brilliant-cv

Documentation

Authors: mintyfrankie

Build Date: 2025-02-17

Version: 2.0.4

Contents

1. Introduction

Brilliant CV is a Typst template for making Résume, CV or Cover Letter inspired by the famous LaTeX CV template Awesome-CV.

2. Setup

Step 1: Install Fonts

In order to make Typst render correctly, you will have to install the required fonts Roboto and Source Sans Pro (or Source Sans 3) in your local system.

Step 2: Check Documentation

You are reading this documentation now, woah!

Step 3: Bootstrap Template

In your local system, just working like `git clone`, boostrap the template using this command:

```
typst init @preview/brilliant-cv:<version>
```

Replace the `<version>` with the latest or any releases (after 2.0.0).

Step 4: Compile Files

Adapt the `metadata.toml` to suit your needs, then `typst c cv.typ` to get your first CV!

Step 5: Go beyond

It is recommended to:

1. Use `git` to manage your project, as it helps trace your changes and version control your CV.
2. Use `typstyle` and `pre-commit` to help you format your CV.
3. Use `typos` to check typos in your CV if your main locale is English.
4. (Advanced) Use `LTex` in your favorite code editor to check grammars and get language suggestions.

3. Migration from `v1` to `v2`

With an existing CV project using the `v1` version of the template, a migration is needed, including replacing some files / some content in certain files.

1. Delete `brilliant-CV` folder, `.gitmodules`. (Future package management will directly be managed by Typst)
2. Migrate all the config on `metadata.typ` by creating a new `metadata.toml`. Follow the example toml file in the repo,

it is rather straightforward to migrate.

3. For `cv.typ` and `letter.typ`, copy the new files from the repo, and adapt the modules you have in your project.
4. For the module files in `/modules_*` folders:

   a. Delete the old import `#import "../brilliant-CV/template.typ": *`, and replace it by the import statements in the new template files.

   b. Due to the Typst path handling mecanism, one cannot directly pass the path string to some functions anymore. This concerns, for example, the logo argument in cvEntry, but also on `cvPublication` as well. Some parameter names were changed, but most importantly, you should pass a function instead of a string (i.e. `image("logo.png")` instead of `"logo.png"`). Refer to new template files for reference.

   c. You might need to install `Roboto` and `Source Sans Pro` on your local system now, as new Typst package discourages including these large files.

   d. Run `typst c cv.typ` without passing the `font-path` flag. All should be good now, congrats!

Feel free to raise an issue for more assistance should you encounter a problem that you cannot solve on your own :)

4. Confuguration via `metadata.toml`

The `metadata.toml` file is the main configuration file for your CV. By changing the key-value pairs in the config file, you can setup the names, contact information, and other details that will be displayed in your CV.

Here is an example of a `metadata.toml` file:

```toml
 # INFO: value must matches folder suffix; i.e "zh" -> "./modules_zh"
language = "en"

[layout]
# Optional values: skyblue, red, nephritis, concrete, darknight
awesome_color = "skyblue"

# Skips are for controlling the spacing between sections and entries
before_section_skip = "1pt"
before_entry_skip = "1pt"
before_entry_description_skip = "1pt"

[layout.header]
# Optional values: left, center, right
header_align = "left"

 # Decide if you want to display profile photo or not
display_profile_photo = true
profile_photo_path = "template/src/avatar.png"

[layout.entry]
# Decide if you want to put your company in bold or your position in bold
display_entry_society_first = true

# Decide if you want to display organisation logo or not
display_logo = true

[inject]
# Decide if you want to inject AI prompt or not
inject_ai_prompt = false

# Decide if you want to inject keywords or not
inject_keywords = true
injected_keywords_list = ["Data Analyst", "GCP", "Python", "SQL", "Tableau"]

[personal]
first_name = "John"
last_name = "Doe"

# The order of this section will affect how the entries are displayed
# The custom value is for any additional information you want to add
[personal.info]
github = "mintyfrankie"
phone = "+33 6 12 34 56 78"
email = "john.doe@me.org"
linkedin = "johndoe"
# gitlab = "mintyfrankie"
# homepage: "jd.me.org"
# orcid = "0000-0000-0000-0000"
```

```
# researchgate = "John-Doe"
# extraInfo = "I am a cool kid"
# custom-1 = (icon: "", text: "example", link: "https://example.com")

# add a new section if you want to include the language of your choice
# i.e. [[lang.ru]]
# each section must contains the following fields
[lang.en]
header_quote = "Experienced Data Analyst looking for a full time job starting from
now"
cv_footer = "Curriculum vitae"
letter_footer = "Cover letter"

[lang.fr]
header_quote = "Analyste de données expérimenté à la recherche d'un emploi à temps
plein disponible dès maintenant"
cv_footer = "Résumé"
letter_footer = "Lettre de motivation"

[lang.zh]
header_quote = "□□□□□□□□□□□□□□□□□□□□□"
cv_footer = "□□"
letter_footer = "□□□"

 # For languages that are not written in Latin script
 # Currently supported non-latin language codes: ("zh", "ja", "ko", "ru")
[lang.non_latin]
name = "□□□"
font = "Heiti SC"
```

5. Functions

cvEntry

Add an entry to the CV.

Parameters

```
cvEntry(
  title: str,
  society: str,
  date: str,
  location: str,
  description: array,
  logo: image,
  tags: array,
  metadata: array,
  awesomeColors: array
) -> content
```

title `str`

The title of the entry.

Default: `"Title"`

society `str`

The society of the entr (company, university, etc.).

Default: `"Society"`

date `str`

The date of the entry.

Default: `"Date"`

location `str`

The location of the entry.

Default: `"Location"`

description `array`

The description of the entry. It can be a string or an array of strings.

Default: `"Description"`

logo `image`

The logo of the society. If empty, no logo will be displayed.

Default: `""`

tags `array`

The tags of the entry.

Default: `()`

metadata `array`

(optional) the metadata read from the TOML file.

Default: `metadata`

awesomeColors `array`

(optional) the awesome colors of the CV.

Default: `awesomeColors`

cvHonor

Add a Honor to the CV.

Parameters

```
cvHonor(
  date: str,
  title: str,
  issuer: str,
  url: str,
  location: str,
  awesomeColors: array,
  metadata: array
) -> content
```

date `str`

The date of the honor.

Default: `"1990"`

title `str`

The title of the honor.

Default: `"Title"`

issuer `str`

The issuer of the honor.

Default: `""`

url `str`

The URL of the honor.

Default: `""`

location `str`

The location of the honor.

Default: `""`

awesomeColors `array`

(optional) The awesome colors of the CV.

Default: `awesomeColors`

metadata `array`

(optional) The metadata read from the TOML file.

Default: `metadata`

cvPublication
Add the publications to the CV by reading a bib file.

Parameters

```
cvPublication(
  bib: bibliography,
  keyList: list,
  refStyle: str,
  refFull: bool
) -> content
```

bib `bibliography`

The `bibliography` object with the path to the bib file.

Default: `""`

keyList `list`

The list of keys to include in the publication list.

Default: `list()`

refStyle `str`

The reference style of the publication list.

Default: `"apa"`

refFull `bool`

Whether to show the full reference or not.

Default: `true`

cvSection

Add the title of a section.

NOTE: If the language is non-Latin, the title highlight will not be sliced.

Parameters

```
cvSection(
  title: str,
  highlighted: bool,
  letters: int,
  metadata: array,
  awesomeColors: array
) -> content
```

title `str`

The title of the section.

highlighted `bool`

Whether the first n letters will be highlighted in accent color.

Default: `true`

letters `int`

The number of first letters of the title to highlight.

Default: `3`

metadata `array`

(optional) the metadata read from the TOML file.

Default: `metadata`

awesomeColors `array`

(optional) the awesome colors of the CV.

Default: `awesomeColors`

cvSkill

Add a skill to the CV.

Parameters

```
cvSkill(
  type: str,
  info
) -> content
```

type `str`

The type of the skill. It is displayed on the left side.

- info (str | content): The information about the skill. It is displayed on the right side. Items can be separated by `#hbar()`.

Default: `"Type"`

cvSkillTag

Add a skill tag to the CV.

- skill (str | content): The skill to be displayed.

Parameters

```
cvSkillTag(skill) -> content
```

cvSkillWithLevel

Add a skill with a level to the CV.

Parameters

```
cvSkillWithLevel(
  type: str,
  level: int,
  info
) -> content
```

type `str`

The type of the skill. It is displayed on the left side.

Default: `"Type"`

level `int`

The level of the skill. It is displayed in as circles in the middle. The minimum level is 0 and the maximum level is 5.

- info (str | content): The information about the skill. It is displayed on the right side.

Default: `3`