hillel IT school

Lesson 26

План заняття

- Регулярні вирази;
- Прапори;
- Якорі;
- Квантифікатори;
- Групи захоплення;
- Методи регулярних виразів та рядків;

Регулярні вирази

В JavaScript регулярні вирази реалізовані окремим об’єктом RegExp та інтегровані у методи рядків.

Регулярний вираз, складається з шаблону (також кажуть “патерн”) і необов’язкових прапорів.

Синтаксиси:

1. regexp = new RegExp ("^$", "g");
2. regexp = /^$/gmi;

Основна різниця між цими двома синтаксами полягає в тому, що слеши /.../ не допускають жодних вставок змінних (на зразок тих, що прописуються через ${...}). Вони повністю статичні.

Прапори

У JavaScript є 6 прапорів:

1. i - З цим прапором пошук не залежить від регістру: немає різниці між A та a;
2. g - З цим прапором пошук шукає всі збіги, без нього – лише перше;
3. m - Багаторядковий режим;
4. s - Вмикає режим “dotall”, при якому крапка . може відповідати символу нового рядка \n;
5. u - Вмикає повну підтримку Юнікоду;
6. y - Режим пошуку на конкретній позиції в тексті.

Пошук у рядку

Метод str.match(regexp) для рядка str повертає збіги з регулярним виразом regexp.

У нього є три режими роботи:

1. Якщо в регулярного виразу є прапор g, то він повертає масив всіх збігів;
2. Якщо такого прапора немає, то повертається лише перший збіг у вигляді масиву, в якому за індексом 0 знаходиться збіг, і є властивості з додатковою інформацією про нього;
3. Якщо збігів немає, то, незалежно від наявності прапора g, повертається null.

Заміна у рядку

Метод str.replace(regexp, replacement) замінює збіги з regexp у рядку str на replacement (всі збіги, якщо є прапор g, інакше тільки перше).

Спецсимволи	Дія у рядку заміни
$&	вставляє всі знайдені збіги
$`	вставляє частину рядка до збігу
$'	вставляє частину рядка після збігу
$n	якщо n це 1-2 значне число, вставляє вміст n-ї скобочної групи регулярного виразу
$<name>	вставляє вміст скобочної групи з ім’ям name
$$	вставляє символ "$"

Тест у рядку

Метод regexp.test(str) перевіряє, чи є хоч один збіг, якщо так, то повертає true, інакше false.

Символьні класи

Символьний клас – це спеціальне позначення, яке відповідає будь-якому символу з певного набору.

- \d (digit) відповідає “будь-якій одній цифрі”;
- \s (space) пробільні символи: включає символ пробілу, табуляції \t, перенесення рядка \n і деякі інші рідкісні пробілові символи, що позначаються як \v, \f і \r.
- \w (word) символ “слова” – літера латинського алфавіту, цифра або символ підкреслення _. Нелатинські літери (наприклад, кирилиця чи хінді) не належать до \w.

Зворотні символьні класи

“Зворотній” означає, що він відповідає всім іншим символам:

- \D Не цифра: будь-який символ, окрім \d, наприклад, літера.
- \S Не пробіл: будь-який символ, окрім \s, наприклад, літера.
- \W Будь-який символ, окрім \w, тобто не букви з латиниці, не символ підкреслення і цифра.

Крапка

Крапка . – це спеціальний символьний клас, який відповідає “будь-якому символу, крім символу нового рядка”.

За замовчуванням крапка не відповідає символу нового рядка \n. Але якщо ми б хотіли, щоб крапка означала буквально “будь-який символ”, включаючи новий рядок?

Ось що робить прапор s. Якщо регулярний вираз містить його, то крапка . відповідає буквально будь-якому символу. Прапор s не підтримується в IE.

Якорі

Символи каретки ^ і долара $ мають особливе значення в регулярному виразі. Їх називають “якорі”.

Каретка ^ збігається з початком тексту, а долар $ з кінцем.

Обидва якорі разом ^...$ часто використовуються для перевірки того, чи рядок повністю відповідає шаблону.

\b

Межа слова \b – це така ж сама перевірка як і ^ та $.

Коли механізм регулярних виразів (програмний модуль, який здійснює пошук регулярних виразів) стикається з \b, він перевіряє, чи є позиція в рядку межею слова.

Є три різні позиції, які кваліфікуються як межі слова:

- На початку рядка, якщо його перший символ представляє собою буквенний символ \w.
- Між двома символами в рядку, де один є буквенним символом \w, а інший – ні.
- В кінці рядка, якщо його останній символ представляє собою буквенний символ \w.

Ми можемо використовувати \b не тільки зі словами, а й з цифрами.

Межа слова \b не працює з алфавітами, відмінними від латинського

Екранування

Існують також спеціальні символи, які мають особливе значення в регулярних виразах, наприклад [ ] { } ( ) \ ^ $. | ? * +. Вони використовуються, щоб посилити можливості пошуку.

Щоб використовувати спеціальний символ як звичайний, додайте перед ним бекслеш: \.

Символ слешу '/' не є спеціальним символом, але в JavaScript він використовується для відкриття та закриття регулярного виразу: /...pattern.../, тому ми також повинні екранувати його.

Набори та діапазони [...]

Всередині квадратних дужок [...] означає “шукати будь-який символ з-поміж заданих”.

- Чіткі значення [az]
- Діапозони [a-z]

Діапозони:

- \d – це те саме, що й [0-9],
- \w – це те саме, що й [a-zA-Z0-9_],
- \s – це те саме, що й [\t\n\v\f\r]

Діапазони виключень

Діапазони “виключень” виглядають наступним чином: [^…].

До прикладу:

- [^aeyo] – будь-який символ окрім 'a', 'e', 'y' or 'o'.
- [^0-9] – будь-який символ окрім цифр, так само як і \D.
- [^\s] – будь-який не пробільний символ \S.

Екранування всередині [...]

В квадратних дужках ми можемо використовувати велику кількість спецсимволів без екранування:

- Символ . + ( ) не потребує екранування.
- Дефіс - не потребує екранування на початку, або в кінці (тобто коли не може означати діапазон).
- Каретка ^ екранується лише на початку (без екранування означає набір символів-виключень).
- Закриваюча квадратна дужка] завжди потребує екранування (у випадках, коли нам потрібно знайти цей символ).

Квантифікатори

- Кількість {n} - Квантифікатор додається до символу(або класу символів, набору [...], тощо) і позначає яка їх кількість нам потрібна.
- Один, або більше +
- Нуль, або один ? так само як і {0,1}
- Нуль, або більше * так само, як і {0,}

Жадібні та лініві квантифікатори

Жадібний режим.

Аби знайти збіг, рушій регулярних виразів використовує наступний алгоритм:

Для кожної позиції в рядку:

- Спробувати виявити збіг на цій позиції.
- Якщо збігу немає, перейти до наступної позиції.

В жадібному режимі (типово) квантифікований символ повторюється максимально можливу кількість разів.

Лінивий режим

Його алгоритм: “повторювати мінімальну кількість разів”.

Ми можемо включити його, поставивши знак питання '?' після квантифікатора, і отримати *?, +? чи навіть ?? для '?'.

Група захоплення

Частину виразу можна обгорнути в “група захоплення” - (...).

Це надає:

- Змогу отримати частину збігу в якості окремого елементу в масиві результатів;
- Можливість застосування квантифікатора після дужок до всього їх вмісту.

Ми можемо вставити групу у середину групи. Тоді дужки нумеруються зліва направо. Пошукова система запам’ятовує вміст збігу для кожної з них та дозволяє отримати його всередині результату.

Повний збіг завжди представлений в елементі з нульовим індексом.

Далі йдуть групи, нумеровані зліва направо за відкриваючою дужкою.

Альтернація

Альтернація – це термін у регулярному виразі, який насправді є простим “АБО”.

Вона позначається символом вертикальної лінії |. Альтернація працює з будь-якими виразами.

Методи

- str.match(regexp) - знаходить збіги для regexp в рядку str.
- str.matchAll(regexp) – це “новіший, покращений” варіант str.match.

В основному, його використовують для пошуку всіх збігів з усіма групами.

Існує 3 відмінності від match:

- Він повертає ітерований об’єкт із збігами замість масиву. Ми можемо отримати з нього звичайний масив за допомогою Array.from.
- Кожен збіг повертається у вигляді масиву з групами захоплення (той самий формат, що й str.match без прапору g).
- Якщо результатів нема, метод повертає порожній ітерований об’єкт замість null

Методи

- str.split(regexp|substr, limit) - Ділить рядок, використовуючи регулярний вираз (або підрядок) в якості роздільника.
- str.search(regexp) - повертає позицію першого збігу або -1, якщо нічого не знайдено
- str.replace(str|regexp, str|func) - замінює, видаляє або додає нову строку.
- str.replaceAll(str|regexp, str|func) - Цей метод, по суті, такий самий, що й str.replace, з двома значними відмінностями:

Якщо перший аргумент є рядком, він замінює всі входження в рядку, тоді як replace замінює лише перше входження.

Якщо перший аргумент є регулярним виразом без прапору g, виникне помилка. З прапором g, метод працюватиме аналогічно до replace.

Методи

- regexp.exec(str) - Метод regexp.exec(str) повертає збіг для regexp в рядку str. Його поведінка залежить від наявності в регулярному виразі прапору g.

Якщо нема прапору g, тоді regexp.exec(str) повертає перший збіг у вигляді str.match(regexp). Ця поведінка не додає нічого нового.

Але за наявності g:

- Виклик regexp.exec(str) повертає перший збіг та зберігає позицію після нього всередині властивості regexp.lastIndex.
- Наступний виклик починає пошук з позиції regexp.lastIndex, повертає наступний збіг та зберігає позицію після в regexp.lastIndex.
- …І так далі.

Методи

- regexp.test(str) - шукає збіг та повертає true/false в залежності від його наявності.

Дякую за увагу