Regex symbol list and regex examples

. Period, matches a single character of any single character, except the end of a line. For example, the below regex matches shirt, short and any character between sh and rt.

```
sh.rt
```

^ Carat, matches a term if the term appears at the beginning of a paragraph or a line. For example, the below regex matches a paragraph or a line starts with Apple.

```
^Apple
```

^ Carat inside a bracket, for example, the below regex matches any characters but a, b, c, d, e.

```
[^a-e]
```

$ Dollar sign, matches a term if the term appears at the end of a paragraph or a line. For example, the below regex matches a paragraph or a line ends with bye.

```
bye$
```

[ ] Square brackets, matches any single character from within the bracketed list. For example, the below regex matches bad, bed, bcd, brd, and bod.

```
b[aecro]d
```

– Hyphen, used for representing a range of letters or numbers,often used inside a square bracket. For example, the below regex matches kam, kbm, kcm, k2m, k3m, k4m and k5m.

```
k[a-c2-5]m
```

( ) Parentheses, groups one or more regular expressions. For example, the below regex matches codexpedia.com, codexpedia.net, and codexpedia.org.

```
codexpedia\.(com|net|org)
```

{n} Curly brackets with 1 number inside it, matches exactly n times of the preceding character. For example, the below regular expression matches 4 digits string, and only four digits string because there is ^ at the beginninga nd $ at the end of the regex.

```
^[\d]{4}$
```

{n,m} Curly brackets with 2 numbers inside it, matches minimum and maximum number of times of the preceding character. For example, the below regular expression matches google, gooogle and goooogle.

```
go{2,4}gle
```

{n,}, Curly brackets with a number and a comma, matches minimum number of times the preceding character. For example, the below regex matches google, gooogle, goooooogle, goooooogle, ….

```
go{2,}gle
```

| Pipe, matches either the regular expression preceding it or the regular expression following it. For example, the below regex matches the date format of MM/DD/YYYY, MM.DD.YYYY and MM-DD-YYY. It also matches MM.DD-YYYY, etc.

```
^(0[1-9]|1[012])[-/.](0[1-9]|[12][0-9]|3[01])[-/.][0-9]{4}$
```

? Question mark, matches 1 or 0 character in front of the question mark. For example, the below regular expression matches apple and apples.

```
apples?
```

*** Asterisk**, matches 0 or more characters in front of the asterisk. For example, the below regular expression matches cl,col,cool,cool,…,cooooooooool,…

```
co*l
```

+ Plus, matches 1 or more characters in fron of the plus. For example, the below regular expression matches col,cool, …,cooooooooooool,…

```
co+l
```

! Exclamation, do not matches the next character or regular expression. For example, the below regular expression matches the the characher q if the charachter after q is not a digit, it will matches the q in those strings of abdqk, quit, qeig, but not q2kd, sdkq8d.

```
q(?![0-9])
```

\ Backslash, turns off the special meaning of the next character. For example, the below regex treats the period as a normal character and it matches a.b only.

```
a\.b
```

\b Backslash and b, matches a word boundary. For example, “\bwater” finds “watergun” but not “cleanwater” whereas “water\b” finds “cleanwater” but not “watergun”.
\n Backslash and n, represents a line break.
\t Backslash and t, represents a tab.

\w Backslash and w, it is equivalent to [a-zA-Z0-9_], matches alphanumeric character or underscore. Conversely, Capital **\W** will match non-alphnumeric character and not underscore.
\d Backslash and d, matches digits 0 to 9, equivalent to [0-9] or [:digit]
[:alpha:] or **[A-Za-z]** represents an alphabetic character.
[:digit:] or **[0-9]** or **[\d]** represents a digit.
[:alnum:] or **[A-Za-z0-9]** represents an alphanumeric character.

This regex matches **email addresses**

```
\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}\b
```

This regex matches **websites links** ending with sites of .com, .org, .edu, .gov and .us

```
https?://(www\.)?[A-Za-z0-9]+\.(com|org|edu|gov|us)/?.*
```

This regex matches **social security numbers**.

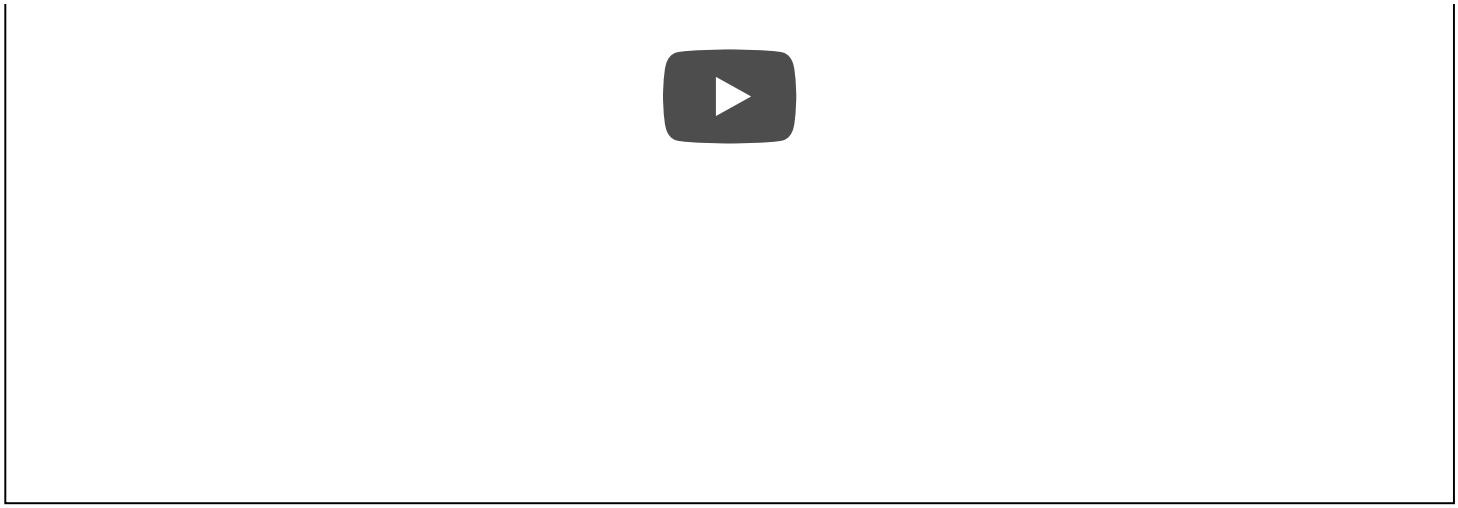
```
^[0-9]{3}-[0-9]{2}-[0-9]{4}$
```


Related Posts

Java regular expression example (https://www.codexpedia.com/java/java-regular-expression-example/)

Regex symbol list and regex examples (https://www.codexpedia.com/regex/regex-symbol-list-and-regex-examples/)

More (https://www.codexpedia.com/)



egular Expressions (RegEx) Tutorial #7 - Special Characters

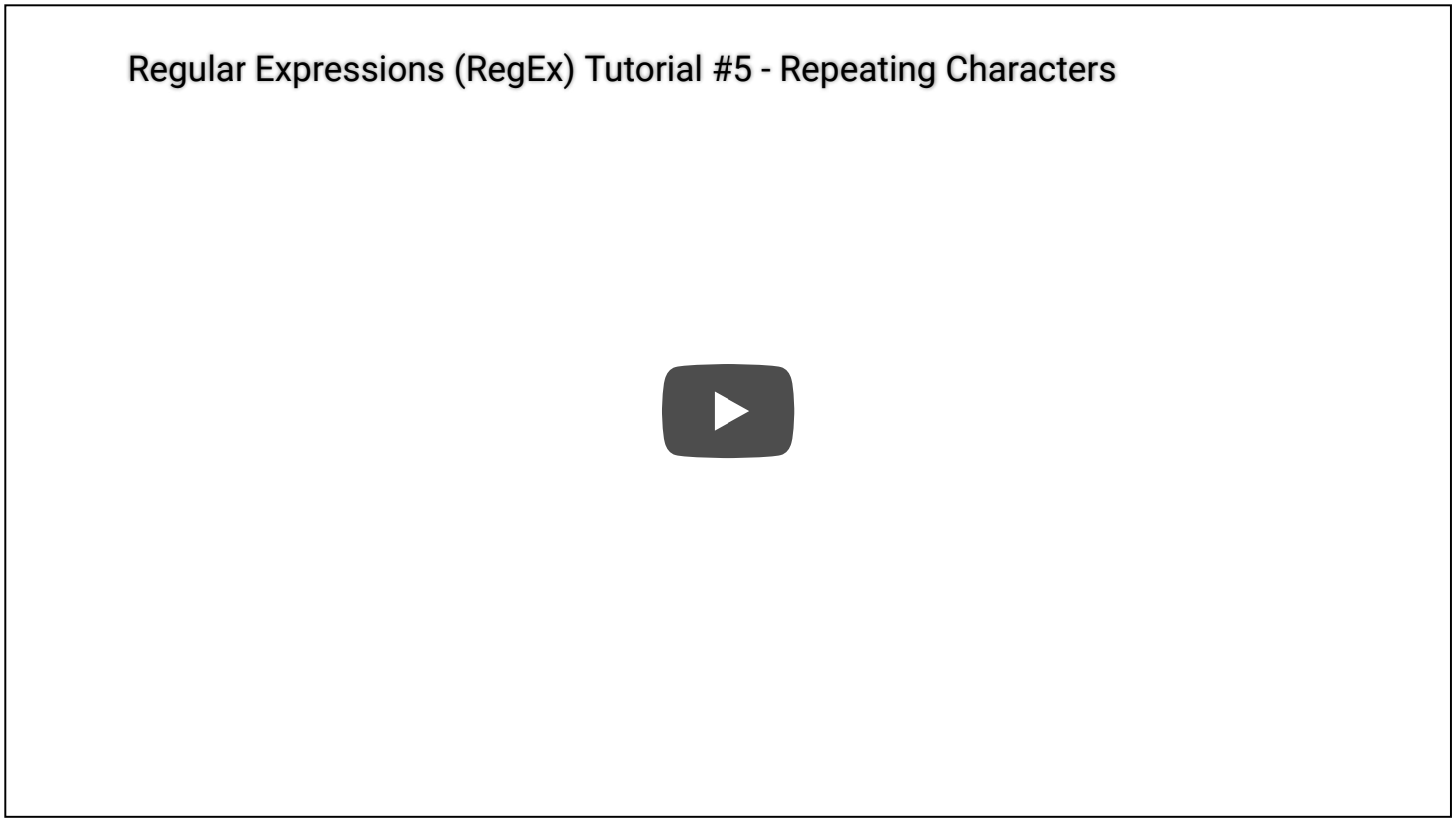
ey all, in this RegEx tutorial I'll introduce you to some more special characters which exhibit very diferent behaviours when aced in a RegEx pattern. DONATE ...

18-02-20T07:58:55.000Z

egular Expressions (Regex) Tutorial: How to Match Any Pattern of Text

this regular expressions (regex) tutorial, we're going to be learning how to match patterns of text. Regular expressions are tremely useful for matching ...

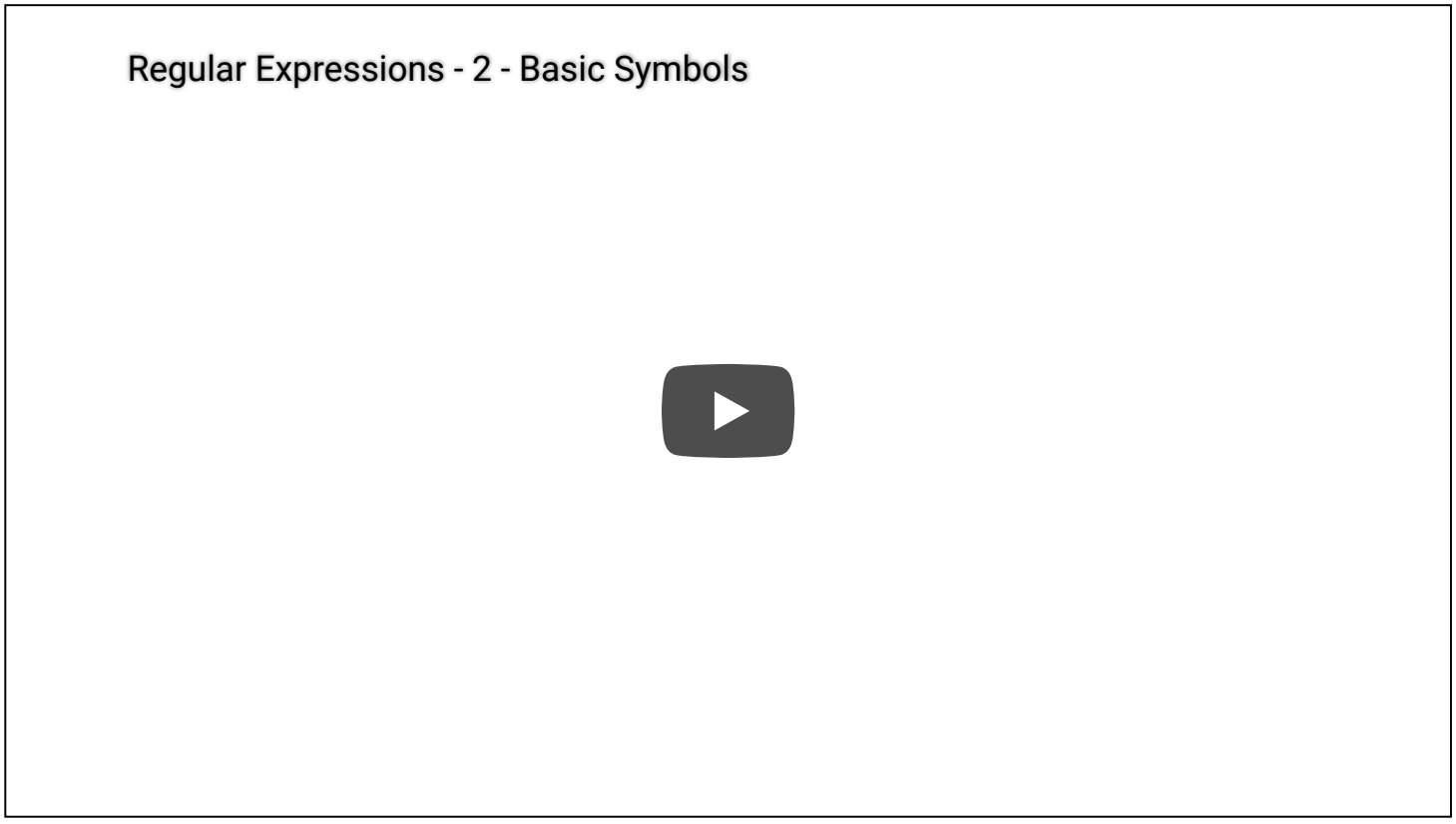
17-10-05T16:30:01.000Z



egular Expressions (RegEx) Tutorial #5 - Repeating Characters

ey all, in this RegEx tutorial I'll show you how wecan easily repeat characters in a pattern, rather than write tem all out by nd. DONATE ...

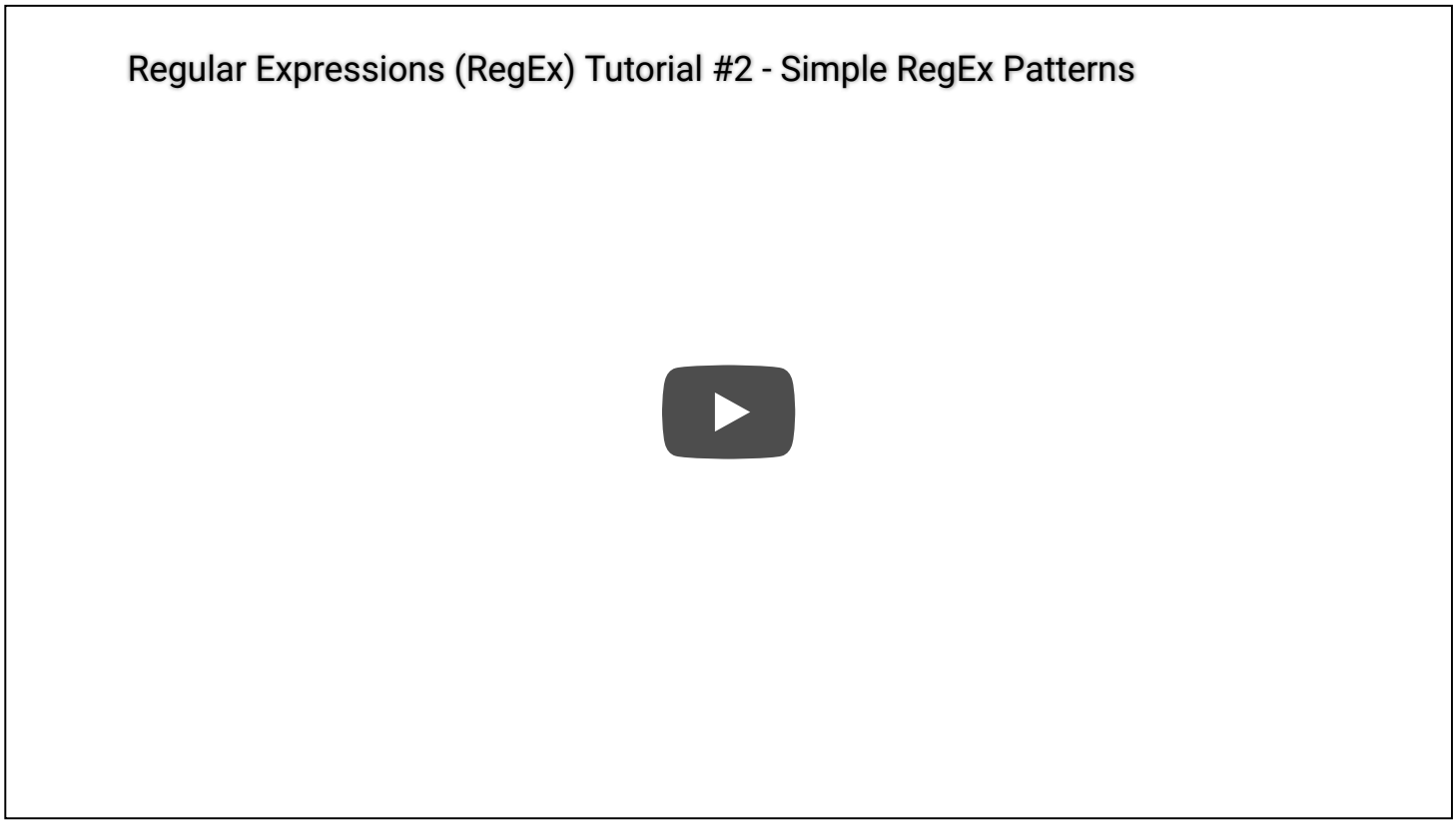
18-02-18T15:45:08.000Z



egular Expressions - 2 - Basic Symbols

ere we take a look more closely at the four basic symbols in a Regular Expression. For the whole series on Regex, check o r playlist: ...

17-06-19T23:17:03.000Z



egular Expressions (RegEx) Tutorial #2 - Simple RegEx Patterns

ey all, in this Regular Expressions tutoriall'll show you an online tool called Regex101 that we'll be using to test our patterns e'll start off slow in this video :).

18-02-13T07:33:26.000Z

← file ownerships and file permisions on Linux and unix like systems (https://www.codexpedia.com/devops/file-ownerships-and-file-permisions-on-linux-and-unix-like-systems/)

installing Ubuntu on Virtual Box → (https://www.codexpedia.com/devops/installing-ubuntu-on-virtual-box/)