HTML tag

Match HTML opening, closing, or self-closing tags.

```
<\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\/?>
```

Token-by-token breakdown

Token	Meaning
`<`	literal text “<”
`\/?`	optional (zero or one): literal “/”
`[a-zA-Z]`	any of: lowercase letters, uppercase letters
`[a-zA-Z0-9]*`	zero or more: any of: lowercase letters, uppercase letters, digits
`(`	start of a capturing group
`\s+`	one or more: whitespace
`[a-zA-Z-]+`	one or more: any of: lowercase letters, uppercase letters, “-”
`(`	start of a capturing group
`\s*`	zero or more: whitespace
`=`	literal text “=”
`\s*`	zero or more: whitespace
`(`	start of a capturing group
`"`	literal text “"”
`[^"]*`	zero or more: any character except “"”
`"`	literal text “"”
`|`	OR — try the alternative
`\'`	literal “'”
`[^\']*`	zero or more: any character except “'”
`\'`	literal “'”
`|`	OR — try the alternative
`[^\s>]+`	one or more: any character except whitespace, “>”
`)`	end of group
`)?`	end of group, optional (zero or one)
`)*`	end of group, zero or more
`\s*`	zero or more: whitespace
`\/?`	optional (zero or one): literal “/”
`>`	literal text “>”

Quick usage in 7 languages

JavaScript	`const re = /<\\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\\/?>/;` `re.test(value);`
Python	`import re` `re.match(r"""<\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\/?>""", value)`
Java	`Pattern.compile("<\\/?[a-zA-Z][a-zA-Z0-9]*(\\s+[a-zA-Z-]+(\\s*=\\s*(\"[^\"]*\"|\\'[^\\']*\\'|[^\\s>]+))?)*\\s*\\/?>").matcher(value).matches();`
C# / .NET	`Regex.IsMatch(value, @"<\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*(""[^""]*""|\'[^\']*\'|[^\s>]+))?)*\s*\/?>");`
Go	`regexp.MustCompile(`<\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\/?>`).MatchString(value)`
Ruby	`/<\\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\\/?>/.match?(value)`
PHP	`preg_match('~<\/?[a-zA-Z][a-zA-Z0-9]*(\s+[a-zA-Z-]+(\s*=\s*("[^"]*"|\'[^\']*\'|[^\s>]+))?)*\s*\/?>~', $value);`

Common pitfalls

• **Not anchored.** Without ^ and $ this can match a substring anywhere in the input — add anchors if you need the whole value to conform.

• **ASCII letters only.** [a-zA-Z] excludes accented and non-Latin letters (é, ü, ß, ñ, and non-Latin scripts). For international input use Unicode properties like \p{L} with the u flag.

• **Watch for backtracking.** This pattern nests quantifiers; on adversarial input that can cause exponential backtracking (ReDoS). Test with long non-matching strings, or use an RE2-based engine.

• **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 html tag against a source of truth (database, API, or checksum) where it matters.