**Regular expression**

A regular expression describes one or more strings to match when you search a body of text. The expression serves as a template for matching a character pattern to the string that is being searched.

**Example**

When working on a particular applications let’s say one of the property values called id is dynamic, that is, on every run you notice that the value of id property is getting changed. So when you recorded your script, the value of id property was REUFT101. When you replayed the same script, value of id changed to REUFT120, on next iteration it changed to REUFT105. You can easily identify a pattern here whereby the last two digits of id value are getting changed.  
Can we write a single string that can match ALL these dynamic values? Yes, and the answer lies in regular expressions. You can write **REUFT1\d\d.**The first five characters were fixed so we need not regularize them, however the last two characters were dynamic digits. We need to ensure that our regular expression should be able to match any single digit between 0 to 9. The regular expression for that is **\d.** Since we need to match two digits we have used it twice **\d\d**.

**Types of regular expression**

|  |  |  |
| --- | --- | --- |
| **Meta**  **character** | **Behavior** | **Example** |
| \* | Matches the previous character or subexpression zero or more times.  Equivalent to {0,}. | zo\* matches z and zoooooooo . |
| + | Matches the previous character or subexpression one or more times.  Equivalent to {1,}. | zo+ matches zo and zooo , but not "z". |
| ? | Matches the previous character or subexpression zero or one time.  Equivalent to {0,1}. | zo? matches z and zo, but not zooo.  do(es)? matches the do or does but not matches dorwfes. |
| ^ | Matches the position at the start of the searched string. If the  **Multiline** property is set, ^ also matches the position following \n or \r. When used as the first character in a bracket expression, ^ negates the character set. | ^123 matches 123 but not matches 123456  ^\d{3} matches 3 numeric digits at the start of the searched string.  ^\d{3} match 458 or 4587455 but not match “12” or “1” .  [^abc] matches any character except a, b, and c.  [^abc ] matches qer12acgghb but not matches “abc”. |
| $ | Matches the position at the end of the searched string. If the **Multiline** property is set, $ also matches the position before \n or \r. | \d{3}$ matches 3 numeric digits at the end of the searched string.  \d{3}$ matches 12345er445 but not matches “12” or “qq12” |
| . | Matches any single character except the newline character \n. To match any character including the \n. | a.c matches axc but not matches “awwsedrc”.  a.+c matches axcfvwawc |
| [] | Marks the start and end of a bracket expression. | [1-4] matches 1234561jkn3knl4. [^aAeEiIoOuU]matches any non-vowel character sokjjubbva  [aeiou$] matches only vowel sokjjubbva.  [^abc] matches asecrewgc  [abc$] matches asecrewgc |
| {} | Marks the start and end of a quantifier expression. | a{2} matches aa but not matches ”aaa”. |
| () | Marks the start and end of a subexpression. Subexpressions can be saved for later use. | A(\d) matches "A0" to "A9". The digit is saved for later use. |
| \b | Matches a word boundary, that is, the position between a word and a space. | er\b matches the er ds er ge . |
| | | Indicates a choice between two or more items. | z|food matches zood food. (z|f)ood matches zood food. |
| \s | Any white-space character. This includes space, tab, and form feed. | \s matches qf aawvr4 124 |
| \S | Any non–white space character. | \S matches qf aawvr4 124 |
| \d | Matches a digit character.  Equivalent to [0-9]. | \d matches asd#$4234  . |
| \D | Matches a nondigit character.  Equivalent to [^0-9]. | \D  matches asd#$4234. |
| \w | Matches any of the following characters: A-Z, a-z, 0-9, and underscore.  Equivalent to [A-Za-z0-9\_]. | \w matches asdss.452g\_\_\_\_\_\_. |
| \W | Matches any character except A-Z, a-z, 0-9, and underscore.  Equivalent to [^A-Za-z0-9\_]. | \W+ matches asd$%^&\*@ #$4234all of the spaces. |