> Menu

NEXT.JS

Font Module

This API reference will help you understand how to use `next/font/google` and `next/font/local`. For features and usage, please see the Optimizing Fonts page.

Font Function Arguments

For usage, review Google Fonts and Local Fonts.

Key	`font/google`	`font/local`	Type	Required
`src`	✗	✓	String or Array of Objects	Yes
`weight`	✓	✓	String or Array	Required/Optional
`style`	✓	✓	String or Array	-
`subsets`	✓	✗	Array of Strings	-
`axes`	✓	✗	Array of Strings	-
`display`	✓	✓	String	-
`preload`	✓	✓	Boolean	-
`fallback`	✓	✓	Array of Strings	-
`adjustFontFallback`	✓	✓	Boolean or String	-
`variable`	✓	✓	String	-
`declarations`	✗	✓	Array of Objects	-

`src`

The path of the font file as a string or an array of objects (with type `Array<{path: string, weight?: string, style?: string}>`) relative to the directory where the font loader function is called.

Used in `next/font/local`

- Required

Examples:

- `src:'./fonts/my-font.woff2'` where `my-font.woff2` is placed in a directory named `fonts` inside the `app` directory
- `src:[{path: './inter/Inter-Thin.ttf', weight: '100',},{path: './inter/Inter-Regular.ttf',weight: '400',},{path: './inter/Inter-Bold-Italic.ttf', weight: '700',style: 'italic',},]`
- if the font loader function is called in `app/page.tsx` using `src:'../styles/fonts/my-font.ttf'`, then `my-font.ttf` is placed in `styles/fonts` at the root of the project

`weight`

The font `weight` ↗ with the following possibilities:

- A string with possible values of the weights available for the specific font or a range of values if it's a variable ↗ font
- An array of weight values if the font is not a variable google font ↗. It applies to `next/font/google` only.

Used in `next/font/google` and `next/font/local`

- Required if the font being used is **not** variable ↗

Examples:

- `weight: '400'`: A string for a single weight value - for the font `Inter` ↗, the possible values are `'100'`, `'200'`, `'300'`, `'400'`, `'500'`, `'600'`, `'700'`, `'800'`, `'900'` or

`'variable'` where `'variable'` is the default)

- `weight: '100 900'`: A string for the range between `100` and `900` for a variable font
- `weight: ['100','400','900']`: An array of 3 possible values for a non variable font

`style`

The font `style` with the following possibilities:

- A string value with default value of `'normal'`
- An array of style values if the font is not a variable google font. It applies to `next/font/google` only.

Used in `next/font/google` and `next/font/local`

- Optional

Examples:

- `style: 'italic'`: A string - it can be `normal` or `italic` for `next/font/google`
- `style: 'oblique'`: A string - it can take any value for `next/font/local` but is expected to come from standard font styles
- `style: ['italic','normal']`: An array of 2 values for `next/font/google` - the values are from `normal` and `italic`

`subsets`

The font `subsets` defined by an array of string values with the names of each subset you would like to be preloaded. Fonts specified via `subsets` will have a link preload tag injected into the head when the `preload` option is true, which is the default.

Used in `next/font/google`

- Optional

Examples:

- `subsets: ['latin']`: An array with the subset `latin`

You can find a list of all subsets on the Google Fonts page for your font.

`axes`

Some variable fonts have extra `axes` that can be included. By default, only the font weight is included to keep the file size down. The possible values of `axes` depend on the specific font.

Used in `next/font/google`

- Optional

Examples:

- `axes: ['slnt']`: An array with value `slnt` for the `Inter` variable font which has `slnt` as additional `axes` as shown here↗. You can find the possible `axes` values for your font by using the filter on the Google variable fonts page↗ and looking for axes other than `wght`

`display`

The font `display`↗ with possible string values↗ of `'auto'`, `'block'`, `'swap'`, `'fallback'` or `'optional'` with default value of `'swap'`.

Used in `next/font/google` and `next/font/local`

- Optional

Examples:

- `display: 'optional'`: A string assigned to the `optional` value

`preload`

A boolean value that specifies whether the font should be preloaded or not. The default is `true`.

Used in `next/font/google` and `next/font/local`

- Optional

Examples:

- `preload: false`

`fallback`

The fallback font to use if the font cannot be loaded. An array of strings of fallback fonts with no default.

- Optional

Used in `next/font/google` and `next/font/local`

Examples:

- `fallback: ['system-ui', 'arial']`: An array setting the fallback fonts to `system-ui` or `arial`

`adjustFontFallback`

- For `next/font/google`: A boolean value that sets whether an automatic fallback font should be used to reduce Cumulative Layout Shift↗. The default is `true`.
- For `next/font/local`: A string or boolean `false` value that sets whether an automatic fallback font should be used to reduce Cumulative Layout Shift↗. The possible values are `'Arial'`, `'Times New Roman'` or `false`. The default is `'Arial'`.

Used in `next/font/google` and `next/font/local`

- Optional

Examples:

- `adjustFontFallback: false`: for `next/font/google`
- `adjustFontFallback: 'Times New Roman'`: for `next/font/local`

`variable`

A string value to define the CSS variable name to be used if the style is applied with the CSS variable method.

Used in `next/font/google` and `next/font/local`

- Optional

Examples:

- `variable: '--my-font'`: The CSS variable `--my-font` is declared

`declarations`

An array of font face descriptor ↗ key-value pairs that define the generated `@font-face` further.

Used in `next/font/local`

- Optional

Examples:

- `declarations: [{ prop: 'ascent-override', value: '90%' }]`

Applying Styles

You can apply the font styles in three ways:

- `className`
- `style`
- CSS Variables

`className`

Returns a read-only CSS `className` for the loaded font to be passed to an HTML element.

```
<p className={inter.className}>Hello, Next.js!</p>
```

`style`

Returns a read-only CSS `style` object for the loaded font to be passed to an HTML element, including `style.fontFamily` to access the font family name and fallback fonts.

```
<p style={inter.style}>Hello World</p>
```

CSS Variables

If you would like to set your styles in an external style sheet and specify additional options there, use the CSS variable method.

In addition to importing the font, also import the CSS file where the CSS variable is defined and set the variable option of the font loader object as follows:

app/page.tsx TypeScript

```tsx
import { Inter } from 'next/font/google'
import styles from '../styles/component.module.css'

const inter = Inter({
  variable: '--font-inter',
})
```

To use the font, set the `className` of the parent container of the text you would like to style to the font loader's `variable` value and the `className` of the text to the `styles` property from the external CSS file.

app/page.tsx TypeScript

```tsx
<main className={inter.variable}>
```

```
  <p className={styles.text}>Hello World</p>
</main>
```

Define the `text` selector class in the `component.module.css` CSS file as follows:

styles/component.module.css

```
.text {
  font-family: var(--font-inter);
  font-weight: 200;
  font-style: italic;
}
```

In the example above, the text `Hello World` is styled using the `Inter` font and the generated font fallback with `font-weight: 200` and `font-style: italic`.

Using a font definitions file

Every time you call the `localFont` or Google font function, that font will be hosted as one instance in your application. Therefore, if you need to use the same font in multiple places, you should load it in one place and import the related font object where you need it. This is done using a font definitions file.

For example, create a `fonts.ts` file in a `styles` folder at the root of your app directory.

Then, specify your font definitions as follows:

styles/fonts.ts TypeScript

```
import { Inter, Lora, Source_Sans_3 } from 'next/font/google'
import localFont from 'next/font/local'

// define your variable fonts
const inter = Inter()
const lora = Lora()
// define 2 weights of a non-variable font
```

```
const sourceCodePro400 = Source_Sans_3({ weight: '400' })
const sourceCodePro700 = Source_Sans_3({ weight: '700' })
// define a custom local font where GreatVibes-Regular.ttf is stored in the styles
const greatVibes = localFont({ src: './GreatVibes-Regular.ttf' })

export { inter, lora, sourceCodePro400, sourceCodePro700, greatVibes }
```

You can now use these definitions in your code as follows:

app/page.tsx TypeScript

```tsx
import { inter, lora, sourceCodePro700, greatVibes } from '../styles/fonts'

export default function Page() {
  return (
    <div>
      <p className={inter.className}>Hello world using Inter font</p>
      <p style={lora.style}>Hello world using Lora font</p>
      <p className={sourceCodePro700.className}>
        Hello world using Source_Sans_3 font with weight 700
      </p>
      <p className={greatVibes.className}>My title in Great Vibes font</p>
    </div>
  )
}
```

To make it easier to access the font definitions in your code, you can define a path alias in your `tsconfig.json` or `jsconfig.json` files as follows:

tsconfig.json

```json
{
  "compilerOptions": {
    "paths": {
      "@/fonts": ["./styles/fonts"]
    }
  }
}
```

You can now import any font definition as follows:

app/about/page.tsx TypeScript

```tsx
import { greatVibes, sourceCodePro400 } from '@/fonts'
```

Version Changes

Version	Changes
`v13.2.0`	`@next/font` renamed to `next/font`. Installation no longer required.
`v13.0.0`	`@next/font` was added.

Previous
Components

Next
<Form>

Was this helpful?

Subscribe to our newsletter

Stay updated on new releases and features, guides, and case studies.

you@domain.com Subscribe