CJK Num Format

Format dates and numbers.

0.1.0-4-g5304eb9 2025-06-18

https://github.com/wensimehrp/cjk-num-format

Created by Jeremy Gao

Contents

Goals

This library provides basic CJK (Chinese, Japanese, and Korean) number and date formatting utilities. It is designed with simplicity in mind, and with maximum compatibility with different CJK languages and regions. Features include:

- Formatting numbers with thousands separators.
  - including *Daxie* used by Chinese.
- Formatting dates in various CJK date formats.
- Automatic handling of locals, based on the current text language and region (i.e., the `lang` and `region` fields of `text`)

I do not aim to cover all possible utilities related to CJK number formatting, but rather to provide a solid foundation for common use cases. If you have specific needs or suggestions, feel free to open an issue on the GitHub repository.

Localization

DISCLAIMER: I am not a Japanese nor a Korean expert, so I have relied on existing resources (i.e. Wikipedia, LLM) to implement features for these languages. If you find any mistakes or have suggestions for improvements, please let me know.

Functions provided by this library can adapt to the current text language and region (hence their outputs are `content` but not `string` or `int`). See each function's documentation for details.

Library Functions

cjk-content

Generate a smart CJK content based on the language and region.

Parameters

```
cjk-content(
  zh: dict str,
  ja: str,
  ko: str
) -> content
```

zh `dict` or `str`

String(s) for Chinese. You can optionally provide a dictionary with keys `cn`, `tw`, `hk`, etc. to specify different strings for different regions. `HK`, `MO`, and `TW` will default to the `tw` string, while other regions (e.g., Singapore) will default to the `cn` string.

Default: `none`

ja `str`

String for Japanese.

Default: `none`

ko `str`

String for Korean.

Default: `none`

cjk-date-format

Format a date in CJK style, including the year, month, and day.

typ
```
#cjk-num-format.cjk-date-format(
  datetime(year: 2023, month: 10, day: 1),
)
```

二〇二三年十月一日

Parameters

```
cjk-date-format(
  date: datetime ,
  pfx: str content ,
  negative-pfx: str content ,
  established: int ,
  arabic: auto bool
) -> content
```

date datetime

The date to format.

pfx str or content

Prefix for the date string.

typ
```
#cjk-num-format.cjk-date-format(
  pfx: "西元",
  datetime(year: 2023, month: 10, day: 1),
)
```

西元二〇二三年十月一日

Default: none

negative-pfx str or content

Prefix for negative years.

typ
```
#cjk-num-format.cjk-date-format(
  negative-pfx: "西元前",
  datetime(year: -2023, month: 10, day: 1),
)
```

西元前二千零二十四年十月一日

Default: none

established int

The year when the era was established.

Default: 1

arabic `auto` or `bool`

Whether to use Arabic numerals for the year.

typ
```
#cjk-num-format.cjk-date-format(
  arabic: true,
  datetime(year: 2023, month: 10, day: 1),
)
```

```
2023年10月1日
```

Default: `auto`

daxie

Format a number in Chinese currency style. This function is based on `numbering("壹", v)` but adds units for the whole number and the first two decimal places.

```typ
#cjk-num-format.daxie(123456)\
#cjk-num-format.daxie(8642.99)\
#cjk-num-format.daxie(2356, u1: "圆", whole:
"正")
```

拾贰万叁仟肆佰伍拾陆元整
捌仟陆佰肆拾贰元玖角玖分
贰仟叁佰伍拾陆圆正

Parameters

```
daxie(
  v: int float,
  u1: str,
  u2: str,
  u3: str,
  whole: str
) -> str
```

v `int` or `float`

The value to format.

u1 `str`

The unit for the whole number (e.g., “元”).

Default: `"元"`

u2 `str`

The unit for the first decimal place (e.g., “角”).

Default: `"角"`

u3 `str`

The unit for the second decimal place (e.g., “分”).

Default: `"分"`

whole `str`

The string to append if the value is a whole number (e.g., “整”).

Default: `"整"`

japan-date-format

Format a date in the Japanese era style (e.g., Meiji, Taisho, Showa, Heisei, Reiwa).

typ
```
#cjk-num-format.japan-date-format(
  datetime(year: 2023, month: 10, day: 1),
)\
#cjk-num-format.japan-date-format(
  datetime(year: 1989, month: 1, day: 6),
)\
#cjk-num-format.japan-date-format(
  datetime(year: 1989, month: 1, day: 7),
)
```

令和五年十月一日
昭和六十四年一月六日
平成元年一月七日

Parameters

```
japan-date-format(
  date: datetime,
  arabic: bool auto
) -> content
```

date `datetime`

The date to format.

arabic `bool` or `auto`

Whether to use Arabic numerals for the year.

Default: `auto`

juche-date-format

Format a date in the Juche calendar style (North Korea's calendar).

Parameters

```
juche-date-format(
  date: datetime ,
  pfx: str  content ,
  negative-pfx: str  content ,
  arabic: auto  bool
) -> content
```

date `datetime`

The date to format.

pfx `str` or `content`

Prefix for the date string.

Default: `auto`

negative-pfx `str` or `content`

Prefix for negative years.

Default: `auto`

arabic `auto` or `bool`

Whether to use Arabic numerals for the year.

Default: `auto`

roc-date-format

Format a date in the Republic of China (ROC) calendar style.

typ
```
#cjk-num-format.roc-date-format(
  datetime(year: 1949, month: 9, day: 30),
)\
#cjk-num-format.roc-date-format(
  datetime(year: 1912, month: 10, day: 1),
)\
#cjk-num-format.roc-date-format(
  datetime(year: 1910, month: 10, day: 1),
)\
#cjk-num-format.roc-date-format(
  datetime(year: 2025, month: 10, day: 1),
)\
#cjk-num-format.roc-date-format(
  datetime(year: 2025, month: 10, day: 1),
  arabic: true,
)
```

```
民国三十八年九月三十日
民国元年十月一日
民前二年十月一日
民国一一四年十月一日
民国114年10月1日
```

Parameters

```
roc-date-format(
  date: datetime,
  pfx: str content,
  negative-pfx: str content,
  arabic: auto bool
) -> content
```

date `datetime`

The date to format.

pfx `str` or `content`

Prefix for the date string.

Default: `auto`

negative-pfx `str` or `content`

Prefix for negative years.

Default: `auto`

arabic `auto` or `bool`

Whether to use Arabic numerals for the year.

Default: `auto`

sep-by-ten-thousands

Generate a string with the given value formatted with thousands separators.

typ

```
#set text(lang: "ja", region: "jp")
#let c = cjk-num-format
#c.sep-by-ten-thousands(12345670000000)\
#c.sep-by-ten-thousands(1145141919810)\
#c.sep-by-ten-thousands(1145141919810000)\
#c.sep-by-ten-thousands(135700012255)\
```

```
12兆3456億7000万
1兆1451億4191万9810
1145兆1419億1981万
1357億1万2255
```

Parameters

```
sep-by-ten-thousands(
  value: int,
  separators: array,
  lang: str,
  region: str
) -> str
```

value `int`

The number to format.

separators `array`

An array of strings to use as thousands separators.

Default: `none`

lang `str`

The language code to use for formatting (e.g., “en”, “zh”).

Default: `none`

region `str`

The region code to use for formatting (e.g., “US”, “CN”).

Default: `none`