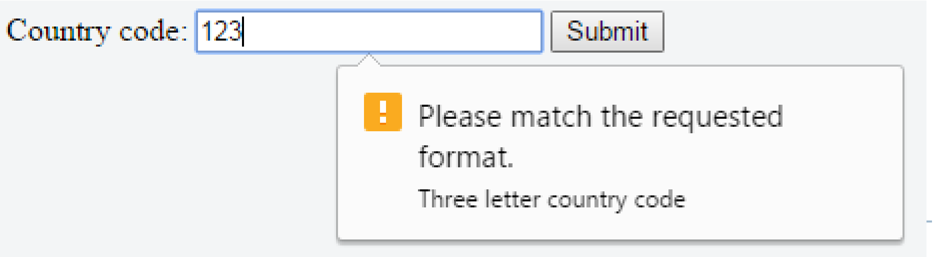
Pattern and regular expressions

The **pattern** attribute specifies a regular expression that the **<input>** element's value is checked against. The **pattern** uses the ECMAScript (i.e. JavaScript) flavor of regex.

Note: The **pattern** attribute works with the following input types: **text**, **date**, **search**, **url**, **tel**, **email**, and **password**.

Tip: Use the global **title** attribute to describe the pattern to help the user.

```
<form action="demo_form.jsp">
 Country code:
 <input type="text" name="country_code" pattern="[A-Za-z]{3}" title=" Three letter country code ">
 <input type="submit">
</form>
```



Country code: 123

Please match the requested format.

Three letter country code

Regular expressions

```
^[a-zA-Z_\-]+@(([a-zA-Z_\-])+\.)+[a-zA-Z]{2,4}$
```

- Test whether a string matches the expression's pattern
- powerful but tough to read
  - (the above regular expression matches email addresses)
- used in all languages:
  - Java, PHP ,JavaScript, HTML, C#, and other languages
- Many IDEs allow regexes in search/replace

Basic regular expressions

The simplest regexes simply matches any string that contains that text.

```
abc
```

above regular expressio**n** matches any string containing "abc“:

- YES: "abc", "abcdef", "defabc", ".=.abc.=.", ...
- NO: " ABC" , " fedcba", "ab c", "PHP", ...

- Note that html5 has implicit anchors ^ and $, so abc is really ^abc$
- Regular expressions are case-sensitive by default.

Wildcards

A dot `.` matches exactly **one-character** except a `\n` line break

`.oo.y` matches "Doocy", "goofy", "LooNy", ...

Special characters: |, (), \

| means OR

`abc|def|g` matches "abc", "def", or "g"

() are for grouping

`(Homer|Marge) Simpson`

matches "Homer Simpson" or "Marge Simpson"

\ escapes a special character

many characters must be escaped to match them literally: / \ $. [ ] ( ) ^ * + ?

`<br\/>` matches lines containing <br /> tags

Quantifiers: *, +, ?

`*` means 0 or more occurrences

- `abc*` matches "ab", "abc", "abcc", "abccc", ...
- `a(bc)*` matches "a", "abc", "abcbc", "abcbcbc", ...
- `a.*a` matches "aa", "aba", "a8qa", "a!?xyz__9a", ...

`+` means 1 or more occurrences

- `a(bc)+` matches "abc", "abcbc", "abcbcbc", ...
- `Goo+gle` matches "Google", "Gooogle", "Goooogle", ...

`?` means 0 or 1 occurrences

- `a(bc)?` matches "a" or "abc"

Anchors: ^ and $

^ represents the beginning of the string or line;

$ represents the end

Jess matches all strings that contain Jess;

^Jess matches all strings that start with Jess;

Jess$ matches all strings that end with Jess;

^Jess$ matches the exact string "Jess" only

^Mart.*Stepp$ matches "MartStepp", "Marty Stepp", "Martin D Stepp", ... but NOT "Marty Stepp stinks" or "I H8 Martin Stepp"

The html5 spec states that ^ and $ are implicit

Character sets: []

[] group characters into a character set, will match any **single character** from the set

- **[bcd]art** matches strings containing "bart", "cart", and "dart"
- equivalent to **(b|c|d)art** but shorter

inside [], many of the modifier keys act as normal characters

- **what[!*?]*** matches "what", "what!", "what?**!", "what??!", ...
- What regular expression matches DNA (strings of A, C, G, or T)?
  [ACGT]+

Character ranges: [start-end]

- inside a character set, specify a range of characters with -
  - `[a-z]` matches any lowercase letter
  - `[a-zA-Z0-9]` matches any lower- or uppercase letter or digit
- an initial `^` inside a character set negates it
  - `[^abcd]` matches any character other than a, b, c, or d
- inside a character set, - must be escaped to be matched
  - `[+\-]?[0-9]+` matches an optional + or -, followed by at least one digit

What regular expression matches letter grades such as A, B+, or D- ?

`[ABCDF][+\-]?`

Escape sequences

Special escape sequence character sets:

`\d` matches any digit (same as [0-9])

`\D` any non-digit ([^0-9])

`\w` matches any word character (same as [a-zA-Z_0-9])

`\W` any non-word char

`\s` matches any whitespace character (, \t, \n, etc.)

`\S` any non-whitespace

b is a zero-width match of a word boundary. (Either start of end of a word, where "word" is defined as \w+) Note: "zero width" **means** if the **b** is within a **regex** that matches, it **does** not add any characters to the text captured by that match.

What regular expression matches dollar amounts of at least $100.00 ?

```
\$[1-9]\d{2,}\.\d{2}
```

Creating a Regular Expression in JS

- Creating a regular expression
  - Using a regular expression literal

```
let exp = /ab+c/;
```

  - Or calling the constructor functions

```
let exp = new RegExp('ab+c');
```

Using RegEx in JS

- Regular expressions are used with the RegExp methods `test()` and `exec()` and with the String methods `match()`, `replace()`, `search()`, and `split()`.

```
const str = "1 First 2 Second 3 Third 100 last";
const exp = /\d+/g;
const arr = str.replace(exp, "");
console.log(arr);
```