



CSS font variant attribute

The `font-variant-*` properties are used to control the appearance of text by enabling or disabling OpenType features available on the font.

Not all fonts have the same set of OpenType Features available so it is possible that a given `font-variant-*` property will produce no results or produce a different result to what you intend.

One of the things that makes working with variable fonts is the uneven support of `font-variant-*` attributes between browsers. That's why we have to use `font-feature-settings` and reset the rule every time we make a change.

To see what features are available in a font you can use tools like Wakmaifondue as described in [Analyzing a font with Wakmaifondue](#)

This post will look the different values for the different `font-variant-*` properties, work on understanding what they do and the `font-variant` shorthand.

The sections below listing `font-variant-*` properties are listed in no particular order. It is also important to note that these properties are not supported in all fonts.

font-variant-numeric

[font-variant-numeric](#) controls the usage of numeric forms (numbers, fractions, and ordinal markers) in fonts.

The property takes either the value `normal` on its own or one or more of the other values defined below:

`normal` : This keyword leads to the deactivation of the use of such alternate glyphs.

`ordinal` : This keyword forces the use of special glyphs for the ordinal markers, like 1st, 2nd, 3rd, 4th in English or a 1a in Italian. It corresponds to the OpenType values `ordn`

`slashed-zero` : This keyword forces the use of a 0 with a slash; this is useful

when a clear distinction between O and 0 is needed. It corresponds to the OpenType values zero

`<numeric-figure-values>` : These values control the figures used for numbers. Two values are possible:

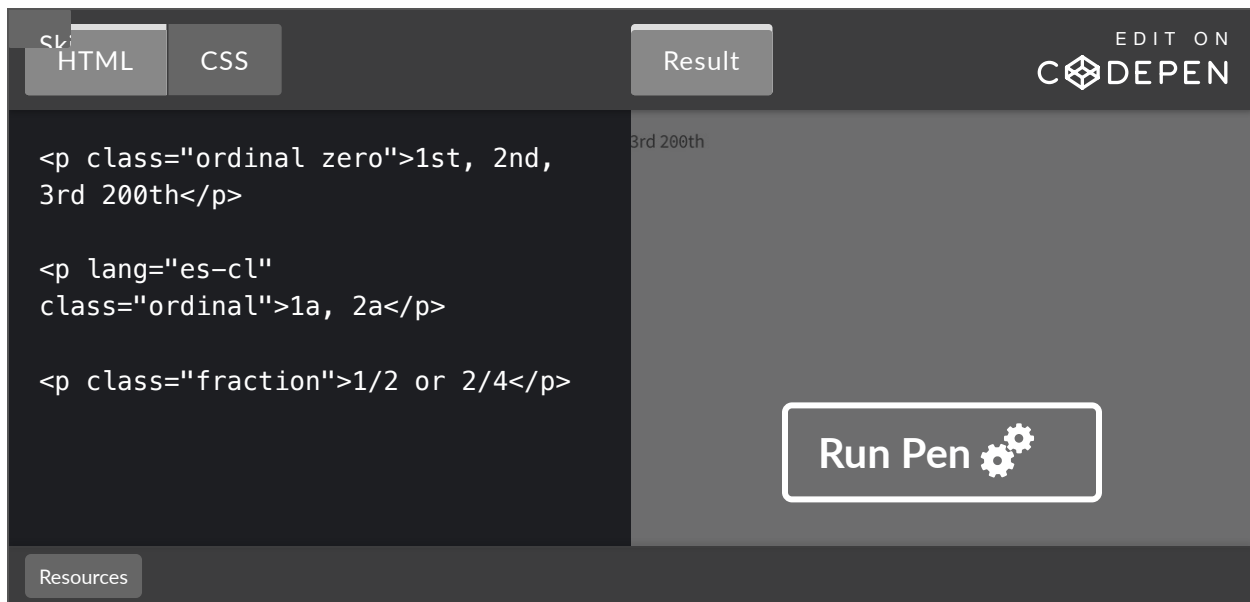
- `lining-nums` activating the set of figures where numbers are all lying on the baseline. It corresponds to the OpenType values `lnum`
- `oldstyle-nums` activating the set of figures where some numbers, like 3, 4, 7, 9 have descenders. It corresponds to the OpenType values `onum`

`<numeric-spacing-values>` : These values controls the sizing of figures used for numbers. Two values are possible:

- `proportional-nums` activating the set of figures where numbers are not all of the same size. It corresponds to the OpenType values `pnum`
- `tabular-nums` activating the set of figures where numbers are all of the same size, allowing them to be easily aligned like in tables. It corresponds to the OpenType values `tnum`

`numeric-fraction-values` : These values controls the glyphs used to display fractions. Two values are possible:

- `diagonal-fractions` activating the set of figures where the numerator and denominator are made smaller and separated by a slash. It corresponds to the OpenType values `frac`
- `stacked-fractions` activating the set of figures where the numerator and denominator are made smaller, stacked and separated by a horizontal line. It corresponds to the OpenType values `afrc`



font-variant-east-asian

[font-variant-east-asian](#) controls the use of alternate glyphs for East Asian scripts, like Japanese and Chinese.

Possible values are:

normal : This keyword leads to the deactivation of the use of such alternate glyphs.

ruby : This keyword forces the use of special glyphs for ruby characters. As these are usually smaller, font creators often designs specific forms, usually slightly bolder to improve the contrast. This keyword corresponds to the OpenType values ruby

<east-asian-variant-values : These values specify a set of glyph variants which should be used for display. Possible values are listed in the syntax section of the [MDN font-variant-eastasian article](#) under `east-asian-variant-values

east-asian-width-values : These values control the sizing of figures used for East Asian characters. Two values are possible:

- **proportional-width** activates the set of East Asian characters which vary in width. It corresponds to the OpenType values **pwid**
- **full-width** activates the set of East Asian characters which are all of the same, roughly square, width metric. It corresponds to the OpenType values **fwid**

font-variant-caps

[font-variant-caps](#) handles alternate glyphs for capital letters.

When a given font includes capital letter glyphs of multiple different sizes, this property selects the most appropriate ones. If petite capital glyphs are not available, they are rendered using small capital glyphs. If these are not present, the browser synthesizes them from the uppercase glyphs.

Using all uppercase letters in large blocks of text may have accessibility implications for people with Dyslexia and other reading disabilities.

The possible values for the rule are: `normal` : Deactivates of the use of alternate glyphs.

`small-caps` : Enables display of small capitals (OpenType feature: `smcp`). Small-caps glyphs typically use the form of uppercase letters but are reduced to the size of lowercase letters.

`all-small-caps` : Enables display of small capitals for both upper and lowercase letters (OpenType features: `c2sc`, `smcp`).

`petite-caps` : Enables display of petite capitals (OpenType feature: `pcap`).

`all-petite-caps` : Enables display of petite capitals for both upper and lowercase letters (OpenType features: `c2pc`, `pcap`).

`unicase` : Enables display of mixture of small capitals for uppercase letters with normal lowercase letters (OpenType feature: `unic`).

`titling-caps` : Enables display of titling capitals (OpenType feature: `titl`). Uppercase letter glyphs are often designed for use with lowercase letters. When used in all uppercase titling sequences they can appear too strong

font-variant-ligatures

[font-variant-ligatures](#) enables or disables alternate glyphs for [ligatures](#) if the font has any available.

What ligatures the font supports will dictate if the different values for the property will provide the desired effect or if they will provide any effect at all.

The possible values are:

`normal` (**default**) : This keyword activates ligatures and contextual forms needed for correct rendering. The font, language and type of script control what gets activated

`none` : This keyword deactivates all ligatures and contextual forms are disabled, even common ones


`<common-lig-values>` : These values control the most common ligatures, like for **fi**, **ffi**, **th** and others. They correspond to the OpenType values `liga` and `clig`. Two values are possible:

- `common-ligatures` activates these ligatures. **The keyword `normal` also activates these ligatures**
- `no-common-ligatures` deactivates the ligatures

`<discretionary-lig-values>` : These values control ligatures specific to the font and defined by the type designer. They correspond to the OpenType values

dlig. Two values are possible:

- discretionary-ligatures activates these ligatures
- no-discretionary-ligatures deactivates the ligatures. **The keyword normal usually deactivates these ligatures**

<historical-lig-values> : These values control the ligatures used historically, in old books, like the German tz digraph being displayed as . They correspond to the OpenType values hlig. Two values are possible:

- historical-ligatures activates the ligatures
- no-historical-ligatures deactivates the ligatures. **The keyword normal usually deactivates these ligatures**

<contextual-alt-values> : These values control whether letters adapt to the surrounding letters. These values correspond to the OpenType value calt. Two values are possible:

- contextual specifies that the contextual alternates are to be used **The keyword normal usually activates these ligatures too**
- no-contextual prevents their use

font-variant-alternates

Note:

Currently, `font-variant-alternates` only works in Firefox and Safari (desktop and iOS/iPadOS)

Chromium browsers have equivalent support under `font-feature-settings`.

[font-variant-alternates](#) controls the usage of alternate glyphs. This `font-variant-*` property is tightly coupled with the font you use as it's almost certain that the results will vary from font to font.

The possible values for the property are the keyword `normal` or one or more of the values below:

`historical-forms` : This keyword enables historical forms — glyphs that were common in the past but not today. It corresponds to the OpenType value `hist`

`stylistic()` : This function enables stylistic alternates for individual characters. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `salt`, like `salt 2`

`styleset()` : This function enables stylistic alternatives for sets of characters. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `ssXY`, like `ss02`

`character-variant()` : This function enables specific stylistic alternatives for characters. It is similar to `styleset()`, but doesn't create coherent glyphs for a set of characters; individual characters will have independent and not necessarily coherent styles. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `cvXY`, like `cv02`

`swash()` : This function enables [swash](#) glyphs. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType values `swsh` and `cswh`, like `swsh 2` and `cswh 2`

`ornaments()` : This function enables ornaments, like [fleurons](#) and other dingbat glyphs. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `ornm`, like `ornm 2`

`annotation()` : This function enables annotations, like circled digits or inverted characters. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `na1t`, like `na1t 2`

The font-variant shorthand

[font-variant](#) is the shorthand for the properties we've discussed so far.

The shorthand makes it easier to set all the necessary properties in one place so that, if we need to change them, we only need to do it in one place.

The values for `font-variant` are highly dependent on the font you use. Run the font through an analysis tool or work with the foundry you purchased it from to see what variants the font supports and whether it meets your needs or not.