> times	\s	space
{ , <i>m</i> }	no more than <i>m</i> times	\S	not space
{ <i>n</i> , }	at least <i>n</i> times	.	any symbol



Anchors

- Anchors match between characters.
- Used to assert that the characters you are matching must appear in a certain place.
- `/\bat\b/` matches “at work” but not “batch”.

Anchor	Matches
<code>^</code>	start of line
<code>\$</code>	end of line
<code>\b</code>	<i>word boundary</i>
<code>\B</code>	not boundary
<code>\A</code>	start of string (rare)
<code>\Z</code>	end of string (rare)
<code>\z</code>	raw end of string (rare)



ALTERNATION – “|” (pipe)

- In **RE**, “|” means “or”.
- You can put a full expression on the left and another full expression on the right.
- Either can match.
- **/seek | seeks | sought/**
 - matches “seek”, “seeks”, or “sought”.
- **/seeks? | sought/**
 - matches “seek”, “seeks”, or “sought”.



Grouping

- Everything within (...) is grouped into a single element for the purposes of *repetition* and *alternation*.
- The expression / **(la)+** / matches “**la**”, “**lala**”, “**lalalala**” but not “**all**”.
- / **schema(ta)?** / matches “**schema**” and “**schemata**” but not “**schematic**”.



Grouping Example

- What regular expression matches **“eat”, “eats”, “ate” and “eaten”**?



Grouping Example

- What regular expression matches “**eat**”, “**eats**”, “**ate**” and “**eaten**”?
- `/eat(s|en)?|ate/`
- Add word boundary anchors to exclude “**sate**” and “**eating**”:
- `/\b(eat(s|en)?|ate)\b/`



Lab: Part I (in the practicum file)

Repeater	Count	Shrtct	Name	Anchor	Matches
?	zero or one	\d	digit	^	start of line
+	one or more	\D	not digit	\$	end of line
*	zero or more	\w	word	\b	word boundary
{ <i>n</i> }	exactly <i>n</i> times	\W	not word	\t	TAB symbol
{ <i>n</i> , <i>m</i> }	between <i>n</i> and <i>m</i> times	\s	space	\n	new line
{ , <i>m</i> }	no more than <i>m</i> times	\S	not space	 	“or” alternation
{ <i>n</i> , }	at least <i>n</i> times	.	any symbol	(...)	capture group
				[...]	class



Replacement

- Regex most often used for search/replace
- Text editors:
 - Search Window: **pattern**
 - Replace Window: **replacement**



Capture

- During searches, (...) groups capture patterns for use in replacement.
- Special variables \1, \2, \3 etc. contain the capture
 - in *Sublime Text*: \$1, \$2, \$3
- /(\d\d\d) - (\d\d\d\d) / “123-4567”
 - \1 (\$1) contains “123”
 - \2 (\$2) contains “4567”



CAPTURE & REFORMAT

- How to convert “Schwarzenegger, Arnold” to “Arnold Schwarzenegger”?

-

-

-

-



CAPTURE & REFORMAT

- How to convert “Schwarzenegger, Arnold” to “Arnold Schwarzenegger”?
- Search: `/(\w+) , (\w+)/`
- Replace (a): `/\2 \1/`
- Replace (b): `/$2 $1/`
- (!) Before hitting “Replace”, make sure that your match does not catch what you do NOT want to change



Lab: Part II (in the practicum file)

Repeater	Count	Shrtct	Name	Anchor	Matches
?	zero or one	\d	digit	^	start of line
+	one or more	\D	not digit	\$	end of line
*	zero or more	\w	word	\b	word boundary
{ <i>n</i> }	exactly <i>n</i> times	\W	not word	\t	TAB symbol
{ <i>n</i> , <i>m</i> }	between <i>n</i> and <i>m</i> times	\s	space	\n	new line
{ , <i>m</i> }	no more than <i>m</i> times	\S	not space	 	“or” alternation
{ <i>n</i> , }	at least <i>n</i> times	.	any symbol	(...)	capture group
				[...]	class



Finding Toponyms

- *Very Simple*: Construct regular expressions that finds references all Austrian cities.



Finding Toponyms

- *Very Simple*: Construct regular expressions that finds references all Austrian cities.
- Simply connect all toponyms from the list with a pipe symbol “|”



Finding Toponyms

- *A Bit Tricky*: Construct regular expression that finds only cities from 1) Lower Austria; 2) Salzburg.



Finding Toponyms

- *A Bit Tricky*: Construct regular expression that finds only cities from 1) Lower Austria; 2) Salzburg.

- Option I:

```
\b([\w ]+) \ (Lower Austria\)  
\b([\w ]+) \ (Salzburg\)
```

- Option II (cooler):

```
\b([\w ]+) (?= ( \ (Lower Austria\)))  
\b([\w ]+) (?= ( \ (Salzburg\)))
```



RE for Highlighting Toponyms

157	حسب وتجرى في هذه الشعب الفراتية السفن الى
158	الكوفة وفي دجلة الى الموصل ، وذكر الشمشاطي PageV01P120
159	في تاريخه ان المنصور لما أراد بناء مدينة السلام احضر
160	أكبر من عرف من أهل الفقه والعدالة والأمانة والمعرفة
161	بالهندسة وكان فيهم ابو حنيفة النعمان ابن ثابت والحجاج بن أرطاة وحشر
162	الصناع والفعلة من الشام والموصل والجبل وسائر اعماله وامر بخطها وحفر
163	الأساسات في سنة 145 وتمت في سنة 49 وجعل عرض السور من
164	أسفل خمسين ذراعاً وجعلها بثمانية أبواب اربعة داخله صغار
165	واربعة خارجة كبار باب البصرة وباب الشام
166	وباب خراسان وباب الكوفة وجعل الجامع والقصر وسطها وقبله
167	جامع الرصافة اصح منه ، ووجدت في بعض
168	خزائن الخلفاء ان المنصور أنفق على مدينة السلام اربعة آلاف ألف وثمانمائة
169	وثلاثة وثلاثين درهماً لان اجرة الأستاذ كانت قيراطاً والروزكاري
170	حبتين النهروان مدينة ذات جانبيين الشرقي
171	اعمر رحبة عامرة بينهما الجسر الجامع في الجانب الشرقي
172	والحاج ينزلون على هذا الشط الدسكرة مدينة
173	صغيرة سوقها واحد طويل الجامع أسفله
174	غام بأزاج جلولا حولها أشجار غير حصينة
175	وهذه المدن وخانقين على جادة خلوان ليس لهن بهاء ولا هن
176	لائقات ببغداد وصرصر أيضاً كبعض قرى فلسطين النهر الى جانبها وكذلك نهر
177	الملك والصراة قرى واما قصر هبيرة فمدينة كبيرة جيدة
178	الأسواق يجيئهم الماء من الفرات كثيرة الحاكة واليهود والجامع في السوق
179	وبابل صغيرة نائية عن الطريق والجادة على جسرهما ، وسائر
180	مدن هذا الوجه على ما وصفنا مثل النيل وعبدس وكوثا ، ومدينة إبراهيم



RE for Finding Date Statements

- Download settings and examples for EditPad Pro:
<https://tinyurl.com/tutorial-re>
- In EditPad Pro:
 - Open: **"0748Dhahabi.TarikhIslam.Shamela0035100-ara1.sample"**
 - Open: **"year_Statement_re.txt"**
 - Copy/paste the regular expression into the search window
 - Switch back to **"0748Dhahabi.TarikhIslam.Shamela0035100-ara1.sample"**
- The date statements in the excerpt from al-Dahabī's *Ta'rīḥ al-islām* should appear highlighted with a different color



RE for Finding Date Statements

265	### \$153 - BIO MAN\$ - الفضل بن أحمد بن محمد بن متويه،
266	أبو عمرو الكاكوي، [الوفاة: 506 هـ]
267	كان يقال لأبيه كاكو.
268	--- teachers #
269	سمع من: عبد الخافر الفارسي، وأبي عثمان الصابوني، وابن مسرور.
270	بإفادة والده.
271	قال أبو سعد السمعاني: أجاز لي، وحدثني عنه جماعة، وتوفي ليلة
272	عيد الفطر، وكان مولده في سنة تسع وثلاثين.
273	ومن الرواة عنه ولده، وبقي إلى سنة أربع وخمسين، وروى أبوه أحمد.
274	كاكو عن: أبي عبد الله بن نظيف، PageV11P080
275	### \$154 - BIO MAN\$ - الفضل بن محمد بن عبيد بن محمد
276	بن محمد بن مهدي، أبو محمد القشيري، النيسابوري، [الوفاة: 506 هـ]
277	شيخ، ثقة، مشهور، من بيت العدالة والصلاح، كان مبالغاً في
278	الاحتياط [ص: 81] في الشهادات، ومن أعيان العدول، وكان صوفياً.
279	مليحاً، خيراً.
280	--- teachers #
281	سمع: عبد الرحمن بن حمدان النصروبي، وعبد القاهر.
282	أبا منصور البغدادي، وأبا حسان المزكي، وأبا الحسين الفارسي، وحدث
283	ببغداد لما حج، روى عنه: أبو الفتح محمد بن عبد السلام الكاتب.
284	وغيره.
285	ولد سنة عشرين وأربعمائة، وتوفي في رمضان، وهو أخو عبيد القشيري.
286	سيأتي، PageV11P080

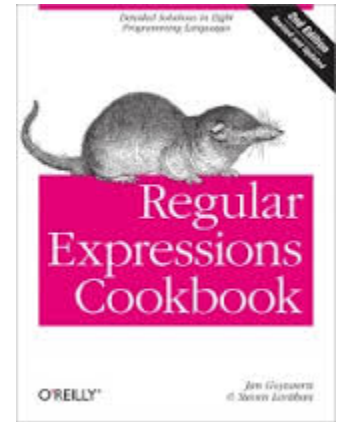


To keep in mind

- RE are “greedy,” i.e. they tend to catch more than you may need. Always test!
- Test before applying! (In text editors *Ctrl*+*Z* (*Win*), *Cmd*+*Z* (*Mac*) can help to revert changes)
- Check the language/application-specific documentation: some common shortcuts are not universal (**\1** vs **\$1**, for example)



SOME READINGS



- Amazon.com

- <http://www.amazon.com/Regular-Expressions-Cookbook-Jan-Goyvaerts/dp/1449319432/>
- <http://www.amazon.com/Mastering-Regular-Expressions-Jeffrey-Friedl/dp/0596528124/>

- Free Online Readings

- <http://www.regular-expressions.info/>
- <http://ruby.bastardsbook.com/chapters/regexes/>

- Cheat Sheets

- <http://krijnhoetmer.nl/stuff/regex/cheat-sheet/>
- <http://www.rexegg.com/regex-quickstart.html>

- Interactive tutorial

- <http://regexone.com/>

