

US date (MM/DD/YYYY)

Slash-separated US date format.

```
^(0[1-9]|1[0-2])/(0[1-9]|[12]\d|3[01])/\d{4}$
```

Token-by-token breakdown

Token	Meaning
^	start of string (or line in multiline mode)
(	start of a capturing group
0	literal text "0"
[1-9]	any of: 1–9
	OR — try the alternative
1	literal text "1"
[0-2]	any of: 0–2
)	end of group
/	literal text "/"
(	start of a capturing group
0	literal text "0"
[1-9]	any of: 1–9
	OR — try the alternative
[12]	any of: "1", "2"
\d	any digit (0–9)
	OR — try the alternative
3	literal text "3"
[01]	any of: "0", "1"
)	end of group
/	literal text "/"
\d{4}	exactly 4 times: digit (0–9)
\$	end of string (or line in multiline mode)

Quick usage in 7 languages

JavaScript `const re = /^(0[1-9]|1[0-2])\/(0[1-9]|[12]\d|3[01])\/\d{4}$/;
re.test(value);`

Python `import re
re.match(r"^(0[1-9]|1[0-2])\/(0[1-9]|[12]\d|3[01])\/\d{4}$", value)`

Java `Pattern.compile("(0[1-9]|1[0-2])\/(0[1-9]|[12]\d|3[01])\/\d{4}$").matcher(value).match
es();`

C# / .NET	<code>Regex.IsMatch(value, @"^(0[1-9] 1[0-2])/(0[1-9] [12]\d 3[01])/\d{4}\$");</code>
Go	<code>regexp.MustCompile(`^(0[1-9] 1[0-2])/(0[1-9] [12]\d 3[01])/\d{4}\$`).MatchString(value)</code>
Ruby	<code>/^(0[1-9] 1[0-2])\/(0[1-9] [12]\d 3[01])\/\d{4}\$/ .match?(value)</code>
PHP	<code>preg_match('~^(0[1-9] 1[0-2])/(0[1-9] [12]\d 3[01])/\d{4}\$~', \$value);</code>

Common pitfalls

- **Anchored to the whole string.** This pattern uses ^ and \$, so it requires the entire input to match. To find it inside a longer text, drop the anchors and use the global (g) flag.
- **Escape it correctly per language.** In Java and JavaScript strings each backslash must be doubled (\d); in Python, Go, and C# use raw/verbatim strings so the backslashes survive.
- **Validate beyond format.** Matching the format doesn't guarantee the value is real. Confirm the us date (mm/dd/yyyy) against a source of truth (database, API, or checksum) where it matters.