The mahjong package*

Daniel Schmitz

me@schmytzi.com

January 6, 2025

Contents

Abstract

The mahjong package provides a LaTeX 2ε and LaTeX 3 interface for typesetting mahjong tiles using an extended version of MPSZ algebraic notation. Features include spaces, rotated, blank, and concealed tiles, as well as red fives. The size of the mahjong tiles and their symbols can be controlled using package options and optional arguments. It is primarily aimed at Riichi (aka. Japanese) Mahjong but can be used to typeset any style of mahjong, save for flower tiles.

*This document corresponds to mahjong v1.1, dated 2025/01/06

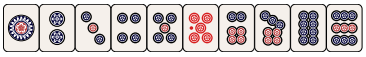
1 Introduction

Mahjong is a tile-based game originating from China which is popular in East and South-East Asia and has since spread throughout the world. The mahjong package provides an interface for typesetting mahjong tiles and hands using MPSZ algebraic notation. This documentation assumes familiarity with the game in general but none of its many styles. Nonetheless, some basic terms will be defined because of differing vocabulary among players.

2 Mahjong Tiles

2.1 Suited Tiles

The suited tiles are referred to as follows:

Suit	Tiles
Bamboo	
Dots	
Character	

Suited tiles are referred to using the pattern ⟨*value*⟩ ⟨*suit*⟩. For instance,  is called *4 Bamboo*.

2.2 Honor Tiles

This documentation refers to the seven honor tiles as follows:

Winds

 East Wind (E)

 South Wind (S)

 West Wind (W)

 North Wind (N)

Dragons

 White Dragon

 Green Dragon

 Red Dragon

Table 1: MPSZ notation reference. Each tile is identified by its column's number and its row's letter.

	0	1	2	3	4	5	6	7	8	9
s										
p										
m										
z										

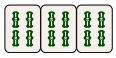
3 MPSZ Algebraic Notation

3.1 Standard Notation

MPSZ notation assigns each tile an identifier consisting of a digit and a letter (table 1). For suited tiles, the digit corresponds to the tile's value and the letter to its suit, Bamboo (`s`), Dots (`p`) or Character (`m`). For instance, `2m` identifies  (2 Character). The only exception ot this rule are red fives, whose numeric value is 0. Red 5 Bamboo, for example, has identifier `0s`. Honor tiles are assigned the "suit" `z`, with `1z` – `4z` corresponding to E, S, W and N, and `5z` – `7z` to the white, green and red dragon, respectively.

Collections of tiles, such as melds or hands, are represented by concatenating the identifiers of the tiles they comprise. For instance, `3s4s5s` corresponds to  . Groups of tiles sharing the same suit can be abbreviated by omitting all but the last suit identifier. The above example can also be expressed as `345s`. Spaces are ignored and the notation is case-insensitive.

3.2 Extensions

Spaces. Spaces can be inserted using `-`: `444s-567s` produces   .

Concealed Tiles.  Concealed (or face-down) tiles are represented by `X`. They don't need a suit identifier and don't act as one. `123s X 456s` and `123 X 456s` are therefore equivalent.

Blank Tiles.  Blank or unknown tiles are represented by `?`. Just like concealed tiles, they don't change the current suit. `123s ? 456s` and `123 ? 456s` are equivalent, for instance.

Rotation. Inserting an apostrophe (`'`) rotates the *preceeding* tile counter-clockwise. For instance, `6'66m` is rendered as  . This can only be done once per tile, i.e. it is not possible to rotate them 180° or 270°. When you want to rotate the last tile of a group, it doesn't matter whether the apostrophe appears before or after the suit, so `77'm` and `77m'` are equivalent.

Rotation and Stacking. Quotes (`"`) cause the *preceeding* tile to be rendered as two rotated and stacked tiles. For instance, `77"7z` produces  .

4 Typesetting Mahjong Tiles in Your Document

`\mahjong` The main interface is `\mahjong [`⟨*height*⟩`] [`⟨*scale*⟩`] {`⟨*hand*⟩`}`. ⟨*hand*⟩ refers to a tile sequence in MPSZ notation as discussed above. ⟨*height*⟩ specifies the height of the rendered mahjong tiles. ⟨*scale*⟩ specifies the fraction of vertical space that the tiles' symbols should occupy. The value should be between 0 and 1. If an optional argument is not given, the default (which can be set through a package argument) will be used.

`\mahjong_typeset_hand:n` `\mahjong_typeset_hand:x` `\l_mahjong_tile_height` `\g_mahjong_default_height` `\l_mahjong_tile_scale` `\g_mahjong_default_scale`

The LaTeX 3 interface for rendering mahjong tiles are `\mahjong_typeset_hand:n` and its variants. This macro accepts the hand to be rendered in MPSZ notation. The height can be specified by setting `\l_mahjong_tile_height` and the default height is saved in `\g_mahjong_default_height`. The scale of the tiles' symbols can be changed by setting `\l_mahjong_tile_scale` and the default scale is saved in `\g_mahjong_default_scale`.

4.1 Package Options

height The default height can be set using the package's `height` parameter. For instance, `\usepackage[height=2\baselineskip]{mahjong}` sets the default size of mahjong tiles to double the value of `\baselineskip` in the context they are rendered in.

scale The default scale can be set using the package's `scale` parameter. It should ideally be set to a constant to ensure consistent typesetting. The default is 0.75, i.e. the symbols take up 85% of the tiles' vertical space.

5 Acknowledgments

The mahjong tiles used in this package were created by GitHub user FluffyStuff. The original repository is FluffyStuff/riichi-mahjong-tiles, used under CC-BY Version 4.0.

```
⟨*pkg⟩
⟨@@=mahjong⟩
\NeedsTeXFormat{LaTeX2e}[2019/10/01]
\RequirePackage{expl3}
\ProvidesExplPackage{mahjong}{2025/01/06}{1.1}{Typeset Mahjong Hands}
\RequirePackage{xparse}
```

```
\RequirePackage{l3keys2e}
\RequirePackage{graphicx}
\RequirePackage{stackengine}

\msg_new:nnnn {mahjong} {invalid_token}
{Token ~ #1 ~ is ~ not ~ valid ~ in ~ MPSZ ~ notation}
{Valid ~ tokens ~ are ~ digits ~ 0-9, ~ m, ~ p, ~ s, ~ z, ~ x, ~ -, ~ ?, ~ ', ~ and ~ " }
\msg_new:nnnn {mahjong} {unknown_tile}
{I~don't~know~tile~#1.}
{Please~check~the~documentation~for~recognized~tiles.}
\msg_new:nnnn {mahjong} {unknown_orientation}
{Orientation ~ #1 ~ is ~ unknown}
{This ~ should ~ not ~ happen.~ Please ~ create ~ a ~ bug ~ report.}

\keys_define:nn {mahjong} {
    height .dim_gset:N = \g_mahjong_default_height,
    scale .int_gset:N = \g_mahjong_default_scale
}

% Identifiers for all suits
\cs_new:Npn \c__mahjong_suits_tl {mpsz}
% Allowed tokens
\cs_new:Npn \c__mahjong_allowed_tokens_tl {0123456789mpsz-?x'"}

% Variables have to be declared globally
\tl_new:N \l__mahjong_suit_tl
\tl_new:N \l__mahjong_tiles_tl
\tl_new:N \l__mahjong_reversed_tl
\tl_new:N \l__mahjong_hand_tl
\tl_new:N \l__mahjong_current_suit_tl
\tl_new:N \l__mahjong_current_group_tl
\tl_new:N \l__mahjong_current_char


\dim_set:Nn \g_mahjong_default_height {\baselineskip}
\dim_new:N \l_mahjong_tile_height

\fp_set:Nn \g_mahjong_default_scale {0.75}
\fp_new:N \l_mahjong_tile_scale

\dim_new:N \l__mahjong_symbol_height
\dim_new:N \l__mahjong_baseline_offset

\int_new:N \l__mahjong_tile_orientation_int
\seq_new:N \l__mahjong_tile_maps_seq
\str_new:N \l__mahjong_file_path_str


\ProcessKeysPackageOptions{mahjong}
```

`\__mahjong_make_tile:nn`
`\__mahjong_make_tile:VV`
`\__mahjong_make_tile:xV`
`\__mahjong_make_tile:nV`

Inserts a mahjong tile into the input stream. This functions only handles tiles that use the front background and have a forground, i.e. regular and blank tiles.

```
\cs_set:Npn \__mahjong_make_tile:nn #1#2 {
```

```
    \file_if_exist:nTF {#1} {
        \int_case:nnF {#2} {
            {0} {
                \stackinset{c}{0pt}{c}{0pt}{
                    \includegraphics[
                        angle=0,
                        height=\l__mahjong_symbol_height]
                    {#1}
                }{
                    \includegraphics[
                        angle=0,
                        height=\l_mahjong_tile_height]
                    {tiles/mahjong-Front.pdf}
                }
            } {1} {
                \stackinset{c}{0pt}{c}{0pt}{
                    \includegraphics[
                        angle=90,
                        width=\l__mahjong_symbol_height]
                    {#1}
                }{
                    \includegraphics[
                        angle=90,
                        width=\l_mahjong_tile_height]
                    {tiles/mahjong-Front.pdf}
                }
            } {2} {
                % Stack 2 rotated tiles on top of each other.
                \stackon [0pt] {
                    \__mahjong_make_tile:nn {#1} {1}
                } {
                    \__mahjong_make_tile:nn {#1} {1}
                }
            }
        } {
            \msg_fatal:nnx {mahjong} {unknown_orientation} {#2}
        }
    } {
        \msg_error:nnx {mahjong} {unknown_tile} {#1}
    }
}

\cs_generate_variant:Nn \__mahjong_make_tile:nn {VV, xV, nV}
```

(*End definition for* `\__mahjong_make_tile:nn`.)

`\mahjong_typeset_hand:n`
`\mahjong_typeset_hand:x`

Parses and typesets a mahjong hand in MPSZ notation. Set `\l_mahjong_tile_height` to control the tiles' size and `\l_mahjong_tile_scale` to control the size of the symbol relative to the tile.

```
% Parses a full hand
\cs_set:Npn \mahjong_typeset_hand:n #1 {
    % Set computed dimensions for symbol height and baseline offset
    \dim_set:Nn \l__mahjong_symbol_height {\fp_to_decimal:n {\l_mahjong_tile_scale}\l_mahjong
```

```
    \dim_set:Nn \l__mahjong_baseline_offset {(\l_mahjong_tile_height - \l__mahjong_symbol_hei
    % Start sequence processing
    \tl_set:Nx \l__mahjong_hand_tl {\text_lowercase:n {#1}}
    % MPSZ notation is easier to parse right-to-left, so reverse string
    % There is no string reversal function but we can reverse token lists
    % Token lists and strings are freely convertible between each other
    \tl_set:Nx \l__mahjong_reversed_tl {\tl_reverse:V \l__mahjong_hand_tl}
    \tl_map_variable:NNn \l__mahjong_reversed_tl \l__mahjong_current_char {
        % Check if we recognize the current token
        \exp_args:NVV \tl_if_in:nnF \c__mahjong_allowed_tokens_tl \l__mahjong_current_char {
            \msg_error:nnx {mahjong} {invalid_token} {\l__mahjong_current_char}
        }
        \exp_args:NVV \tl_if_in:nnTF \c__mahjong_suits_tl \l__mahjong_current_char {
            % If we've found a suit identifier, change the current suit
            \tl_set:NV \l__mahjong_current_suit_tl \l__mahjong_current_char
        } {
            \str_case:VnF \l__mahjong_current_char {
                {'} {
                    % An apostrophe indicates that the next tile is rotated
                    \int_set:Nn \l__mahjong_tile_orientation_int {1}
                }
                {"} {
                    % Quotes mean the next tile is actually 2 rotated tiles stacked on top of
                    \int_set:Nn \l__mahjong_tile_orientation_int {2}
                }
            } {
                % Default case: We've got a complete tile identifier
                % Turn it into a property list
                \prop_clear:N \l_tmpa_prop
                \prop_put:NnV \l_tmpa_prop {suit} \l__mahjong_current_suit_tl
                \prop_put:NnV \l_tmpa_prop {face} \l__mahjong_current_char
                \prop_put:NnV \l_tmpa_prop {orientation} \l__mahjong_tile_orientation_int
                % Add it to the beginning of the sequence (we are parsing in reverse)
                \seq_put_left:NV \l__mahjong_tile_maps_seq \l_tmpa_prop
                \int_set:Nn \l__mahjong_tile_orientation_int {0}
            }
        }
    }
    % Typesetting begins here. Sequence is in correct order
    \raisebox{-\l__mahjong_baseline_offset}{
        \seq_map_variable:NNn \l__mahjong_tile_maps_seq \l_tmpa_prop {
            \prop_get:NnN \l_tmpa_prop {face} \l_tmpa_tl
            \prop_get:NnN \l_tmpa_prop {orientation} \l_tmpa_int
            \str_case:VnF \l_tmpa_tl {
                {-} {
                    % If the current face is a dash, insert a space
                    \includegraphics[height=\l_mahjong_tile_height]{tiles/mahjong-
Space.pdf}
                } {x} {
                    % Insert a flipped tile
                    \int_case:nn {\l_tmpa_int} {
                        {0} { % Upright
                            \includegraphics[
                            angle=0,
```

```
                            height=\l_mahjong_tile_height]
                            {tiles/mahjong-Back.pdf}
                        } {1} { % Rotated
                            \includegraphics[
                            angle=90,
                            width=\l_mahjong_tile_height]
                            {tiles/mahjong-Back.pdf}
                        } {2} { % Rotated and stacked
                            \stackon [0pt] {
                                \includegraphics[
                                    angle=90,
                                    width=\l_mahjong_tile_height]
                                    {tiles/mahjong-Back.pdf}
                            } {
                                \includegraphics[angle=90,
                                width=\l_mahjong_tile_height]
                                {tiles/mahjong-Back.pdf}
                            }
                        }
                    }
                } {?} { % Blank tile
                    \__mahjong_make_tile:nV {tiles/mahjong-Blank.pdf} \l_tmpa_int
                }
            } { % Any other tile identified by a code.
                \__mahjong_make_tile:xV {tiles/mahjong-\l_tmpa_tl\prop_item:Nn \l_tmpa_prop {
            }
        }
    }
    % Clear the list for the next invocation
    \seq_clear:N \l__mahjong_tile_maps_seq
}

% Have TeX automatically expand the argument for us
\cs_generate_variant:Nn \__mahjong_typeset_hand:n {x}
```

(*End definition for* `\mahjong_typeset_hand:n`*. This function is documented on page* **??**.)

`\mahjong` This is the only LaTeX 2$_\varepsilon$ macro in this package. It typesets a mahjong hand.

```
\NewDocumentCommand{\mahjong}{O{\g_mahjong_default_height} O{\g_mahjong_default_scale} m}{
    \dim_set:Nn \l_mahjong_tile_height {#1}
    \fp_set:Nn \l_mahjong_tile_scale {#2}
    \mahjong_typeset_hand:n {#3}
}
⟨/pkg⟩
```

(*End definition for* `\mahjong`*. This function is documented on page* **??**.)

Change History

v0.5
General: First working version, minimal error handling 1
v0.9
General: Fully functional 1
v1.0
General: First complete release 1
v1.0.1
General: Added package prefix to filenames 1
v1.1
General: Added feature to control size of symbols. Adjusted vertical alignment. 1