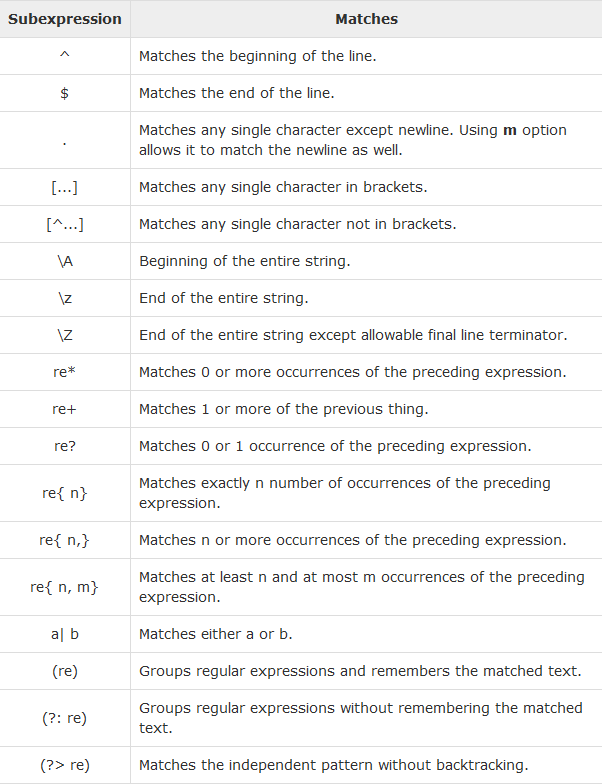
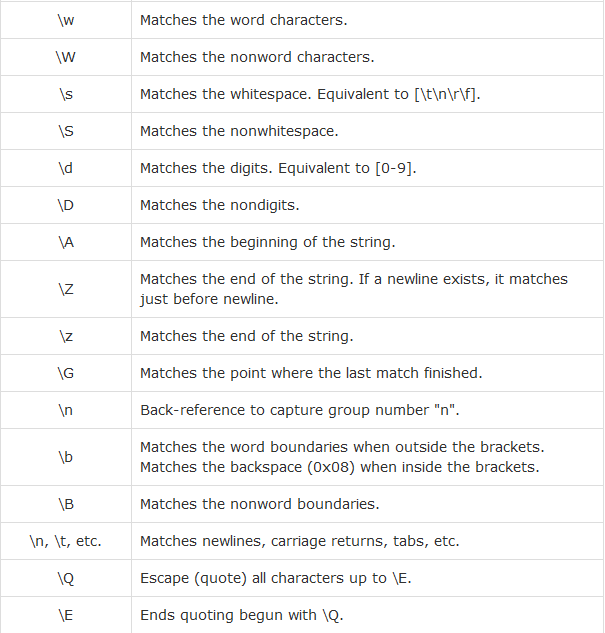
# 0. Table





http://www.tutorialspoint.com/java/java\_regular\_expressions.htm

Có tổng cộng 34 regular expression. Nào cùng tìm hiểu

# 1. […]

***Matches any single character in brackets.***

## Code:

**package** demo.com;

**import** java.util.\*;

**public** **class** test {

**public** **static** **void** main(String[] args) {

String scanMe = "abcdefghijkl";

String delim = "[aeoiu]";

Scanner scanner = **new** Scanner(scanMe);

scanner.useDelimiter(delim);

**while** (scanner.hasNext()) {

System.***out***.println(scanner.next());

}

}

}

## Result:



Ở đây ta sử dụng bất kỳ single character trong bracket để làm split, vì thế a,e,o,i,u sẽ là các split.

# 2. re+, re\*, re?

**re+: Matches 1 or more of the previous thing.**

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

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

## Code:

**package** demo.com;

**import** java.util.\*;

**public** **class** test {

**public** **static** **void** main(String[] args) {

String scanMe = "132xxxxx555";

String delim = "x";

Scanner scanner = **new** Scanner(scanMe);

scanner.useDelimiter(delim);

**while** (scanner.hasNext()) {

System.***out***.println(scanner.next());

}

}

}

## Result:



Ở đây vì ta chỉ sử dụng mỗi x làm split, vì thế nên có nhiều token được tạo ra, mà các token này là null

**Sử dụng re+: Xác định 1 hoặc nhiều x làm split**

String delim = "x+";



**Sử dụng re\*: Xác định 0 hoặc nhiều x làm split**

****

**Sử dụng re?: Xác định 0 hoặc 1 x làm split**

****

# 3. .

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

Nghĩa là nó match với bất kỳ ký tự nào ngoại trừ xuống dòng.

## Code:

**package** demo.com;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

String any = ".\*";

String test1 = "Hello", test2 = "29", test3 = "", test4 = "\n";

System.***out***.println(test1.matches(any));

System.***out***.println(test2.matches(any));

System.***out***.println(test3.matches(any));

System.***out***.println(test4.matches(any));

}

}

## Result:



Vì .\* là match với bất kỳ ký tự nào, với \* là 0 hoặc nhiều cho cái dấu chấm(.) nên 3 kết quả đầu đúng.

## Code2:

**package** demo.com;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

String any = ".+";

String test1 = "Hello", test2 = "29", test3 = "", test4 = "sa\ns";

System.***out***.println(test1.matches(any));

System.***out***.println(test2.matches(any));

System.***out***.println(test3.matches(any));

System.***out***.println(test4.matches(any));

}

## Result2:

Vì bây giờ là dấu +, nghĩa là 1 hoặc nhiều, nên test3 và test 4 false

# 4. \d

**Matches the digits. Equivalent to [0-9]**

Match với số 0 đến 9, tương đương [0-9]

## Code:

**package** demo.com;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

String numbersOnly = "\\d+";

String test1 = "Hello", test2 = "29", test3 = "", test4 = "a5";

System.***out***.println(test1.matches(numbersOnly));

System.***out***.println(test2.matches(numbersOnly));

System.***out***.println(test3.matches(numbersOnly));

System.***out***.println(test4.matches(numbersOnly));

}

}

## Result:

Vì \d match với số, thêm + là 1 hoặc nhiều nên chỉ có test2 là true. Nếu đổi + thành \* nữa thì cái test3 sẽ true luôn.

# 4. \D

Ngược với \d

# 5. \w

**Matches the word characters.**

Match với ký tự là word character: a-z, A-Z, 0-9.

Ở đây chấp nhận luôn các ký tự có gạch dưới, mà trong java chưa gặp nên thôi bỏ qua đi.

## Code:

**package** demo.com;

**import** java.text.DateFormat;

**import** java.text.NumberFormat;

**import** java.util.\*;

**public** **class** test {

**public** **static** **void** main(String[] args) {

String wordCharacter = "\\w\*";

String test1 = "Hello", test2 = "29", test3 = "", test4 = "\n", test5 = "#das";

System.***out***.println(test1.matches(wordCharacter));

System.***out***.println(test2.matches(wordCharacter));

System.***out***.println(test3.matches(wordCharacter));

System.***out***.println(test4.matches(wordCharacter));

System.***out***.println(test5.matches(wordCharacter));

}

}

## Result:

Ta thấy test4 và test5 chứa các ký tự k phải \w đã nói nên 2 nó ra false.

# 5. \W

Ngược với \w