

Регулярные выражения





Олег Булыгин

- Преподаватель на курсах "Основы языка программирования Python", "Продвинутый Python", "Python для анализа данных" в Нетологии
- ІТ-аудитор в ПАО "Сбербанк"



О чём мы поговорим сегодня

- Что такое регулярные выражения и чем они отличаются от простого поиска
- 2. Синтаксис регулярных выражений
- 3. Модуль re в Python
- 4. Примеры использования регулярных выражений



Регулярные выражения

Регулярные выражения (Regular Expressions, regex) — это строки, содержащие совокупность *обычных символов* и *специальных метасимволов*, которые описывают определенный шаблон.

Эти шаблоны, к примеру, можно использовать для того, чтобы:

- находить и заменять что-то в текстовых данных;
- валидировать строковые поля.

Шаблоны могут быть настолько сложными, что другими способами определить их будет крайне трудно.



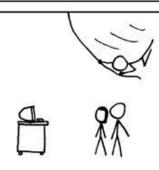
Во время изучения чего-то нового, я самозабвенно выдумываю невероятные ситуации, в которых это умение поможет мне спасти мир















«Если у вас есть проблема, и вы собираетесь решать ее с использованием регулярных выражений, то у вас уже две проблемы.»*

*шутка из интернетов

** это не шутка



Синтаксис регулярных выражений

Обычные символы в регулярных выражениях интерпретируются как *обычные строковые символы*.

Метасимволы могут быть интерпретироваться как:

- определенная группа обычных символов;
- *специфика других символов* (например, их количество или расположение).



Примеры метасимволов регулярных выражений – группы символов

\d	любая цифра. Аналогично [0-9]
\D	все, кроме цифры. Аналогично [^0-9]
\w	любая буква, цифра и символ подчеркивания
\W	все, кроме букв, цифр и символа подчеркивания
\s	любой пробельный символ, включая сам пробел: [\t\n\r\f\v]
\\$	все, кроме пробельных символов



Примеры метасимволов регулярных выражений – квантификаторы

*	любое количество вхождений предыдущего символа (группы символов), от 0 до бесконечности
+	количество вхождений предыдущего символа (группы символов) от 1 до бесконечности
?	0 или 1 вхождение предыдущего символа (группы символов)
{n}	точное количество вхождений – n раз предыдущего символа (группы символов)
{n, m}	количество вхождений не менее n и не более m раз предыдущего символа (группы символов)



raw string – подавляет значение *escape-последовательностей*, обозначается префиксом **r** перед обычной строкой.

(i)

Настоятельно рекомендуется использовать всегда при работе с регулярными выражениями для избежания проблем.



re.findall(pattern, string) – ищет во всем тексте, возвращает список всех найденных совпадений;

re.split(pattern, string) – разделяет строку по заданному шаблону;

re.match(pattern, string) – ищет по заданному шаблону в начале строки;

re.search(pattern, string) – ищет во всем тексте, возвращает первое совпадение;



re.findall(pattern, string) — ищет во всем тексте, возвращает список всех найденных совпадений;

re.split(pattern, string) – разделяет строку по заданному шаблону;

re.match(pattern, string) – ищет по заданному шаблону в начале строки;

re.search(pattern, string) – ищет во всем тексте, возвращает первое совпадение;



re.findall(pattern, string) — ищет во всем тексте, возвращает список всех найденных совпадений;

re.split(pattern, string) – разделяет строку по заданному шаблону;

re.match(pattern, string) – ищет по заданному шаблону в начале строки;

re.search(pattern, string) – ищет во всем тексте, возвращает первое совпадение;



Еще примеры метасимволов

•	любой символ
^	1) начало строки, 2) инвертирование («всё, кроме») при применении с []
\$	конец строки
\	символ экранирования. Например, символ точки: \.
	логическое "ИЛИ"
[]	применяет "ИЛИ" к набору символов, любой из которых может встретиться в тексте
	Например, [а-яёА-ЯЁ] – любая буква русского алфавита в любом регистре

re.sub(pattern, repl, string) – ищет шаблон в строке и заменяет его на указанную подстроку.



Скобочные группы

Часть шаблона можно заключить в скобки (...). Это называется «скобочная группа».

Это позволяет:

- 1. поместить часть шаблона в отдельную структуру (группу);
- 2. если установить квантификатор после скобок, то он будет применяться ко всему содержимому скобки, а не к одному символу.
- квантификатор внутри группы и после нее это совершенно разные вещи!

При наличии подгрупп в шаблоне можно ссылаться на них через их порядковый номер при помощи \1, \2, \3, ...
Группам можно давать собственные имена при помощи (?Р<имя>...)



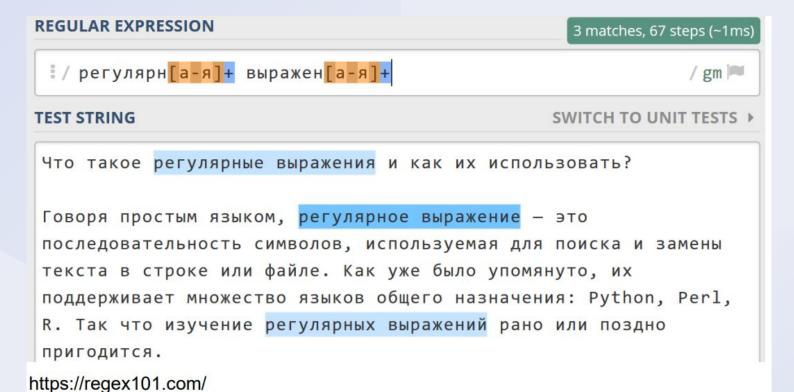
Non-capturing groups и lookarounds

Non-capturing группы позволяют найти группу, но не включать ее в требуемый шаблон

Позиционная проверка (lookaround) – используется при поиске шаблона, которому обязательно предшествует или следует другой шаблон.

?:	Выделить группу, но не включать ее в результат
?=	Положительный lookahead (ищет шаблон, если он находится до указанной группы)
?!	Отрицательный lookahead (ищет шаблон, если он не находится до указанной группы)
? <=	Положительный lookbehind (ищет шаблон, если он находится после указанной группы)
? </th <th>Отрицательный lookbehind (ищет шаблон, если он не находится после указанной группы)</th>	Отрицательный lookbehind (ищет шаблон, если он не находится после указанной группы)

Сервисы для проверки регулярных выражений





https://docs.python.org/3/library/re.html – документация по регулярным выражениям

https://regex101.com/ - тестер регулярных выражений

https://tproger.ru/translations/regular-expression-python/ – хорошая понятная статья по регулярным выражениям





Регулярные выражения

Вопросы?

Соцсеть fb.com/obulygin91

Почта obulygin91@ya.ru