Shorthand character classes (Regular Expressions) | JavaScript Tutorial In Hindi #49

Shorthand character classes (Regular Expressions) | JavaScript Tutorial In Hindi #49

As we know, the character ranges are very useful while writing a regular expression. However, they can be hectic to write out every time we want to match common ranges, such as those that designate alphabetical characters or digits. To get rid of this pain, ***shorthand character classes*** represent common ranges, and they make writing regular expressions much simpler. As we have seen from the previous tutorial examples about character set of regular expressions, certain character classes, such as digits [0-9] or characters [0-9A-Za-z_], are used in most regex patterns.

These shorthand classes include:

- **\w:** This is the **"word character"** class that represents the regex range [A-Za-z0-9_], and it will match a *single uppercase character, lower-case character, digit, or underscore*
- **\d:** This is the **"digit character"** class represents the regex range [0-9], and it will match the single-digit character
- **\s:** This is the **"whitespace character"** class that represents the regex range, matching a single space, carriage return, tab, line break, form feed, or vertical tab

Along with shorthand character classes(\w, \d, and \s), we can also use the *negated shorthand character classes*. These negated shorthands will match any character that is NOT in the regular shorthand classes.

These negated shorthand classes include:

- **\W:** the "non-word character" class represents the regex range [^A-Za-z0-9_], matching any character that is not included in the range represented by \w
- **\D**: the "non-digit character" class represents the regex range [^0-9], matching any character that is not included in the range represented by \d
- **\S**: the “non-whitespace character” class represents the regex range [^ \t\r\n\f\v], matching any character that is not included in the range represented by \s

Here is the table of shorthand character classes with examples. This table contains all the information explained above.

Character Classes (Shorthand)

Character	**DESCRIPTION**	**Example**

Character	**DESCRIPTION**	**Example**
\d	This character means any digit character; functionally equivalent to [0-9] or [[:digit:]]	\d matches 1,56,77 12, 123, etc., but not 1b7 or aabb6. One or more of any digit characters.
\D	This character means any non-digit character; functionally equivalent to [^0-9] or [^[:digit:]]	\D matches a, ab, abcd, ab&, but not 1.
\w	This character means any "word" character. That is, any alphanumeric character and its functionally equivalent to [_A-Za-z0-9] or [_[:alnum:]]	\w matches a, ab, a1,abc123 but not !&. One or more upper- or lower-case letters or digits, but not punctuation or other special symbols/characters.
\W	This character means any non-alphanumeric character; functionally equivalent to [^_A-Za-z0-9] or [^_[:alnum:]]	\W matches *, &, but not race or y1. One or more of any character but upper- or lower-case letters and digits.
\s	This character means any white space character; space, new line, tab, non-breaking space, etc.; functionally equivalent to [[:space]]	vegetable\s matches "vegetable" followed by any non-white space character.
\S	This character means any non-whitespace character; anything other than space, newline, tab, non-breaking space, etc.; functionally equivalent to [^[: space]]	vegetable\S matches "vegetable" followed by any non-whitespace character.

Assertions:-

An Assertion is a regular expression that will succeed if a match is found and fails if a match is not found. Assertion consists of Anchors and Lookarounds.

- **^:** The symbol "^" matches at the beginning of the string.
- **$:** The symbol "$" matches only at the end of the string.
- **\b:** The character "\b" matches only at a word boundary.
- **\B:** the character "\B" Matches only if not at a word boundary.
- **(?=«pattern»):** This is a positive lookahead. It matches the regular expression with the pattern only if the pattern matches what comes next. The pattern is used only to look ahead but otherwise ignored.
- **(?!«pattern»):** This is the negative lookahead: It matches the regular expression with the pattern only if the pattern does not match what comes next. The pattern is used only to look ahead but otherwise ignored.

Code as described/written in the video

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
            integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
        <title>Document</title>
    </head>

    <body>
        <div class="container">
            <h1 id="heading" class='yourhead rhia is'> Welcome to Code With Harry</h1>
            <div id="myfirst" class="child red good" id="first">child 1

                <ul class="this" id='myul'>
                    <li class="childul" id='fui'>this</li>
                    <li class="childul">is</li>
                    <li class="childul">a</li>
                    <li class="childul">list </li>
                    <li class="childul" id='lui'>of my dreams</li>
            </div>
            <div class="child">child 2</div>
            <div class="child red">child 3</div>
            <div class="child">child 4</div>
            <form action="none.html" method="post">
                <a href="//codewithharry.com">Go to Code With Harry</a>
                <br>
                <br>
                Search this website: <input type="text" name="Hello" id="">
                <button id="btn">Submit form</button>
                <!-- <input type="button" id='btn' value="submit"> -->
            </form>
        </div>
        <!-- <br>
        <div class="no">this is a dummy div1</div>
        <div class="no">this is a dummy div2</div>
        <div class="no">this is a dummy div3</div> -->
        <div class="container">
            <h1>Student list</h1>
            <ul id="students"></ul>
            <button id="myBtn" class="btn btn-primary">Your Button</button>
            <button class="btn btn-primary">Fetch Data</button>
            <div id="content"></div>
        </div>

        <div class="container my-3">
            <h1>Pull the results by clicking the button below:</h1>

            <button class="btn btn-primary" id="meanings">Get meanings</button>
            <div class="my-2">
                <ul id="defs"></ul>
            </div>
        </div>

        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
            integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
            crossorigin="anonymous"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
            integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
            crossorigin="anonymous"></script>
        <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
            integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
            crossorigin="anonymous"></script>
    </body>
    <!-- <script src="js/tut12.js"></script> -->
    <!-- <script src="js/tut14.js"></script> -->
    <!-- <script src="js/tut15.js"></script> -->
    <!-- <script src="js/tut16.js"></script> -->
    <!-- <script src="js/tut17.js"></script> -->
    <!-- <script src="js/tut18.js"></script> -->
    <!-- <script src="js/tut20.js"></script> -->
    <!-- <script src="js/tut21.js"></script> -->
    <!-- <script src="js/tut23.js"></script> -->
    <!-- <script src="js/tut24.js"></script> -->
    <!-- <script src="js/tut25.js"></script> -->
    <!-- <script src="js/tut27.js"></script> -->
    <!-- <script src="js/tut28.js"></script> -->
    <!-- <script src="js/tut30.js"></script> -->
```

```
<!-- <script src="js/tut31.js"></script> -->
<!-- <script src="js/tut32.js"></script> -->
<!-- <script src="js/tut34.js"></script> -->
<!-- <script src="js/tut37.js"></script> -->
<!-- <script src="js/tut39.js"></script> -->
<!-- <script src="js/tut39b.js"></script> -->
<!-- <script src="js/tut41.js"></script> -->
<!-- <script src="js/tut43.js"></script> -->
<!-- <script src="js/tut44.js"></script> -->
<!-- <script src="js/tut 45.js"></script> -->
<!-- <script src="js/tut46.js"></script> -->
<!-- <script src="js/tut47.js"></script> -->
<!-- <script src="js/tut48.js"></script> -->
<script src="js/tut49.js"></script>

</html>
```

Code as described/written in the video

```
console.log("This is tutorial 49");

// Character classes
let regex = /\war/;     //word character - _ or alphabet or numbers
regex = /\w+d1r/;       // \w+ means one or more word characters
regex = /\Wbhai/;       // Non word character
regex = /\W+bhai/;      // \W+ means more than one Non word character
regex = /number \d999/; // \d means digit
regex = /number \d+/;   // \d+ means more than one digit
regex = /\D999/;        // \D means non digit
regex = /\D+999/;       // \D+ means more than one non digit
regex = /\ska number/;  // Match whitespace character
regex = /\s+ka number/; // \s+ means match one or more than one whitespace characters
regex = /\Ska number/;  // Match non whitespace character
regex = /\S+ka number/; // Match one or more than one non whitespace character
regex = /4r5r\b/;  // word boundary

// Assertions
regex = /h(?=y)/;
regex = /h(?!y)/;
str = "harh7rd1r4r5ry%%$@bhai hdrryika number 899999harry9999";



let result = regex.exec(str);
console.log("The result from exec is ", result);

if(regex.test(str)){
    console.log(`The string ${str} matches the expression ${regex.source}`);
}
else{
    console.log(`The string ${str} does not match the expression ${regex.source}`);
}
```

Previous   Next

CodeWithHarry