Week 14 Regular Expressions (RegEx)-For text searching & manipulation

Reference sites

<http://www.regexr.com> , <https://www.tutorialspoint.com/python/python_reg_expressions.htm>

HOW TO

import the [re](https://docs.python.org/3/library/re.html#module-re) module, and compile a RE:

Ex.

**p = re.compile(r'\d+')**

MATCHING EXPRESSIONS

|  |  |
| --- | --- |
| **Method** **/** **Attribute** | **Purpose** |
| match() | Determine if the RE matches at the beginning of the string. |
| search() | Scan through a string, looking for any location where this RE matches. |
| findall() | Find all substrings where the RE matches, and returns them as a list. |

**match()** and **search()** return None if no match can be found. If successful, a match object instance is returned, containing information about the match: where it starts and ends, the substring it matched, and more.

Ex.

**p.findall('12 drummers drumming, 11 pipers piping, 10 lords a-leaping')**

**['12', '11', '10']**

-MATCH object instances

| **Method/Attribute** | **Purpose** |
| --- | --- |
| group() | Return the string matched by the RE |
| start() | Return the starting position of the match |
| end() | Return the ending position of the match |

**Python specifics:**

The *re.search* function returns a **match** object on success, **none** on failure. Use *group(num)* or *groups()* function of

**match** object to get matched expression.

Ex.

**group(num=0) =>** Returns specific **subgroup** num *or* entire match

**groups() =>**  Returns **all** matching subgroups in a tuple (empty if there weren't any)

COMMON REGEX

Following table lists the regular expression syntax that is available in Python −

**Sr.No. Pattern & Description**

1 **^** Matches beginning of line.

2 **$** Matches end of line.

3 **.** Matches any single character except newline. Using m option allows it to match newline as well.

4 **[...]** Matches any single character in brackets.

5 **[^...]** Matches any single character not in brackets

6 **re\*** Matches 0 or more occurrences of preceding expression.

7 **re+** Matches 1 or more occurrence of preceding expression.

8 **re?** Matches 0 or 1 occurrence of preceding expression.

9 **re{ n}** Matches exactly n number of occurrences of preceding expression.

10 **re{ n,}** Matches n or more occurrences of preceding expression.

11 **re{ n, m}** Matches at least n and at most m occurrences of preceding expression.

12 **a| b** Matches either a or b.

13 **\w** Matches word characters.

14 **\W** Matches nonword characters.

15 **\s** Matches whitespace. Equivalent to [\t\n\r\f].

16 **\S** Matches nonwhitespace.

17 **\d** Matches digits. Equivalent to [0-9].

18 **\D** Matches nondigits.

19 **\b** Matches word boundaries when outside brackets. Matches backspace (0x08) when inside brackets.