Регулярные выражения

Отдел лингвистики

При написании регулярки можно пользоваться обозначениями, приведенными в таблице:

Представление	Значение	Эквивалент	Комментарий	Пример
\w	любая буква	[A-яA-zЁё]	Будет ловиться буква как кириллического, так и латинского алфавитов вне зависимости от регистра. При этом не ловятся буквы с диакритиками (Àà, Èè, Ùù, Éé, Çç, å) и особые буквы некоторых алфавитов типа норвержских æ, ø. Буквы других алфавитов (арабского, китайского) ловиться не будут. Цифры и нижнее подчёркивание не ловятся.	Регулярное выражение "Пет\w" будет ловить слова "Петя", "ПетD", "Петю", "Петр" и т.д. Регулярное выражение "\wоля" будет ловить слова "Коля", "коля", "Uоля", "толя" и т.д.
\W	не буква	[^A-яA-zЁё]	Соответственно, помимо небуквенных знаков типа -!,.?%;№, ловит еще и буквы с диакритиками и нетипичные буквы алфавитов, составленных на основе латиницы или кириллицы (например, корякскую "ƕ"), а также буквы не кириллического и не латинского алфавита.	
\d	любая цифра	[0-9]		
\D	не цифра	[^0-9]		
\s	пробел или таб	[\t]	Ловит не только пробел, но и табуляцию. Но не перевод строки.	
\S	не пробел и не таб	[^ \t]	Причем, перенос строки здесь тоже не матчится.	
.	любой символ, за исключением переноса строки			(.*) – все, что угодно (буквы, символы, пунктуационные знаки, разделители любого рода) любое количество раз, в том числе и ноль раз. Такое лучше не писать никогда даже внутри выражения. А если же написать регулярку, состоящую только из (.*), то TM гарантированно сломается. Кстати, начиная с версии TM 6.9, так поступить не получится: будет выдаваться ошибка (в более поздних версиях она звучит как "*Слишком короткое регулярное выражение*"; эта ошибка возникнет при создании регулярки, которая может сматчить слишком короткий текст – длиной в ноль или один символ).
[ ]	символьный класс		Внутри скобок задаются символы, один из которых может встретиться в перехватываемой строке. При этом в составе такого	Регулярное выражение "Пет[яюе]" поймает и "Петя", и "Пете", и "Петю". [0-9] соответствует всем цифрам от нуля до девяти. [А-яа-я] соответствует всем буквам русского алфавита, кроме "Ёё".

Представление	Значение	Эквивалент	Комментарий	Пример
			класса теряют свои свойства служебные символы [ ] \ ^ $. | ? * + ( ) { } Но здесь нужно сделать оговорку про скобки [и]. Чтобы не было недоразумений, лучше экранировать или выносить вперед (в начало списка) закрывающую скобку -]. Например, на выражение ***[][1s]{3}*** поймаются такие последовательности: *111*, *sss*, *]]]*, *[[[*, *1s]*, *1s[* и другие.	Также возможна запись [А-я]. [А-Г] соответствует русским заглавным буквам от "А" до "Г" (иначе можно было бы это записать так: [АБВГ], а [А-о] - всем русским заглавным буквам, кроме "Ё", а также русским строчным буквам от "а" до "о", за исключением "ё". Внутри квадратных скобок можно задавать одновременно и записанный с помощью дефиса символьный класс, и ряд обычных символов: [A-zA-я0-9Ёё!,%] будет реагировать на все буквы русского и английского алфавитов, на цифры, а также на восклицательный знак, запятую и знак процента.
[^]	исключение из набора			[^0-9] – не цифра.
()	группировка		Заключённое в скобки выражение рассматривается как единое целое.	О(хо)+ – После "О" идет как минимум одно "хо": поймается и "Охо", и "Охохохохохохо".
*	ноль или более	{0,}	Квантификатор ставится после символа, группы или символьного класса. Является жадным.	[0-9]* – любое количество цифр, в том числе и ни одной цифры.
+	один или более	{1,}	Квантификатор ставится после символа, группы или символьного класса. Является жадным.	\w+ – минимум одна буква.
?	ноль или одно	{0,1}	Квантификатор ставится после символа, группы или символьного класса.	Петр(ушка)? – реагирует как на "Петр", так и на "Петрушка".
{}	количество повторений		Ставится после символа, группы или символьного класса.	[0-9]{9} – девять любых цифр подряд. [0-9]{2,11} – от двух до одиннадцати любых цифр подряд. \w{0,20} – не больше двадцати букв.
|	или		В ситуации, когда выбор идет между несколькими односимвольными альтернативами, вместо | можно использовать [] Ср. "Пет(я|ю)" и "Пет[яю]".	Регулярное выражение "паспорт (номер|№)" поймает как строку "паспорт номер", так и "паспорт №".
\	экранирование		Экранировать следует те символы, которые являются служебными при написании регулярных выражений: [ ] \ ^ $. | ? * + ( ) { }	Регулярное выражение "[кч]то?" ловит слова "кто", "что", "кт" и "чт". Регулярное выражение "[кч]то\?" ловит только "кто?" и "что?", но не "что!"
\r\n\t	возврат каретки, новая строка, табуляция		Поскольку разные текстовые редакторы могут маркировать перенос строки различным образом, надежнее задавать перенос строки в регулярных выражениях так: [\r\n]{1,2}	Регулярное выражение "бутылка(\t)рома" поймает строку вида "бутылка+табуляция+рома", но не "бутылка+пробел+рома".

Представление	Значение	Эквивалент	Комментарий	Пример
^	начало строки			Регулярное выражение "^[Сс]ветило" сматчит только первое слово в строке "Светило дневное светило".
$	конец строки			Регулярное выражение "[Сс]ветило$" сматчит только последнее слово в строке "Светило дневное светило".

Вопрос отделения строки от других слов

Например, мы имеем выражение "[Рр]ом([ауе]|ом)?", представляющее собой все формы единственного числа лексемы "ром". Но такое выражение будет срабатывать и на словах "гром", "погром", "бром", "Роман", "романский" и т.д. Есть несколько способов избежать этого:

а) ввести пробелы в регулярное выражение: "\s[Рр]ом([ауе]|ом)?\s";

б) способ поуниверсальней - в начало и в конец поставить обозначение небуквенных и нецифровых символов: "[\D\W][Рр]ом([ауе]|ом)?[\D\W]";

в) другие способы: в список "нежелательных" в начале и конце строки символов добавить что нам надо. Например: "[^\d\w.][Рр]ом([ауе]|ом)?[^\d\w_]", чтобы вместе с "погромом" и "романским" не ловились последовательности вроде ".Рома_".

Что касается операторов "^" и "$", то крышечка означает, что весь перехваченный текст начинается со строки, заданной регулярным выражением, а знак доллара – что весь текст такой строкой заканчивается. Иными словами, регулярное выражение ^[Рр]ом([ауе]|ом)?$ будет срабатывать только на тексте, состоящем из одного слова: ром, рома, рому, ромом или роме.

Про жадность квантификаторов

Следует помнить, что квантификаторы "*" и "+" являются "жадными", то есть они пытаются съесть максимально длинный кусок строки, на который они могут налезть.
Например, мы создали регулярное выражение вида "\(.*\)". Оно предназначено для того, чтобы отлавливать информацию, данную в круглых скобках. Но в строке типа *Петя (он был очень любопытен) решил посмотреть, что находится в заброшенном доме (хотя мама просила его туда не лазить)* наше выражение увидит только одну группу в скобках, а именно: *(он был очень любопытен) решил посмотреть, что находится в заброшенном доме (хотя мама просила его туда не лазить) .* То есть оно среагирует на самую первую открывающую скобку и на самую последнюю закрывающую. Если бы мы ввели ограничение на то, что в скобках могут быть только символы русского алфавита и пробелы, этого бы не произошло. Так, регулярное выражением вида "\([А-я\s]+\)" поймало бы обе группы в скобках.

Пример. Разбор одного из регулярных выражений для детектирования паспорта гражданина РФ

Вот так выглядит это регулярное выражение:
*([0-9]{2})[\-:]?([0-9]{2})(:| *) *(([Нн]омер)|([Нн]ом\.)|N|№|([Nn]umber)|([Nn]um\.)|(No))(:| *) *([0-9][\-:]?){6}*

1. Начинается оно с группы *([0-9]{2})*. Она означает, что в тексте мы ожидаем увидеть две цифры в диапазоне от 0 до 9.
2. Затем следует символьный класс *[\-:]*, за которым стоит оператор "?". В символьном классе находятся три элемента: пробел, дефис (он здесь экранирован, так как внутри символьного класса дефис используется как служебный символ) и двоеточие. Знак вопроса означает, что элемент этого символьного класса необязателен: он может быть, а может и не быть в искомой строке.
3. После этого вновь следует группа, содержащая символьный класс, в котором находятся цифры, и указатель количества повторений элементов этого символьного класса: *([0-9]{2}).* Вместе элементы регулярного выражения 3-5 задают возможный вид серии паспорта гражданина РФ: цифра+цифра+(пробел, или дефис, или двоеточие или ничего)+цифра+цифра.

4. Далее следует группа, в которой через оператор "или" задано двоеточие или любое количество пробелов: *(:| *)*
5. Потом идет пробел, оперируемый звездочкой, что означает, что пробел может встретиться любое количество раз, в том числе и ноль. Эта часть регулярного выражения делает несколько избыточной ту, что задана в пункте 6: там ведь тоже задано любое количество пробелов — в качестве альтернативы двоеточию. Учитывая то, что мы имеем в пункте 7, часть выражения, описанную в пункте 6, можно было бы упростить следующим образом: *:?* Необязательного двоеточия вполне хватит для того, чтобы выразить часть выражения из пункта 6, учитывая, что условие "любое количество пробелов" все равно будет задано в седьмом пункте.
6. Далее следует группа, задающая различные способы записи слова "номер": *(([Нн]омер)|([Нн]ом\.)|N|№|([Nn]umber)|([Nn]um\.)|(No))*. Это может быть собственно слово "номер" (с большой или с маленькой буквы), сокращение "ном." (также с большой или с маленькой буквы), записанные латиницей слова "number" или "num." (тоже с большой или с маленькой буквы), а также значок "№", латинская буква "N", а еще "No".
7. Затем вновь идет группа вида "двоеточие или любое количество пробелов": *(:| *)*.
8. Потом — вновь условие "любое количество пробелов", которое делает предыдущее условие избыточным, как это было показано выше.
9. Далее идет группа *([0-9][\-:]?){6}*. В ней записано два символьных класса, один из которых задает цифры от 0 до 9, а второй — возможные разделители этих цифр: двоеточие, пробел и дефис, который в этой части выражения должен быть экранирован. Разделители при этом, как видно, необязательны (т.к. после символьного класса с разделителями стоит знак вопроса), то есть после каждой из шести цифр может стоять один из трех заданных символов, а может и не стоять. В принципе, эту часть регулярного выражения можно было бы переписать следующим образом: (\d[\-:]?). Вся эта группа (цифры с опциональными разделителями) должна повториться ровно шесть раз.