A MONOSPACE SIBLING OF NAME SANS, FROM ARROWTYPE ↘

NAME mono

v0.3

A MONOSPACE SIBLING OF NAME SANS, FROM ARROWTYPE ↘

NAME MONO IS *A MONOSPACE SIBLING*
OF NAME SANS, FROM ARROWTYPE.

Name Mono started as ***Name Sans.***
It is adapted into a fixed-width
system *for code, data, captions,*
writing, art, fashion, and more.

The *Italics* are a more *free and*
expressive take on the concept,
for *better ergonomics* in code.

Font metrics *(x-height, etc)* are matched between Name Mono & Name Sans for natural pairing:

Name Sans Text Bold

Optical Size = 12

Name Mono Bold

STYLES IN NAME MONO

Black	**& *Italic***	**# v0.3**
ExtraBold	**& *Italic***	**# v0.3**
Bold	**& *Italic***	**# v0.3**
SemiBold	**& *Italic***	**# v0.3**
Medium	& *Italic*	# v0.3
Regular	& *Italic*	# v0.3
Light	*& Italic*	*# soon*
ExtraLight	*& Italic*	*# soon*
Thin	*& Italic*	*# soon*

Uprights are unique and charming, yet straightforward. Italics get *a bit weird* for *punchy emphasis*.

Upright +Italic

A Á Ă Ắ Ặ Ằ Ẳ Ẵ Ă Â Ấ Ậ Ầ Ẩ Ẫ Ä Ạ À Ả Ā Ą Å Ã
Æ B C Ć Č Ç D Đ Ď Đ E É Ĕ Ě Ê Ế Ệ Ề Ể Ễ Ë Ė Ẹ
È Ẻ Ē Ę Ẽ F G Ğ Ģ H I Í Ĭ Î Ï İ Ị Ì Ỉ Ī Į Ĩ Ʊ
Ứ J J́ K Ķ L Ĺ Ľ Ļ Ł M N Ń Ň Ņ Ñ Ŋ O Ó Ŏ Ô Ố Ộ
Ồ Ổ Ỗ Ö Ọ Ò Ỏ Ơ Ớ Ợ Ờ Ở Õ Ő Ō Ø Ǿ Õ Œ P Þ Q R
Ŕ Ř Ŗ S Ś Š Ş Ș ß T Ŧ Ť Ţ Ț U Ú Ŭ Ǔ Û Ü Ụ Ù Ủ
Ư Ứ Ự Ừ Ử Ữ Ű Ū Ų Ů Ũ V W X Y Ý Ÿ Ỵ Ỳ Ỷ Ỹ Z Ź
Ž Ż a á ă ắ ặ ằ ẳ ẵ ă â ấ ậ ầ ẩ ẫ ä ạ à ả ā ą
å ã æ b c ć č ç d ð ď đ e é ĕ ě ê ế ệ ề ể ễ ë
ė ẹ è ẻ ē ę ẽ f g ğ ģ h i ı í ĭ î ï ị ì ỉ ī į
ĩ ij ǐj ǰ ȷ ȷ́ k ķ l Ĺ ľ ļ ł m n ń ň ņ ñ ŋ o ó ŏ
ô ố ộ ồ ổ ỗ ö ọ ò ỏ ơ ớ ợ ờ ở õ ő ō ø ǿ õ œ p
þ q r ŕ ř ŗ s ś š ş ș ß t ŧ ť ţ ț u ú ŭ ǔ û ü
ụ ù ủ ư ứ ự ừ ử ữ ű ū ų ů ũ v w x y ý ÿ ỿ ỳ ỷ
ỹ z ź ž ż 0 1 2 3 4 5 6 7 8 9 ⁰ ¹ ² ³ ⁴ ⁵ ⁶ ⁷
⁸ ⁹ ⁄ ₀ ₁ ₂ ₃ ₄ ₅ ₆ ₇ ₈ ₉ · , : ; … ! ¡ ? ¿ ·
▪ * # / \ - – — _ ( ) { } [ ] ⟨ ⟩ , „ “ ” ‘ ’
« » ‹ › “ ‘ ƒ @ & ¶ § ® ° ′ ″ | ¦ † ‡ € ¢ $ ₹
£ ₩ ¥ + − × ÷ = ≠ > < ≥ ≤ ± ≈ ¬ ~ ^ ∫ Δ ∏ ∑ √
∂ µ % ↑ ↗ → ↘ ↓ ↙ ← ↖ ↔ ↕ ◊ *v0.3 characters*

Name Mono has *numerous glyph alts*
for added legibility & flexibility.

a r g f i l 0

ss01 ss03 ss05 ss07 ss09 ss11 zero

a r g f i l 0

a r g f i l 0

ss02 ss04 ss06 ss08 ss10 ss12 zero

a r g f i l 0

v0.3 adds alts: a *classic uppercase, brutalist punctuation, oldstyle numerals,* and *high-legibility 6 & 9*

ABCDE ABCDE
FGHIJ FGHIJ
KLMNO KLMNO
PQRST PQRST
UVWXY UVWXY
Z”Ø96 Z"Ø96

Franklin St – Prospect Av

Saratoga Av – Delancey St

Bay 50th St – Nostrand Av

Atlantic Av – Winthrop St

Sterling St – Rockaway Av

Montrose Av – Chambers St

Chauncey St – New Lots Av

Kingston Av – Bleecker St

Crescent St – Parkside Av

```tsx
export default async function Article(props: ArticleProps) {
  const data = await getData(props);
  const article = data.viewer.slug?.article;
  if (!article) notFound();

  return (
    <div className={styles.article}>
      <div className={styles.article__header}>
        {article.images?.length ? (
          <div className={styles.customer__article__hero}>
            {article.images.map((image, i) =>
              image.tags?.includes("custom-font-title") ? (
                <InlineSVG href={image.url ? image.url : ""} key={i} />
              ) : null,
            )}
          </div>
        ) : null}

        <div className={styles.article__header__info}>
          <h1
            className={article.tags?.includes("custom") ? "custom-title"
            >
            {article.title}
          </h1>
        </div>
      </div>

      <div className={styles.article__body + " markdown"}>
        <FontdueHTML html={article.body} />
      </div>
    </div>
  );
```

```scss
@import "../../styles/settings/breakpoints";
@import "../../styles/tools/breakpoints";

.about-page-main {
  max-width: 100%;

  @include minWidth(desktop) {
    display: grid;
    grid-template-columns: 100%;
    grid-template-columns: minmax(25%, 1fr) minmax(0, var(--maxwidth-art
        25%,
        1fr
      );
    grid-gap: var(--outer-gutter);
  }

  @include minWidth(hires) {
    grid-gap: var(--gutter);
  }

  h1 {
    @include minWidth(desktop) {
      grid-column: 1 / -1;
    }

    @include minWidth(xl) {
      grid-column: 2 / -1;
    }
  }

  [class="about-text"] {
    @include minWidth(desktop) {
```

One good way to avoid remembering this or worrying about syntax is by aut

Automating it with a function

The function below will make a clean zip if you call if with a path, like

```sh
function zipit {
  currentDir=$(pwd)                          # get current dir so you
  cd $(dirname $1)                           # change to target's dir
  target=$(basename $1)                      # get target's name
  zip -r $target.zip $target -x '*/.DS_Store'  # make a zip of the targe
  echo "zip made of " $1                     # announce completion
  cd $currentDir                             # return to where you wer
}
```

It may seem a little bit complicated, but due to the way `zip` works, it

Of course, it is worth making sure the terminal has this function in memo

The Result

When you exclude the `.DS_Store` files, it avoids both problems, and the

An easier solution

I mentioned this article in a [Twitter thread](https://twitter.com/Arrow

```c
/* SPDX-License-Identifier: GPL-2.0-or-later */

#pragma once

/* Comment or uncomment the next line to change the orientation of the pr
// #define PRO_MICRO_FLIPPED

/* USB Device descriptor parameter */

/* key matrix size */
#define MATRIX_ROWS 4
#define MATRIX_COLS 10

#define MATRIX_ROW_PINS { GP14, GP13, GP0, GP12 }
#define MATRIX_COL_PINS { GP1, GP2, GP3, GP4, GP5, GP6, GP8, GP9, GP10, G

/* Mechanical locking support. Use KC_LCAP, KC_LNUM or KC_LSCR instead in
#define LOCKING_SUPPORT_ENABLE
/* Locking resynchronize hack */
#define LOCKING_RESYNC_ENABLE

#define RP2040_BOOTLOADER_DOUBLE_TAP_RESET
#define RP2040_BOOTLOADER_DOUBLE_TAP_RESET_TIMEOUT 500U

```

```
    17: 'styleName'      # Style Name
}

# GET / SET NAME HELPER FUNCTIONS

def getFontNameID(font, ID, platformID=3, platEncID=1):
    name = str(font['name'].getName(ID, platformID, platEncID))
    return name

def setFontNameID(font, ID, newName):
    print(f"\n\t▪ name {ID}:")
    macIDs = {"platformID": 3, "platEncID": 1, "langID": 0x409}
    winIDs = {"platformID": 1, "platEncID": 0, "langID": 0x0}

    oldMacName = font['name'].getName(ID, *macIDs.values())
    oldWinName = font['name'].getName(ID, *winIDs.values())

    if oldMacName != newName:
        print(f"\n\t\t Mac name was '{oldMacName}'")
        font['name'].setName(newName, ID, *macIDs.values())
        print(f"\t\t Mac name now '{newName}'")

    if oldWinName != newName:
        print(f"\n\t\t Win name was '{oldWinName}'")
        font['name'].setName(newName, ID, *winIDs.values())
        print(f"\t\t Win name now '{newName}'")


def cleanName(name, particleToRemove):
    """
        Because these are being left in right now, and appearing in font
    """
```

arrow type