

UNIT-5 : Regular Expression, Rollover and Frames

- **Regular Expression**

A regular expression (often abbreviated as regex or regexp) in JavaScript is a sequence of characters that forms a search pattern. It's used for pattern matching within strings, enabling powerful search, replace, and validation operations.

- **Commonly Used Flags**

g: Global search (find all matches, not just the first).

i: Case-insensitive search.

m: Multiline search.

- **Basic Regex Patterns**

.: Matches any character except a newline.

*: Matches 0 or more occurrences of the preceding element.

+: Matches 1 or more occurrences of the preceding element.

?: Matches 0 or 1 occurrence of the preceding element.

^: Matches the beginning of a string.

\$: Matches the end of a string.

[]: Character set. Matches any one of the enclosed characters.

1. **test() Method**

Purpose: The test() method checks if a pattern exists within a string. It returns a boolean (true or false).

Syntax:

```
regex.test(string);
```

2. **match() Method**

Purpose: The match() method searches a string for matches of the pattern. It returns an array of results or null if no match is found.

Syntax:

```
string.match(regex);
```

3. search() Method

Purpose: The search() method looks for a match within a string and returns the index of the first match. If no match is found, it returns -1.

Syntax:

```
string.search(regex);
```

4. exec() Method

Purpose: The exec() method executes a search for a match in a string. It returns an array of matched information (with details like the matched text and its index) or null if no match is found. This method is especially useful when working with capturing groups.

Syntax:

```
regex.exec(string);
```

- **Example of test():**

```
<!DOCTYPE html>
```

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

```
<head>
```

```
  <meta charset="UTF-8">
```

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

```
  <title>Document</title>
```

```
</head>
```

```
<body>
```

```
  <script>
```

```
    function test1() {
```

```
var s = document.getElementById("t1").value;
var p = /[0-9]{10}$/;
if(p.test(s)) {
    document.getElementById("p").innerHTML = "Mobile Number is valid!";
} else {
    document.getElementById("p").innerHTML = "Mobile Number is invalid!";
}
}
```

```
function test2() {
    var s = document.getElementById("t2").value;
    var p = /91+\s[0-9]{10}$/;
    if(p.test(s)) {
        document.getElementById("p").innerHTML = "Mobile Number is valid!";
    } else {
        document.getElementById("p").innerHTML = "Mobile Number is invalid!";
    }
}
```

```
function test3() {
    var s = document.getElementById("t3").value;
    var p = /[([91])]\s[0-9]{10}$/;
    if(p.test(s)) {
        document.getElementById("p").innerHTML = "Mobile Number is valid!";
    } else {
        document.getElementById("p").innerHTML = "Mobile Number is invalid!";
    }
}
```

```

</script>

<h1>Mobile Number</h1>

<input type="text" id="t1">

<button onclick="test1()">Check</button><br>

<input type="text" id="t2">

<button onclick="test2()">Check</button><br>

<input type="text" id="t3">

<button onclick="test3()">Check</button><br>

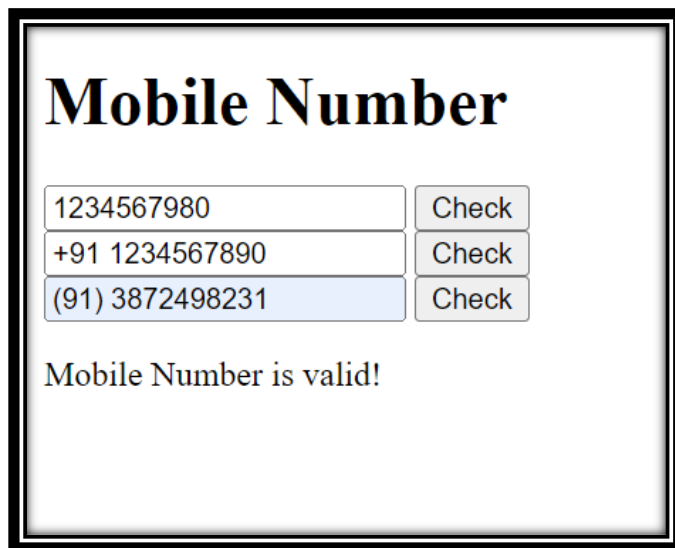
<p id="p">Check Here</p>

</body>

</html>

```

- **Output:**



The screenshot shows a web form with a title "Mobile Number". It contains three input fields, each followed by a "Check" button. The first input field contains "1234567980". The second input field contains "+91 1234567890". The third input field contains "(91) 3872498231" and is highlighted with a blue background. Below the input fields, a message states "Mobile Number is valid!".

- **Example of search():**

```

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```
<title>Document</title>
</head>
<body>
  <script>
    function searching() {
      var s = document.getElementById("t1").value;
      var p = /js/i;
      if(s.search(p) == -1) {
        document.getElementById("p").innerHTML = "value does not exist!";
      } else {
        document.getElementById("p").innerHTML = "value exist!";
      }
    }
  </script>
  <h1>Searching Values</h1>
  <input type="text" id="t1">
  <button onclick="searching()">Search</button><br>
  <p id="p">Check Here</p>
</body>
</html>
```

- **Output:**



- **Example of match():**

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

</head>

<body>

  <script>

    function matching() {

      var s = document.getElementById("t1").value;

      var p = /css/i;

      var r = s.match(p);

      document.getElementById("p").innerHTML = `Matching values are: ${r}`;

    }

  </script>

  <h1>Matching Values</h1>

  <input type="text" id="t1">

  <button onclick="matching()">Match</button><br>
```

```
<p id="p">Check Here</p>
</body>
</html>
```

- **Output:**

Matching Values

Matching values are: css

- **Frames**

Frames were commonly used to create web layouts that divided a page into different sections (e.g., a navigation bar in one frame and content in another). This was achieved using the `<frameset>` and `<frame>` elements.

1. **`<frameset>`:**

Replaces the `<body>` tag and defines a layout that divides the window into columns or rows.

2. **`<frame>`:**

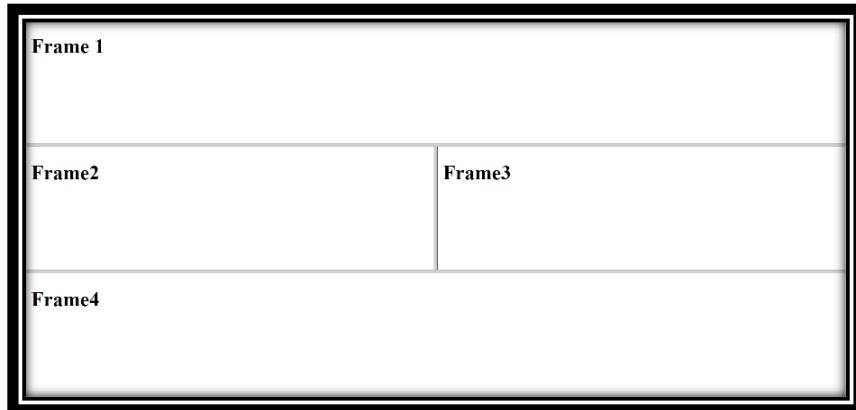
Specifies each individual frame and the HTML content it will display.

- **Example:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <frameset rows="33%, 33%, 34%" >
    <frame src="f1.html" >
      <frameset cols="50%, 50%" >
        <frame src="f2.html" >
        <frame src="f3.html" >
      </frameset>
    <frame src="f4.html" >
  </frameset>
</head>
<body>

</body>
</html>
```


- **Output:**



3. **<iframe>:**

Instead of using frames, the `<iframe>` element is used in modern HTML to embed one HTML document within another.

- **Example:**

```
<!DOCTYPE html>
```

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

```
<head>
```

```
  <meta charset="UTF-8">
```

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

```
  <title>Table Frame</title>
```

```
</head>
```

```
<body>
```

```
  <table border="1">
```

```
    <thead>
```

```
      <th colspan="2">Frames</th>
```

```
    </thead>
```

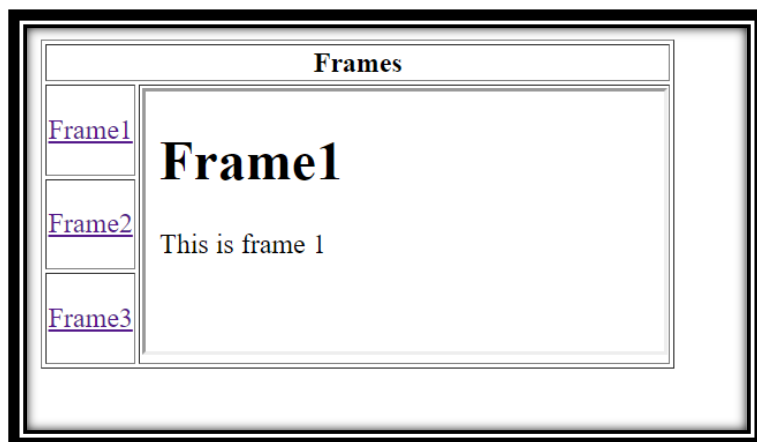
```
    <tbody>
```

```

<tr>
  <td><a href="frame1.html" target="mainframe">Frame1</a></td>
  <td rowspan="3">
    <iframe name="mainframe"></iframe>
  </td>
</tr>
<tr>
  <td><a href="frame2.html" target="mainframe">Frame2</a></td>
</tr>
<tr>
  <td><a href="frame3.html" target="mainframe">Frame3</a></td>
</tr>
</tbody>
</table>
</body>
</html>

```

- **Output:**



- **Rollover**

To create a rollover effect using the JavaScript events onmouseover and onmouseout, you can swap images, change colors, or trigger other changes when a user hovers over an element.

Syntax Overview:

- onmouseover: This event is triggered when the mouse pointer moves over an element.
- onmouseout: This event is triggered when the mouse pointer leaves the element.

- **Example:**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Image Rollover</title>
</head>
<body>
  
</body>
</html>
```

- **Output:**

TRANQUILITY	LOVE	HEALTH	CONFIDENCE	NATURE
AUTHORITY	DETACHMENT	HAPPINESS	CONFIDENCE	DEATH
WISDOM	WARMTH	PROFUNDITY	ENTHUSIASM	PROSPERITY
STABILITY	ROMANCE	ENTHUSIASM	SARCASTIC	HEALTH
CLEANLINESS	PASSION	ENERGETIC	OPTIMISM	HOPE
FRESHNESS	SPEED	YOUTH	WARMTH	LUCK
FREEDOM	LUCK	FUN	ZEN	LIFE
COLD	FACE	RUN	ILLNESS	DENY
SADNESS	BLOOD	DANGER	DANGER	POISON
DEPRESSION	AGGRESSION	DESOLATION	WISDOM	CORRUPTION
FINANCE	LUXURY	LUXURY	LIGHT	STRENGTH
PLAYING	MYSTERY	DARKNESS	WISDOM	CALM
INNOCENCE	SPIRITUALITY	SOPHISTICATION	CLEANLINESS	TIMELESSNESS
DELICATE	ATTRACTION	AUTHORITY	SPIRITUALITY	NEUTRALITY
PLAYFUL	FUTURE	ELEGANCE	INNOCENCE	AUTHORITY
SWEET	ROYALTY	MYSTERY	PURITY	WISDOM
KIND	MAGIC	POWER	HOPE	STABILITY
IMMATURE	ILLUSION	FEAR	COLD	SUFL
DECEPTION	DECEPTION	LONGINESS	ISOLATION	LIFELESS
MATERIALISM	DETACHMENT	HOPELESSNESS	EMPTINESS	ABANDONMENT