wcwidth

Indices and tables

v: stable

downloads 66M/month codecov 100% license MIT

Introduction

This library is mainly for CLI programs that carefully produce output for Terminals, or make pretend to be an emulator.

Problem Statement: The printable length of *most* strings are equal to the number of cells they occupy on the screen `1 character : 1 cell`. However, there are categories of characters that *occupy 2 cells* (full-wide), and others that *occupy 0* cells (zero-width).

Solution: POSIX.1-2001 and POSIX.1-2008 conforming systems provide wcwidth(3) and wcswidth(3) C functions of which this python module's functions precisely copy. *These functions return the number of cells a unicode string is expected to occupy.*

Installation

The stable version of this package is maintained on pypi, install using pip:

```
pip install wcwidth
```

Example

Problem: given the following phrase (Japanese),

```
>>>  text = u'コンニチハ'
```

Python **incorrectly** uses the *string length* of 5 codepoints rather than the *printable length* of 10 cells, so that when using the *rjust* function, the output length is wrong:

```
>>> print(len('コンニチハ'))
5

>>> print('コンニチハ'.rjust(20, '_'))
_______________コンニチハ
```

By defining our own "rjust" function that uses wcwidth, we can correct this:

```
>>> def wc_rjust(text, length, padding=' '):
...    from wcwidth import wcswidth
...    return padding * max(0, (length - wcswidth(text))) + text
...
```

Our **Solution** uses wcswidth to determine the string length correctly:

```
>>> from wcwidth import wcswidth
>>> print(wcswidth('コンニチハ'))
10

>>> print(wc_rjust('コンニチハ', 20, '_'))
__________コンニチハ
```

Choosing a Version

Export an environment variable, `UNICODE_VERSION`. This should be done by *terminal emulators* or those developers experimenting with authoring one of their own, from shell:

```
$ export UNICODE_VERSION=13.0
```

If unspecified, the latest version is used. If your Terminal Emulator does not export this variable, you can use the jquast/ucs-detect utility to automatically detect and export it to your shell.

wcwidth, wcswidth

Use function `wcwidth()` to determine the length of a *single unicode character*, and `wcswidth()` to determine the length of many, a *string of unicode characters*.

Briefly, return values of function `wcwidth()` are:

`-1`
: Indeterminate (not printable).

`0`
: Does not advance the cursor, such as NULL or Combining.

`2`
: Characters of category East Asian Wide (W) or East Asian Full-width (F) which are displayed using two terminal cells.

`1`
: All others.

Function `wcswidth()` simply returns the sum of all values for each character along a string, or `-1` when it occurs anywhere along a string.

Full API Documentation at https://wcwidth.readthedocs.org

Developing

Install wcwidth in editable mode:

```
pip install -e .
```

Execute unit tests using tox:

```
tox -e py27,py35,py36,py37,py38,py39,py310,py311,py312
```

Updating Unicode Version

Regenerate python code tables from latest Unicode Specification data files:

```
tox -e update
```

The script is located at `bin/update-tables.py`, requires Python 3.9 or later. It is recommended but not necessary to run this script with the newest Python, because the newest Python has the latest `unicodedata` for generating comments.

Building Documentation

This project is using sphinx 4.5 to build documentation:

```
tox -e sphinx
```

The output will be in `docs/_build/html/`.

Updating Requirements

This project is using pip-tools to manage requirements.

To upgrade requirements for updating unicode version, run:

```
tox -e update_requirements_update
```

To upgrade requirements for testing, run:

```
tox -e update_requirements37,update_requirements39
```

To upgrade requirements for building documentation, run:

```
tox -e update_requirements_docs
```

Utilities

Supplementary tools for browsing and testing terminals for wide unicode characters are found in the bin/ of this project's source code. Just ensure to first `pip install -r requirements-develop.txt` from this projects main folder. For example, an interactive browser for testing:

```
python ./bin/wcwidth-browser.py
```

Uses

This library is used in:

- jquast/blessed: a thin, practical wrapper around terminal capabilities in Python.
- prompt-toolkit/python-prompt-toolkit: a Library for building powerful interactive command lines in Python.
- dbcli/pgcli: Postgres CLI with autocompletion and syntax highlighting.
- thomasballinger/curtsies: a Curses-like terminal wrapper with a display based on compositing 2d arrays of text.
- selectel/pyte: Simple VTXXX-compatible linux terminal emulator.
- astanin/python-tabulate: Pretty-print tabular data in Python, a library and a command-line utility.
- rspeer/python-ftfy: Fixes mojibake and other glitches in Unicode text.
- nbedos/termtosvg: Terminal recorder that renders sessions as SVG animations.
- peterbrittain/asciimatics: Package to help people create full-screen text UIs.
- python-cmd2/cmd2: A tool for building interactive command line apps
- stratis-storage/stratis-cli: CLI for the Stratis project
- ihabunek/toot: A Mastodon CLI/TUI client
- saulpw/visidata: Terminal spreadsheet multitool for discovering and arranging data

Other Languages

- timoxley/wcwidth: JavaScript
- janlelis/unicode-display_width: Ruby
- alecrabbit/php-wcwidth: PHP
- Text::CharWidth: Perl
- bluebear94/Terminal-WCWidth: Perl 6
- mattn/go-runewidth: Go
- grepsuzette/wcwidth: Haxe

- aperezdc/lua-wcwidth: Lua
- joachimschmidt557/zig-wcwidth: Zig
- fumiyas/wcwidth-cjk: *LD_PRELOAD* override
- joshuarubin/wcwidth9: Unicode version 9 in C

History

0.2.13 *2024-01-06*

- **Bugfix** zero-width support for Hangul Jamo (Korean)

0.2.12 *2023-11-21*

- re-release to remove .pyi file misplaced in wheel files Issue #101.

0.2.11 *2023-11-20*

- Include tests files in the source distribution (PR #98, PR #100).

0.2.10 *2023-11-13*

- **Bugfix** accounting of some kinds of emoji sequences using U+FE0F Variation Selector 16 (PR #97).
- **Updated** Specification.

0.2.9 *2023-10-30*

- **Bugfix** zero-width characters used in Emoji ZWJ sequences, Balinese, Jamo, Devanagari, Tamil, Kannada and others (PR #91).
- **Updated** to include Specification of character measurements.

0.2.8 *2023-09-30*

- Include requirements files in the source distribution (PR #82).

0.2.7 *2023-09-28*

- **Updated** tables to include Unicode Specification 15.1.0.
- Include `bin`, `docs`, and `tox.ini` in the source distribution

0.2.6 *2023-01-14*

- **Updated** tables to include Unicode Specification 14.0.0 and 15.0.0.
- **Changed** developer tools to use pip-compile, and to use jinja2 templates for code generation in *bin/update-tables.py* to prepare for possible compiler optimization release.

0.2.1 .. 0.2.5 *2020-06-23*

- **Repository** changes to update tests and packaging issues, and begin tagging repository with matching release versions.

0.2.0 *2020-06-01*

- **Enhancement**: Unicode version may be selected by exporting the Environment variable `UNICODE_VERSION`, such as `13.0`, or `6.3.0`. See the jquast/ucs-detect CLI utility for automatic detection.
- **Enhancement**: API Documentation is published to readthedocs.org.
- **Updated** tables for *all* Unicode Specifications with files published in a programmatically consumable format, versions 4.1.0 through 13.0

0.1.9 *2020-03-22*

- **Performance** optimization by Avram Lubkin, PR #35.
- **Updated** tables to Unicode Specification 13.0.0.

0.1.8 *2020-01-01*

- **Updated** tables to Unicode Specification 12.0.0. (PR #30).

0.1.7 *2016-07-01*

- **Updated** tables to Unicode Specification 9.0.0. (PR #18).

0.1.6 *2016-01-08 Production/Stable*

- `LICENSE` file now included with distribution.

0.1.5 *2015-09-13 Alpha*

- **Bugfix**: Resolution of “combining character width” issue, most especially those that previously returned -1 now often (correctly) return 0. resolved by Philip Craig via PR #11.

- **Deprecated**: The module path `wcwidth.table_comb` is no longer available, it has been superseded by module path `wcwidth.table_zero`.

0.1.4 *2014-11-20 Pre-Alpha*

- **Feature**: `wcswidth()` now determines printable length for (most) combining characters. The developer's tool bin/wcwidth-browser.py is improved to display combining characters when provided the `--combining` option (Thomas Ballinger and Leta Montopoli PR #5).
- **Feature**: added static analysis (prospector) to testing framework.

0.1.3 *2014-10-29 Pre-Alpha*

- **Bugfix**: 2nd parameter of wcswidth was not honored. (Thomas Ballinger, PR #4).

0.1.2 *2014-10-28 Pre-Alpha*

- **Updated** tables to Unicode Specification 7.0.0. (Thomas Ballinger, PR #3).

0.1.1 *2014-05-14 Pre-Alpha*

- Initial release to pypi, Based on Unicode Specification 6.3.0

This code was originally derived directly from C code of the same name, whose latest version is available at https://www.cl.cam.ac.uk/~mgk25/ucs/wcwidth.c:

```
   * Markus Kuhn -- 2007-05-26 (Unicode 5.0)
   *
   * Permission to use, copy, modify, and distribute this software
   * for any purpose and without fee is hereby granted. The author
   * disclaims all warranties with regard to this software.
```

Unicode release files

This library aims to be forward-looking, portable, and most correct. The most current release of this API is based on the Unicode Standard release files:

`DerivedGeneralCategory-4.1.0.txt`
: *Date: 2005-02-26, 02:35:50 GMT [MD]*

`DerivedGeneralCategory-5.0.0.txt`
: *Date: 2006-02-27, 23:41:27 GMT [MD]*

`DerivedGeneralCategory-5.1.0.txt`
: *Date: 2008-03-20, 17:54:57 GMT [MD]*

`DerivedGeneralCategory-5.2.0.txt`
: *Date: 2009-08-22, 04:58:21 GMT [MD]*

`DerivedGeneralCategory-6.0.0.txt`
: *Date: 2010-08-19, 00:48:09 GMT [MD]*

`DerivedGeneralCategory-6.1.0.txt`
: *Date: 2011-11-27, 05:10:22 GMT [MD]*

`DerivedGeneralCategory-6.2.0.txt`
: *Date: 2012-05-20, 00:42:34 GMT [MD]*

`DerivedGeneralCategory-6.3.0.txt`
: *Date: 2013-07-05, 14:08:45 GMT [MD]*

`DerivedGeneralCategory-7.0.0.txt`
: *Date: 2014-02-07, 18:42:12 GMT [MD]*

`DerivedGeneralCategory-8.0.0.txt`
: *Date: 2015-02-13, 13:47:11 GMT [MD]*

`DerivedGeneralCategory-9.0.0.txt`
: *Date: 2016-06-01, 10:34:26 GMT*

`DerivedGeneralCategory-10.0.0.txt`
: *Date: 2017-03-08, 08:41:49 GMT*

`DerivedGeneralCategory-11.0.0.txt`
: *Date: 2018-02-21, 05:34:04 GMT*

`DerivedGeneralCategory-12.0.0.txt`
: *Date: 2019-01-22, 08:18:28 GMT*

`DerivedGeneralCategory-12.1.0.txt`
: *Date: 2019-03-10, 10:53:08 GMT*

`DerivedGeneralCategory-13.0.0.txt`
: *Date: 2019-10-21, 14:30:32 GMT*

`DerivedGeneralCategory-14.0.0.txt`
: *Date: 2021-07-10, 00:35:08 GMT*

`DerivedGeneralCategory-15.0.0.txt`
: *Date: 2022-04-26, 23:14:35 GMT*

`DerivedGeneralCategory-15.1.0.txt`
: *Date: 2023-07-28, 23:34:02 GMT*

`EastAsianWidth-4.1.0.txt`
  Date: 2005-03-17, 15:21:00 PST [KW]

`EastAsianWidth-5.0.0.txt`
  Date: 2006-02-15, 14:39:00 PST [KW]

`EastAsianWidth-5.1.0.txt`
  Date: 2008-03-20, 17:42:00 PDT [KW]

`EastAsianWidth-5.2.0.txt`
  Date: 2009-06-09, 17:47:00 PDT [KW]

`EastAsianWidth-6.0.0.txt`
  Date: 2010-08-17, 12:17:00 PDT [KW]

`EastAsianWidth-6.1.0.txt`
  Date: 2011-09-19, 18:46:00 GMT [KW]

`EastAsianWidth-6.2.0.txt`
  Date: 2012-05-15, 18:30:00 GMT [KW]

`EastAsianWidth-6.3.0.txt`
  Date: 2013-02-05, 20:09:00 GMT [KW, LI]

`EastAsianWidth-7.0.0.txt`
  Date: 2014-02-28, 23:15:00 GMT [KW, LI]

`EastAsianWidth-8.0.0.txt`
  Date: 2015-02-10, 21:00:00 GMT [KW, LI]

`EastAsianWidth-9.0.0.txt`
  Date: 2016-05-27, 17:00:00 GMT [KW, LI]

`EastAsianWidth-10.0.0.txt`
  Date: 2017-03-08, 02:00:00 GMT [KW, LI]

`EastAsianWidth-11.0.0.txt`
  Date: 2018-05-14, 09:41:59 GMT [KW, LI]

`EastAsianWidth-12.0.0.txt`
  Date: 2019-01-21, 14:12:58 GMT [KW, LI]

`EastAsianWidth-12.1.0.txt`
  Date: 2019-03-31, 22:01:58 GMT [KW, LI]

`EastAsianWidth-13.0.0.txt`
  Date: 2029-01-21, 18:14:00 GMT [KW, LI]

`EastAsianWidth-14.0.0.txt`
  Date: 2021-07-06, 09:58:53 GMT [KW, LI]

`EastAsianWidth-15.0.0.txt`
  Date: 2022-05-24, 17:40:20 GMT [KW, LI]

`EastAsianWidth-15.1.0.txt`
  Date: 2023-07-28, 23:34:08 GMT

`emoji-variation-sequences-12.0.0.txt`
  Date: 2019-01-15, 12:10:05 GMT

`emoji-variation-sequences-15.1.0.txt`
  Date: 2023-02-01, 02:22:54 GMT

v: stable

Specification

This document defines how the wcwidth library measures the printable width of characters of a string.

Width of -1

The following have a column width of -1 for function **`wcwidth.wcwidth()`**

- `C0` control characters (U+0001 through U+001F).
- `C1` control characters and `DEL` (U+007F through U+00A0).

If any character in sequence contains `C0` or `C1` control characters, the final return value of of **`wcwidth.wcswidth()`** is -1.

Width of 0

Any characters defined by category codes in DerivedGeneralCategory.txt files:

- 'Me': Enclosing Combining Mark, aprox. 13 characters.
- 'Mn': Nonspacing Combining Mark, aprox. 1,839 characters.
- 'Mc': Spacing Mark, aprox. 443 characters.
- 'Cf': Format control character, aprox. 161 characters.
- 'Zl': U+2028 LINE SEPARATOR only
- 'Zp': U+2029 PARAGRAPH SEPARATOR only
- 'Sk': Modifier Symbol, aprox. 4 characters of only those where phrase `'EMOJI MODIFIER'` is present in comment of unicode data file.

The NULL character (U+0000).

Any character following ZWJ (U+200D) when in sequence by function **`wcwidth.wcswidth()`**.

Hangul Jamo Jungseong and "Extended-B" code blocks, U+1160 through U+11FF and U+D7B0 through U+D7FF.

Width of 1

String characters are measured width of 1 when they are not measured as Width of 0 or Width of 2.

Width of 2

Any character defined by East Asian Fullwidth (`F`) or Wide (`W`) properties in EastAsianWidth.txt files, except those that are defined by the Category codes of Nonspacing Mark (`Mn`) and Spacing Mark (`Mc`).

Any characters of Modifier Symbol category, `'Sk'` where `'FULLWIDTH'` is present in comment of unicode data file, aprox. 3 characters.

Any character in sequence with U+FE0F (Variation Selector 16) defined by emoji-variation-sequences.txt as `emoji style`.

Public API

This package follows SEMVER rules. Therefore, for the functions of the below list, you may safely use version dependency definition `wcwidth<2` in your requirements.txt or equivalent. Their signatures will never change.

`wcwidth.`**`wcwidth`**`(`*`wc, unicode_version='auto'`*`)` [source]

Given one Unicode character, return its printable length on a terminal.

Parameters:
- **wc** (*str*) – A single Unicode character.
- **unicode_version** (*str*) –

  A Unicode version number, such as `'6.0.0'`. A list of version levels suported by wcwidth is returned by **`list_versions()`**.

  Any version string may be specified without error – the nearest matching version is selected. When `latest` (default), the highest Unicode version level is used.

Returns: The width, in cells, necessary to display the character of Unicode string character, `wc`. Returns 0 if the `wc` argument has no printable effect on a terminal (such as NUL '0'), -1 if `wc` is not printable, or has an indeterminate effect on the terminal, such as a control character. Otherwise, the number of column positions the character occupies on a graphic terminal (1 or 2) is returned.

Return type: int

See Specification for details of cell measurement.

`wcwidth.`**`wcswidth`**`(`*`pwcs, n=None, unicode_version='auto'`*`)` [source]

Given a unicode string, return its printable length on a terminal.

Parameters:
- **pwcs** (*str*) – Measure width of given unicode string.
- **n** (*int*) – When `n` is None (default), return the length of the entire string, otherwise only the first `n` characters are measured. This argument exists only for compatibility with the C POSIX function signature. It is suggested instead to use python's string slicing capability, `wcswidth(pwcs[:n])`
- **unicode_version** (*str*) – An explicit definition of the unicode version level to use for determination, may be `auto` (default), which uses the Environment Variable, `UNICODE_VERSION` if defined, or the latest available unicode version, otherwise.

Return type: int

Returns: The width, in cells, needed to display the first `n` characters of the unicode string `pwcs`. Returns `-1` for C0 and C1 control characters!

See Specification for details of cell measurement.

`wcwidth.`**`list_versions`**`()` [source]

Return Unicode version levels supported by this module release.

Any of the version strings returned may be used as keyword argument `unicode_version` to the `wcwidth()` family of functions.

Returns: Supported Unicode version numbers in ascending sorted order.

Return type: list[str]

Private API

These functions should only be used for wcwidth development, and not used by dependent packages except with care and by use of frozen version dependency, as these functions may change names, signatures, or disappear entirely at any time in the future, and not reflected by SEMVER rules!

If stable public API for any of the given functions is needed, please suggest a Pull Request!

`wcwidth._bisearch(ucs, table)` [source]

Auxiliary function for binary search in interval table.

Parameters:
- **ucs** (*int*) – Ordinal value of unicode character.
- **table** (*list*) – List of starting and ending ranges of ordinal values, in form of `[(start, end), ...]`.

Return type: int

Returns: 1 if ordinal value ucs is found within lookup table, else 0.

`wcwidth._wcversion_value(ver_string)` [source]

Integer-mapped value of given dotted version string.

Parameters: **ver_string** (*str*) – Unicode version string, of form `n.n.n`.

Return type: tuple(int)

Returns: tuple of digit tuples, `tuple(int, [...])`.

`wcwidth._wcmatch_version(given_version)` [source]

Return nearest matching supported Unicode version level.

If an exact match is not determined, the nearest lowest version level is returned after a warning is emitted. For example, given supported levels `4.1.0` and `5.0.0`, and a version string of `4.9.9`, then `4.1.0` is selected and returned:

```
>>> _wcmatch_version('4.9.9')
'4.1.0'
>>> _wcmatch_version('8.0')
'8.0.0'
>>> _wcmatch_version('1')
'4.1.0'
```

Parameters: **given_version** (*str*) – given version for compare, may be `auto` (default), to select Unicode Version from Environment Variable, `UNICODE_VERSION`. If the environment variable is not set, then the latest is used.

Return type: str

Returns: unicode string, or non-unicode `str` type for python 2 when given `version` is also type `str`.

Index

_ | L | W

v: stable

Search

Searching for multiple words only shows matches that contain all words.

search

```
wcwidth
=======

.. toctree::

   intro
   unicode_version
   specs
   api

Indices and tables
------------------

* :ref:`genindex`
* :ref:`modindex`
* :ref:`search`
```

wcwidth

Indices and tables

```
|pypi_downloads| |codecov| |license|

============
Introduction
============

This library is mainly for CLI programs that carefully produce output for
Terminals, or make pretend to be an emulator.

**Problem Statement**: The printable length of *most* strings are equal to the
number of cells they occupy on the screen ``1 character : 1 cell``.  However,
there are categories of characters that *occupy 2 cells* (full-wide), and
others that *occupy 0* cells (zero-width).

**Solution**: POSIX.1-2001 and POSIX.1-2008 conforming systems provide
`wcwidth(3)`_ and `wcswidth(3)`_ C functions of which this python module's
functions precisely copy.  *These functions return the number of cells a
unicode string is expected to occupy.*

Installation
------------

The stable version of this package is maintained on pypi, install using pip::

    pip install wcwidth

Example
-------

**Problem**: given the following phrase (Japanese),

   >>>  text = u'コンニチハ'

Python **incorrectly** uses the *string length* of 5 codepoints rather than the
*printable length* of 10 cells, so that when using the `rjust` function, the
output length is wrong::

    >>> print(len('コンニチハ'))
    5

    >>> print('コンニチハ'.rjust(20, '_'))
    _______________コンニチハ

By defining our own "rjust" function that uses wcwidth, we can correct this::

   >>> def wc_rjust(text, length, padding=' '):
   ...    from wcwidth import wcswidth
   ...    return padding * max(0, (length - wcswidth(text))) + text
   ...

Our **Solution** uses wcswidth to determine the string length correctly::

   >>> from wcwidth import wcswidth
   >>> print(wcswidth('コンニチハ'))
   10

   >>> print(wc_rjust('コンニチハ', 20, '_'))
   __________コンニチハ


Choosing a Version
------------------

Export an environment variable, ``UNICODE_VERSION``. This should be done by
*terminal emulators* or those developers experimenting with authoring one of
their own, from shell::

   $ export UNICODE_VERSION=13.0

If unspecified, the latest version is used. If your Terminal Emulator does not
export this variable, you can use the `jquast/ucs-detect`_ utility to
automatically detect and export it to your shell.
```

```
wcwidth, wcswidth
-----------------
Use function ``wcwidth()`` to determine the length of a *single unicode
character*, and ``wcswidth()`` to determine the length of many, a *string
of unicode characters*.

Briefly, return values of function ``wcwidth()`` are:

``-1``
  Indeterminate (not printable).

``0``
  Does not advance the cursor, such as NULL or Combining.

``2``
  Characters of category East Asian Wide (W) or East Asian
  Full-width (F) which are displayed using two terminal cells.

``1``
  All others.

Function ``wcswidth()`` simply returns the sum of all values for each character
along a string, or ``-1`` when it occurs anywhere along a string.

Full API Documentation at https://wcwidth.readthedocs.org

==========
Developing
==========

Install wcwidth in editable mode::

   pip install -e .

Execute unit tests using tox_::

   tox -e py27,py35,py36,py37,py38,py39,py310,py311,py312

Updating Unicode Version
------------------------

Regenerate python code tables from latest Unicode Specification data files::

   tox -e update

The script is located at ``bin/update-tables.py``, requires Python 3.9 or
later. It is recommended but not necessary to run this script with the newest
Python, because the newest Python has the latest ``unicodedata`` for generating
comments.

Building Documentation
----------------------

This project is using `sphinx`_ 4.5 to build documentation::

   tox -e sphinx

The output will be in ``docs/_build/html/``.

Updating Requirements
---------------------

This project is using `pip-tools`_ to manage requirements.

To upgrade requirements for updating unicode version, run::

   tox -e update_requirements_update

To upgrade requirements for testing, run::

   tox -e update_requirements37,update_requirements39

To upgrade requirements for building documentation, run::
   tox -e update_requirements_docs

Utilities
---------

Supplementary tools for browsing and testing terminals for wide unicode
characters are found in the `bin/`_ of this project's source code.  Just ensure
to first ``pip install -r requirements-develop.txt`` from this projects main
folder. For example, an interactive browser for testing::

  python ./bin/wcwidth-browser.py

====
Uses
====

This library is used in:

- `jquast/blessed`_: a thin, practical wrapper around terminal capabilities in
  Python.

- `prompt-toolkit/python-prompt-toolkit`_: a Library for building powerful
  interactive command lines in Python.

- `dbcli/pgcli`_: Postgres CLI with autocompletion and syntax highlighting.

- `thomasballinger/curtsies`_: a Curses-like terminal wrapper with a display
  based on compositing 2d arrays of text.

- `selectel/pyte`_: Simple VTXXX-compatible linux terminal emulator.

- `astanin/python-tabulate`_: Pretty-print tabular data in Python, a library
  and a command-line utility.

- `rspeer/python-ftfy`_: Fixes mojibake and other glitches in Unicode
  text.

- `nbedos/termtosvg`_: Terminal recorder that renders sessions as SVG
  animations.

- `peterbrittain/asciimatics`_: Package to help people create full-screen text
  UIs.

- `python-cmd2/cmd2`_: A tool for building interactive command line apps

- `stratis-storage/stratis-cli`_: CLI for the Stratis project

- `ihabunek/toot`_: A Mastodon CLI/TUI client

- `saulpw/visidata`_: Terminal spreadsheet multitool for discovering and
  arranging data

===============
Other Languages
===============

- `timoxley/wcwidth`_: JavaScript
- `janlelis/unicode-display_width`_: Ruby
- `alecrabbit/php-wcwidth`_: PHP
- `Text::CharWidth`_: Perl
- `bluebear94/Terminal-WCWidth`_: Perl 6
- `mattn/go-runewidth`_: Go
- `grepsuzette/wcwidth`_: Haxe
- `aperezdc/lua-wcwidth`_: Lua
- `joachimschmidt557/zig-wcwidth`_: Zig
- `fumiyas/wcwidth-cjk`_: `LD_PRELOAD` override
- `joshuarubin/wcwidth9`_: Unicode version 9 in C

=======
History
=======

0.2.13 *2024-01-06*
```

```
  * **Bugfix** zero-width support for Hangul Jamo (Korean)

0.2.12 *2023-11-21*
  * re-release to remove .pyi file misplaced in wheel files `Issue #101`_.

0.2.11 *2023-11-20*
  * Include tests files in the source distribution (`PR #98`_, `PR #100`_).

0.2.10 *2023-11-13*
  * **Bugfix** accounting of some kinds of emoji sequences using U+FE0F
    Variation Selector 16 (`PR #97`_).
  * **Updated** `Specification <Specification_from_pypi_>`_.

0.2.9 *2023-10-30*
  * **Bugfix** zero-width characters used in Emoji ZWJ sequences, Balinese,
    Jamo, Devanagari, Tamil, Kannada and others (`PR #91`_).
  * **Updated** to include `Specification <Specification_from_pypi_>`_ of
    character measurements.

0.2.8 *2023-09-30*
  * Include requirements files in the source distribution (`PR #82`_).

0.2.7 *2023-09-28*
  * **Updated** tables to include Unicode Specification 15.1.0.
  * Include ``bin``, ``docs``, and ``tox.ini`` in the source distribution

0.2.6 *2023-01-14*
  * **Updated** tables to include Unicode Specification 14.0.0 and 15.0.0.
  * **Changed** developer tools to use pip-compile, and to use jinja2 templates
    for code generation in `bin/update-tables.py` to prepare for possible
    compiler optimization release.

0.2.1 .. 0.2.5 *2020-06-23*
  * **Repository** changes to update tests and packaging issues, and
    begin tagging repository with matching release versions.

0.2.0 *2020-06-01*
  * **Enhancement**: Unicode version may be selected by exporting the
    Environment variable ``UNICODE_VERSION``, such as ``13.0``, or ``6.3.0``.
    See the `jquast/ucs-detect`_ CLI utility for automatic detection.
  * **Enhancement**:
    API Documentation is published to readthedocs.org.
  * **Updated** tables for *all* Unicode Specifications with files
    published in a programmatically consumable format, versions 4.1.0
    through 13.0

0.1.9 *2020-03-22*
  * **Performance** optimization by `Avram Lubkin`_, `PR #35`_.
  * **Updated** tables to Unicode Specification 13.0.0.

0.1.8 *2020-01-01*
  * **Updated** tables to Unicode Specification 12.0.0. (`PR #30`_).

0.1.7 *2016-07-01*
  * **Updated** tables to Unicode Specification 9.0.0. (`PR #18`_).

0.1.6 *2016-01-08 Production/Stable*
  * ``LICENSE`` file now included with distribution.

0.1.5 *2015-09-13 Alpha*
  * **Bugfix**:
    Resolution of "combining_ character width" issue, most especially
    those that previously returned -1 now often (correctly) return 0.
    resolved by `Philip Craig`_ via `PR #11`_.
  * **Deprecated**:
    The module path ``wcwidth.table_comb`` is no longer available,
    it has been superseded by module path ``wcwidth.table_zero``.

0.1.4 *2014-11-20 Pre-Alpha*
  * **Feature**: ``wcswidth()`` now determines printable length
    for (most) combining_ characters.  The developer's tool
    `bin/wcwidth-browser.py`_ is improved to display combining_
    characters when provided the ``--combining`` option
    (`Thomas Ballinger`_ and `Leta Montopoli`_ `PR #5`_).
  * **Feature**: added static analysis (prospector_) to testing
    framework.

0.1.3 *2014-10-29 Pre-Alpha*
  * **Bugfix**: 2nd parameter of wcswidth was not honored.
    (`Thomas Ballinger`_, `PR #4`_).

0.1.2 *2014-10-28 Pre-Alpha*
  * **Updated** tables to Unicode Specification 7.0.0.
    (`Thomas Ballinger`_, `PR #3`_).

0.1.1 *2014-05-14 Pre-Alpha*
  * Initial release to pypi, Based on Unicode Specification 6.3.0

This code was originally derived directly from C code of the same name,
whose latest version is available at
https://www.cl.cam.ac.uk/~mgk25/ucs/wcwidth.c::

 * Markus Kuhn -- 2007-05-26 (Unicode 5.0)
 *
 * Permission to use, copy, modify, and distribute this software
 * for any purpose and without fee is hereby granted. The author
 * disclaims all warranties with regard to this software.

.. _`Specification_from_pypi`: https://wcwidth.readthedocs.io/en/latest/specs.html
.. _`tox`: https://tox.wiki/en/latest/
.. _`prospector`: https://github.com/landscapeio/prospector
.. _`combining`: https://en.wikipedia.org/wiki/Combining_character
.. _`bin/`: https://github.com/jquast/wcwidth/tree/master/bin
.. _`bin/wcwidth-browser.py`: https://github.com/jquast/wcwidth/blob/master/bin/wcwidth-browser.py
.. _`Thomas Ballinger`: https://github.com/thomasballinger
.. _`Leta Montopoli`: https://github.com/lmontopo
.. _`Philip Craig`: https://github.com/philipc
.. _`PR #3`: https://github.com/jquast/wcwidth/pull/3
.. _`PR #4`: https://github.com/jquast/wcwidth/pull/4
.. _`PR #5`: https://github.com/jquast/wcwidth/pull/5
.. _`PR #11`: https://github.com/jquast/wcwidth/pull/11
.. _`PR #18`: https://github.com/jquast/wcwidth/pull/18
.. _`PR #30`: https://github.com/jquast/wcwidth/pull/30
.. _`PR #35`: https://github.com/jquast/wcwidth/pull/35
.. _`PR #82`: https://github.com/jquast/wcwidth/pull/82
.. _`PR #91`: https://github.com/jquast/wcwidth/pull/91
.. _`PR #97`: https://github.com/jquast/wcwidth/pull/97
.. _`PR #98`: https://github.com/jquast/wcwidth/pull/98
.. _`PR #100`: https://github.com/jquast/wcwidth/pull/100
.. _`Issue #101`: https://github.com/jquast/wcwidth/issues/101
.. _`jquast/blessed`: https://github.com/jquast/blessed
.. _`selectel/pyte`: https://github.com/selectel/pyte
.. _`thomasballinger/curtsies`: https://github.com/thomasballinger/curtsies
.. _`dbcli/pgcli`: https://github.com/dbcli/pgcli
.. _`prompt-toolkit/python-prompt-toolkit`: https://github.com/prompt-toolkit/python-prompt-toolkit
.. _`timoxley/wcwidth`: https://github.com/timoxley/wcwidth
.. _`wcwidth(3)`:  https://man7.org/linux/man-pages/man3/wcwidth.3.html
.. _`wcswidth(3)`: https://man7.org/linux/man-pages/man3/wcswidth.3.html
.. _`astanin/python-tabulate`: https://github.com/astanin/python-tabulate
.. _`janlelis/unicode-display_width`: https://github.com/janlelis/unicode-display_width
.. _`rspeer/python-ftfy`: https://github.com/rspeer/python-ftfy
.. _`alecrabbit/php-wcwidth`: https://github.com/alecrabbit/php-wcwidth
.. _`Text::CharWidth`: https://metacpan.org/pod/Text::CharWidth
.. _`bluebear94/Terminal-WCWidth`: https://github.com/bluebear94/Terminal-WCWidth
.. _`mattn/go-runewidth`: https://github.com/mattn/go-runewidth
.. _`grepsuzette/wcwidth`: https://github.com/grepsuzette/wcwidth
.. _`jquast/ucs-detect`: https://github.com/jquast/ucs-detect
.. _`Avram Lubkin`: https://github.com/avylove
.. _`nbedos/termtosvg`: https://github.com/nbedos/termtosvg
.. _`peterbrittain/asciimatics`: https://github.com/peterbrittain/asciimatics
.. _`aperezdc/lua-wcwidth`: https://github.com/aperezdc/lua-wcwidth
.. _`joachimschmidt557/zig-wcwidth`: https://github.com/joachimschmidt557/zig-wcwidth
.. _`fumiyas/wcwidth-cjk`: https://github.com/fumiyas/wcwidth-cjk
.. _`joshuarubin/wcwidth9`: https://github.com/joshuarubin/wcwidth9
.. _`python-cmd2/cmd2`: https://github.com/python-cmd2/cmd2
.. _`stratis-storage/stratis-cli`: https://github.com/stratis-storage/stratis-cli
.. _`ihabunek/toot`: https://github.com/ihabunek/toot
.. _`saulpw/visidata`: https://github.com/saulpw/visidata
.. _`pip-tools`: https://pip-tools.readthedocs.io/
.. _`sphinx`: https://www.sphinx-doc.org/
.. |pypi_downloads| image:: https://img.shields.io/pypi/dm/wcwidth.svg?logo=pypi
    :alt: Downloads
    :target: https://pypi.org/project/wcwidth/
.. |codecov| image:: https://codecov.io/gh/jquast/wcwidth/branch/master/graph/badge.svg
    :alt: codecov.io Code Coverage
    :target: https://app.codecov.io/gh/jquast/wcwidth/
.. |license| image:: https://img.shields.io/pypi/l/wcwidth.svg
    :target: https://pypi.org/project/wcwidth/
    :alt: MIT License
```

```
=====================
Unicode release files
=====================

This library aims to be forward-looking, portable, and most correct.
The most current release of this API is based on the Unicode Standard
release files:


``DerivedGeneralCategory-4.1.0.txt``
  *Date: 2005-02-26, 02:35:50 GMT [MD]*

``DerivedGeneralCategory-5.0.0.txt``
  *Date: 2006-02-27, 23:41:27 GMT [MD]*

``DerivedGeneralCategory-5.1.0.txt``
  *Date: 2008-03-20, 17:54:57 GMT [MD]*

``DerivedGeneralCategory-5.2.0.txt``
  *Date: 2009-08-22, 04:58:21 GMT [MD]*

``DerivedGeneralCategory-6.0.0.txt``
  *Date: 2010-08-19, 00:48:09 GMT [MD]*

``DerivedGeneralCategory-6.1.0.txt``
  *Date: 2011-11-27, 05:10:22 GMT [MD]*

``DerivedGeneralCategory-6.2.0.txt``
  *Date: 2012-05-20, 00:42:34 GMT [MD]*

``DerivedGeneralCategory-6.3.0.txt``
  *Date: 2013-07-05, 14:08:45 GMT [MD]*

``DerivedGeneralCategory-7.0.0.txt``
  *Date: 2014-02-07, 18:42:12 GMT [MD]*

``DerivedGeneralCategory-8.0.0.txt``
  *Date: 2015-02-13, 13:47:11 GMT [MD]*

``DerivedGeneralCategory-9.0.0.txt``
  *Date: 2016-06-01, 10:34:26 GMT*

``DerivedGeneralCategory-10.0.0.txt``
  *Date: 2017-03-08, 08:41:49 GMT*

``DerivedGeneralCategory-11.0.0.txt``
  *Date: 2018-02-21, 05:34:04 GMT*

``DerivedGeneralCategory-12.0.0.txt``
  *Date: 2019-01-22, 08:18:28 GMT*

``DerivedGeneralCategory-12.1.0.txt``
  *Date: 2019-03-10, 10:53:08 GMT*

``DerivedGeneralCategory-13.0.0.txt``
  *Date: 2019-10-21, 14:30:32 GMT*

``DerivedGeneralCategory-14.0.0.txt``
  *Date: 2021-07-10, 00:35:08 GMT*

``DerivedGeneralCategory-15.0.0.txt``
  *Date: 2022-04-26, 23:14:35 GMT*

``DerivedGeneralCategory-15.1.0.txt``
  *Date: 2023-07-28, 23:34:02 GMT*

``EastAsianWidth-4.1.0.txt``
  *Date: 2005-03-17, 15:21:00 PST [KW]*

``EastAsianWidth-5.0.0.txt``
  *Date: 2006-02-15, 14:39:00 PST [KW]*

``EastAsianWidth-5.1.0.txt``
  *Date: 2008-03-20, 17:42:00 PDT [KW]*
```

```
``EastAsianWidth-5.2.0.txt``
  *Date: 2009-06-09, 17:47:00 PDT [KW]*

``EastAsianWidth-6.0.0.txt``
  *Date: 2010-08-17, 12:17:00 PDT [KW]*

``EastAsianWidth-6.1.0.txt``
  *Date: 2011-09-19, 18:46:00 GMT [KW]*

``EastAsianWidth-6.2.0.txt``
  *Date: 2012-05-15, 18:30:00 GMT [KW]*

``EastAsianWidth-6.3.0.txt``
  *Date: 2013-02-05, 20:09:00 GMT [KW, LI]*

``EastAsianWidth-7.0.0.txt``
  *Date: 2014-02-28, 23:15:00 GMT [KW, LI]*

``EastAsianWidth-8.0.0.txt``
  *Date: 2015-02-10, 21:00:00 GMT [KW, LI]*

``EastAsianWidth-9.0.0.txt``
  *Date: 2016-05-27, 17:00:00 GMT [KW, LI]*

``EastAsianWidth-10.0.0.txt``
  *Date: 2017-03-08, 02:00:00 GMT [KW, LI]*

``EastAsianWidth-11.0.0.txt``
  *Date: 2018-05-14, 09:41:59 GMT [KW, LI]*

``EastAsianWidth-12.0.0.txt``
  *Date: 2019-01-21, 14:12:58 GMT [KW, LI]*

``EastAsianWidth-12.1.0.txt``
  *Date: 2019-03-31, 22:01:58 GMT [KW, LI]*

``EastAsianWidth-13.0.0.txt``
  *Date: 2029-01-21, 18:14:00 GMT [KW, LI]*

``EastAsianWidth-14.0.0.txt``
  *Date: 2021-07-06, 09:58:53 GMT [KW, LI]*

``EastAsianWidth-15.0.0.txt``
  *Date: 2022-05-24, 17:40:20 GMT [KW, LI]*

``EastAsianWidth-15.1.0.txt``
  *Date: 2023-07-28, 23:34:08 GMT*

``emoji-variation-sequences-12.0.0.txt``
  *Date: 2019-01-15, 12:10:05 GMT*

``emoji-variation-sequences-15.1.0.txt``
  *Date: 2023-02-01, 02:22:54 GMT*
```

```
.. _Specification:

=============
Specification
=============

This document defines how the wcwidth library measures the printable width
of characters of a string.

Width of -1
-----------

The following have a column width of -1 for function :func:`wcwidth.wcwidth`

- ``C0`` control characters (`U+0001`_ through `U+001F`_).
- ``C1`` control characters and ``DEL`` (`U+007F`_ through `U+00A0`_).

If any character in sequence contains ``C0`` or ``C1`` control characters, the final
return value of of :func:`wcwidth.wcswidth` is -1.

Width of 0
----------

Any characters defined by category codes in `DerivedGeneralCategory.txt`_ files:

- 'Me': Enclosing Combining Mark, aprox. 13 characters.
- 'Mn': Nonspacing Combining Mark, aprox. 1,839 characters.
- 'Mc': Spacing Mark, aprox. 443 characters.
- 'Cf': Format control character, aprox. 161 characters.
- 'Zl': `U+2028`_ LINE SEPARATOR only
- 'Zp': `U+2029`_ PARAGRAPH SEPARATOR only
- 'Sk': Modifier Symbol, aprox. 4 characters of only those where phrase
  ``'EMOJI MODIFIER'`` is present in comment of unicode data file.

The NULL character (`U+0000`_).

Any character following ZWJ (`U+200D`_) when in sequence by
function :func:`wcwidth.wcswidth`.

Hangul Jamo Jungseong and "Extended-B" code blocks, `U+1160`_ through
`U+11FF`_ and `U+D7B0`_ through `U+D7FF`_.


Width of 1
----------

String characters are measured width of 1 when they are not
measured as `Width of 0`_ or `Width of 2`_.

Width of 2
----------

Any character defined by East Asian Fullwidth (``F``) or Wide (``W``)
properties in `EastAsianWidth.txt`_ files, except those that are defined by the
Category codes of Nonspacing Mark (``Mn``) and Spacing Mark (``Mc``).

Any characters of Modifier Symbol category, ``'Sk'`` where ``'FULLWIDTH'`` is
present in comment of unicode data file, aprox. 3 characters.

Any character in sequence with `U+FE0F`_ (Variation Selector 16) defined by
`emoji-variation-sequences.txt`_ as ``emoji style``.


.. _`U+0000`: https://codepoints.net/U+0000
.. _`U+0001`: https://codepoints.net/U+0001
.. _`U+001F`: https://codepoints.net/U+001F
.. _`U+007F`: https://codepoints.net/U+007F
.. _`U+00A0`: https://codepoints.net/U+00A0
.. _`U+1160`: https://codepoints.net/U+1160
.. _`U+11FF`: https://codepoints.net/U+11FF
.. _`U+200D`: https://codepoints.net/U+200D
.. _`U+2028`: https://codepoints.net/U+2028
.. _`U+2029`: https://codepoints.net/U+2029
.. _`U+D7B0`: https://codepoints.net/U+D7B0
.. _`U+D7FF`: https://codepoints.net/U+D7FF
.. _`U+FE0F`: https://codepoints.net/U+FE0F
.. _`DerivedGeneralCategory.txt`:
https://www.unicode.org/Public/UCD/latest/ucd/extracted/DerivedGeneralCategory.txt
.. _`EastAsianWidth.txt`: https://www.unicode.org/Public/UCD/latest/ucd/EastAsianWidth.txt`
.. _`emoji-variation-sequences.txt`: https://www.unicode.org/Public/UCD/latest/ucd/emoji/emoji-
variation-sequences.txt
```

```
==========
Public API
==========

This package follows SEMVER_ rules.  Therefore, for the functions of the below
list, you may safely use version dependency definition ``wcwidth<2`` in your
requirements.txt or equivalent. Their signatures will never change.

.. autofunction:: wcwidth.wcwidth

.. autofunction:: wcwidth.wcswidth

.. autofunction:: wcwidth.list_versions

.. _SEMVER: https://semver.org

===========
Private API
===========

These functions should only be used for wcwidth development, and not used by
dependent packages except with care and by use of frozen version dependency,
as these functions may change names, signatures, or disappear entirely at any
time in the future, and not reflected by SEMVER_ rules!

If stable public API for any of the given functions is needed, please suggest a
Pull Request!

.. autofunction:: wcwidth._bisearch

.. autofunction:: wcwidth._wcversion_value

.. autofunction:: wcwidth._wcmatch_version
```