

The **ONEDOWN** package

www.ctan.org/pkg/onedown

Jacob Wiersma

jack46@schuli-wirsi.de

v1.6 from 2021/04/27

Dealer: S ♠ A32											
Vul: N-S ♥ QJ10											
♦ KQJ109											
♣ QJ											
♠ 8765			♠ 94								
♥ AK	<table border="1" style="border-collapse: collapse; width: 40px; height: 40px; margin: auto;"> <tr><td></td><td style="text-align: center;">N</td><td></td></tr> <tr><td style="text-align: center;">W</td><td></td><td style="text-align: center;">E</td></tr> <tr><td></td><td style="text-align: center;">S</td><td></td></tr> </table>		N		W		E		S		♥ 5432
	N										
W		E									
	S										
♦ 732			♦ 654								
♣ 10987			♣ 6543								
♠ KQJ10											
♥ 9876											
♦ A8											
♣ AK2											
South	West	North	East								
1NT ^a	p	4NT ^b	p								
6NT	X	All pass									
<hr/>											
^a 15-17											
^b quantitative											
Nº	Lead	2nd	3rd	4th	N	E					
					S	W					
1	W: ♥K	10	2	6	0	1					
2	W: ♥A	J	3	7	0	2					

Abstract

This package implements commands and environments to typeset various bridge diagrams, with or without a bidding sequence. It offers following features:

- It is possible to use an own font and/or a font-size with which the diagrams will be typeset independently from the main font used in the document. This also allows an easy production of overhead slides and for digital projection. Different fonts can be used for e.g. the bidding diagram, its header, the compass, the hands etc. Most diagrams can be centered both horizontally and vertically.
- A special feature is the automated check on consistency of suits and hands. If a hand holds more than 13 cards an error is printed, if there are less then 13 cards a warning. If a suit over the 4 hands has more than 13 cards or if a card appears more than once an error like `Error: Card ♠ 8 occurs 2 times` is printed. These warnings and error messages are controlled by the `err` and `warn` options, e.g. when loading the package.
- The output of the implemented bridge terms like `Double` (which you get by calling the command `\double`) are multilingual and get translated automatically. When the language `german` is active the command `\double` would produce `Kontra`. Also the basic symbols like A(ce), K(ing), Q(ueen) and J(ack) are multilingual. So `♦ AKQJ` would automatically become `♦ AKDB` in a German text.
- It is possible to add annotations to a card diagram, like the board number, the dealer or the vulnerability etc. on several positions in the diagram (if a board number is given, the dealer and vulnerability are computed automatically). One can also add explanations to the bidding diagram, as well as the real world names of the bidders.
- There are two specials: a command to typeset a quiz with answers, and an environment to typeset the sequence of playing tricks, where also the total number of tricks won by each side is calculated and displayed.

Contents

1	Preface	5
2	Introduction	5
3	Usage	7
3.1	Initialization	7
3.1.1	Requirements	7
3.1.2	Loading the package	7
3.1.3	Options	8
3.1.4	Languages & Dictionaries	9
4	User Commands	10
4.1	Overview	10
4.1.1	The Compass	11
4.1.2	Hooks	11
4.2	Basic Symbols	12
4.2.1	Header of the Bidding Table	14
4.3	The Card Diagrams	14
4.3.1	The hands of the players	14
4.3.2	The single hand	15
4.3.3	Only one suit	16
4.3.4	Showing Card Diagrams	17
4.3.5	Showing Card Diagrams with Bidding	18
4.3.6	Diagram Conditions	18
4.3.7	Sizing and Fonts	19
4.4	Misc	21
4.4.1	Honor Cards	22
4.4.2	Variants of Basic Symbols	22
4.4.3	Conditions in Diagrams	24
4.4.4	Annotations in Bidding Diagrams	25
4.4.5	Specials	26
4.4.6	Re-Initialization	26
4.5	Environments	28
5	Final Remarks	30
5.1	Known Bugs	30
5.2	ToDo	31
5.3	Acknowledgements	31

6	Implementation	31
6.1	Preliminaries	31
6.1.1	Packages we depend upon	31
6.1.2	Options	32
6.1.3	Misc	32
6.1.4	Variables	34
6.1.5	Booleans, Saveboxes, Lengths, Counters and Registers	35
6.1.6	Fonts	37
6.2	Bridge Basic Terms	39
6.2.1	Suit Symbols	39
6.2.2	Names of Directions and Axes	48
6.2.3	Non-Bid Calls	51
6.2.4	Bidding Diagrams	53
6.2.5	Diagram Hands	55
6.2.6	A Single Hand	57
6.2.7	Suits	60
6.2.8	Card Diagrams	63
6.2.9	The Compass	70
6.2.10	Diagram Conditions	76
6.3	The Bidding Environments	82
6.3.1	Special Columntypes	82
6.3.2	The Hidden Implementation	83
6.4	The User Environments	87
6.4.1	Bidding	87
6.4.2	Play	89
6.5	Card Diagrams with Bidding	93
6.6	The Expert Quiz	94
6.7	Resetting the Game	95
6.8	Error Handling	99
6.8.1	Consistency Checks	99
6.8.2	Controlling Messages	100
6.9	Misc Bridge Terms	101
6.9.1	Honour Cards	101
6.9.2	Vulnerability	103
6.9.3	Diagram Annotations	104
6.9.4	Point Units	108
6.9.5	Forcings	109
6.10	Initialization	112
7	References	114

8 Change History	114
9 Index	119
10 List of User Commands	126

1 Preface

I am neither a good bridge player nor a good package writer. But I like to read about bridge and when I write about bridge (sometimes a funny short story, sometimes some training exercises for beginners) I do feel the need for an appropriate tool to support that. Surprisingly enough, there exists no comprehensive package on CTAN for typesetting bridge diagrams. For all those people who feel more or less the same as I do, there is this package called **ONEDOWN**. As some say: *one down is good bridge*¹, I hope that **ONEDOWN** is a good bridge package.

You can generate this documentation by running

```
pdflatex --shell-escape onedown.dtx
makeindex -s gind.ist onedown.idx
makeindex -s gglo.ist -o onedown.gls onedown.glo
pdflatex --shell-escape onedown.dtx
pdflatex --shell-escape onedown.dtx
```

Use:

```
pdflatex --shell-escape '\AtBeginDocument{\NoColortrue}\input{onedown.dtx}'
```

as last run to get a PDF for printing on a monochrome printer. The `--shell-escape` flag is necessary to generate the list of user commands. If you think this is too dangerous, then run `pdflatex` without this flag and you will get the documentation without the list of user command (of about 1 page). In any case you'll find a multi-page reference overview of all commands in `onedown-ref.pdf`.

2 Introduction

There must be a lot of bridge players who also use L^AT_EX to typeset their documents. And it is almost incredible that on CTAN there exist no modern package with a decent documentation that supports this.

¹please, don't discuss the truth-value of this statement with me

In 1990 Kees van der Laan [1] published an article in TUGBoat² in which he describes how well the T_EX-machinery is able to produce beautiful bridge diagrams. Based on this article and examples, Johannes Braams put these commands together in a style file³ and added more. Some time later René Steiner and Thomas Hof produced the `bridge-i` and the `kibitzer`⁴ style files, in which they made a lot of enhancements. Also others made some efforts in this direction.⁵

Around 2005 I used these style files for some tiny projects. The quality of the output was splendid. Putting the text and diagrams together was not always easy and the documentation was poor. In 2015, after a long pause, I had to produce bridge texts again. I enhanced some of the existing stuff and made ad hoc changes in the code, which led to smaller and greater catastrophes. Summer 2016 I decided to write a new package, based on the work of the previously mentioned persons. I called it **ONEDOWN**. The central goal was to offer a user friendly package with detailed documentation. For example you don't need to say `\setlength{\handskip}{5mm}` or `\def\handskip{5mm}` but rather call the command `\handskip{1em}`. Not only the call is somewhat friendlier, but more important, by setting the width in terms of the font used, it will automatically adapt its size accordingly to font and font-size changes. **ONEDOWN** features:

- Sizing of diagrams relative to font and font-size.
- The font-sizes of the diagrams and text are independent.
- Automated translation of all important bridge terms.
- Diagrams can optionally contain information about the dealer, who is vulnerable etc.

The **ONEDOWN** package is designed to be used for typesetting texts that have to do with the game of bridge. It provides not only simple commands like `\sp` which produces the spade symbol ♠. Also complete card diagrams with the hands of the **North**, **East**, etc. player

²<http://tug.org/TUGboat/Articles/tb11-2/tb28laan.pdf>

³`bridge.sty`, last version v1.7c, 1994/12/20

⁴both v1.0, 1995/04/06

⁵Antony Lee released his package `bridge` in 2012 and Gordon Bower his package `grbbridge` in 2013. Both are very interesting but offer only limited features and are not on CTAN at the time of writing. See <http://www.bridgebase.com/forums/topic/51967-latex-package-for-typesetting-bridge-related-stuff/>

can be defined in several ways. One can select which hands are to be shown. Bidding diagrams can be shown stand-alone or in connection with one or more hands. One can add annotations to any bid in a bidding diagrams.

3 Usage

3.1 Initialization

3.1.1 Requirements

The package **ONEDOWN** depends on several other packages, such as `ifthen`, `translator` or `xspace`. All these packages get loaded automatically if not already used in your document. For a complete overview of all required packages, refer to page 31. All the packages are loaded without any option, so the risk of an option clash should be low: Just load your package with options *before* **ONEDOWN**.

Furthermore, for the several languages that **ONEDOWN** supports, there are the dictionary files with translations of the specific bridge terms. These dictionaries follow the naming convention: `ODw-<Language>.dict` and are included in the bundle. The name of the `<Language>` is generally the same as the name that you use as option for `babel`, but starting with a capital.

Should you make a dictionary for a language that is not provided yet, or have corrections for an existing one, please send it to the maintainer, so it can be added to the bundle.

3.1.2 Loading the package

Simply say `\usepackage{onedown}` in the preamble of your document if you want to load **ONEDOWN** with its default settings.

Warning: **ONEDOWN** loads all necessary `ODw`-dictionaries automatically. In order to know which languages must be loaded, these must be specified *before* package `onedown` is loaded. In general this means that if you use `babel` (or `polyglossia`) you must load it before package **ONEDOWN**. If for some reason you cannot or do not want to do that, you can load any `ODw`-dictionary if you put the command: `\uselanguage{<ODw-Language>}` in your preamble, provided that the dictionary is in the `TEX`-path. For a discussion about the caveats of using e.g. `babel`, refer to section 3.1.4.

3.1.3 Options

To change the behaviour of **ONEDOWN** one can load the package with certain options: `\usepackage[<options>]{onedown}`. Of course this 'option loading' takes place in the preamble. But it is also possible to set (or change) options within the document by calling the macro `\setdefaults`. This macro uses the same `key=val` syntax as is used for the options and offers some more keys that cannot be used when loading the package. Refer to page 27 for details.

As said before, the package loads its options using the `key=val` syntax. These options deal with:

`colors colors=0|1|2|3|4A|4B`

The color in which the card symbols will be printed. The color options are **0** (black only), **1** (black and white) **2** (black and red), **3** (grey (for special effects)), **4A** (green, orange, red and blue), **4B** (black, orange, red and green). We also defined some synonyms, as shown in the table below.

Thus loading the package with `\usepackage[colors=X]{onedown}` will print

x=0: ♣, ♦, ♥ and ♠ (synonyms: `mono`, `black`)
x=1: ♣, ◇, ♡ and ♠ (synonyms: `b+w`)
x=2: ♣, ♦, ♥ and ♠ (synonyms: `b+r`)
x=3: ♣, ♦, ♥ and ♠ (synonyms: `gray`, `grey`)
x=4A: ♣, ♦, ♥ and ♠ (synonyms: `4a`, `fourA`)
x=4B: ♣, ♦, ♥ and ♠ (synonyms: `4b`, `fourB`)

The default is `colors=2` for printing in black and red. Please note, that `colors=3` is not meant to typeset the whole document. You can use it if you want e.g. to repeat something and want it to be less apparent.

`err, warn err=on|off warn=|on|off`

These options regulate which messages are to be output. These messages have to do with the consistency of cards in a suit, in a hand or combined hands. It is an error when a hand has more than 13 cards, or when the same card occurs twice or more in a hand or a deal. With the option `err=on` (which is the default) these error messages appear as output. With `err=off` you can suppress that. On the other hand when a suit has less than 13

cards, this must not necessarily be wrong. Maybe only some cards are to be shown, e.g. in an example concerning a finesse. Or when only e.g. E-W hands are concerned, not all cards of the deck will be specified. These situations will be caught by setting `warn=on`. To suppress these spurious warnings use `warn=off`, which is the default. Synonyms for `on` are `1` and `true`. Synonyms for `off` are `0` and `false`. This also applies for other keys that do not control a package option.

3.1.4 Languages & Dictionaries

ONEDOWN uses the `translator` package to automatically translate often appearing bridge terms like e.g. *declarer*. It does so by looking up these terms in the special `ODw`-dictionary for the active language. The current **ONEDOWN** version supports English, German, Dutch, French, Spanish, most Scandinavian languages and Turkish. Some dictionaries may not be complete or may contain errors, please send corrections/additions to the maintainer.

In order to make `translator` do its job (automatically select the right translation when the current language switches to another), it is necessary to specify the languages with the `documentclass`, and not with `babel`!

Warning for people using active characters.

Some language packages fiddle around making characters active. This can have unexpected influence on **ONEDOWN**. The `=` sign is used when loading the package `onedown` to specify options. It appears also in calls like `\setdefaults{warn=on}`. We also use the following characters as tokens for optional arguments with the meaning as shown in this list:

- * to center diagrams or print a long or capitalized text
- ! special action like short names of vertical layout
- to hide what would normally be shown
- + to show what would normally be hidden

To give you an idea what e.g. `babel` can cause we cite from *The Turkish style for babel*:

Turkish typographic rules specify that a little 'white space' should be added before the characters `'.'`, `'!`' and `'='`. In

order to insert this white space automatically these characters are made `\active`, so they have to be treated in a special way.

So `babel-Turkish` makes the equal sign and exclamation mark active. This leads to errors when you call e.g. `\setdefaults{warn=on}` or `\hand!`. If you do not need any character to be active, then load this language with `\usepackage[turkish,shorthands=]{babel}`.

You can also put `\PassOptionsToPackage{shorthands=off}{babel}` above your `documentclass` line. Here is a complete example:

```
\PassOptionsToPackage{shorthands=off}{babel}
\documentclass[a4paper,fontsize=11pt,%
  dutch,%
  german,%
  english,% this is the default language
]{scrartcl}
\usepackage{babel}
```

If you do need the shorthand then you must disable it every time you have to use e.g. the `=` character as a normal character by:

```
\shorthandoff{=}% Make '=' not active any more
\setdefaults{warn=on}
\shorthandon{=}% Restore '=' to active again
```

4 User Commands

4.1 Overview

In the next sections we give a short description of all the user commands and environments that are defined in **ONEDOWN**. The commands marked with **ML** are multilingual. I.e. the text they typeset gets translated automatically into the active language.

In order to make sure that the example diagrams do not disturb the page layout of this document too much, we scaled them down to `footnotesize`.

Sometimes the output of a command is shown as an example. This output is framed in this document like `\boxed{this}`, just to recognize it easily as an output example. In an accompanying file⁶ with examples one

⁶onedown-examples.pdf

can find in more detail how these commands are used and what they produce.

We have loaded the package `ONEDOWN` with the default option for colors, giving us black and `red`. Furthermore, when we describe macros, we use a colored frame that also shows the output of the command. Some commands have optional tokens that produce an output that differs from the naked version. These optional tokens appear in a different background color like `*!`. If an output of a macro is shown, then the output of tokenized calls is shown in parenthesis to demonstrate the difference.

4.1.1 The Compass

The compass $\begin{array}{|c|} \hline N \\ \hline W \quad E \\ \hline S \\ \hline \end{array}$ is not available as a user command itself, but it is used in all user commands that draw a card diagram. It has some special features.

- It can mark the dealer (**North**) $\begin{array}{|c|} \hline N \\ \hline W \quad E \\ \hline S \\ \hline \end{array}$
- It can write the vulnerable side (**North–South**) in red $\begin{array}{|c|} \hline \underline{N} \\ \hline W \quad E \\ \hline \underline{S} \\ \hline \end{array}$
- it can put something (a board number) in the middle $\begin{array}{|c|} \hline N \\ \hline W \quad 3 \quad E \\ \hline S \\ \hline \end{array}$

The machinery is intelligent enough to calculate the dealer and vulnerability from the board number. When only `black` or `b+w` is selected as option for colors, then the vulnerable side is written in italics rather colored. Because underlining the **South**-hand would interfere with the compass frame, we overline it. With the command `\setdefaults` one can customize the look of the compass. In particular, if you want to print the actual board number, specified by calling `\boardnr{Nr}`, you can achieve that by calling `\setdefaults{compmid=\boardtext}`. In the accompanying file with examples you'll find more examples about `\setdefaults`.

4.1.2 Hooks

We use kind of hooks to

1. change the font or the font-size. These are discussed in chapter *Sizing and Fonts* on page 19.
2. add commentary information to card diagrams. These are discussed in chapter *Conditions in Diagrams* on page 24
3. There is one other hook to enable the user to add something to the compass. This is done by calling `\setdefaults{compmid=<Text>}`.

4.2 Basic Symbols

In this section we show the predefined commands that produce terms that occur often in bridge text. On page 21 an easy way is shown to redefine them as to use a different variant of the term in question.

The next 5 macros are shorthands for the suit symbols and **NT**:

<code>\Cl</code>	<code>\Cl</code>	♣
<code>\Di</code>	<code>\Di</code>	♦
<code>\He</code>	<code>\He</code>	♥
<code>\Sp</code>	<code>\Sp</code>	♠
<code>\NT</code>	<code>ML \NT</code>	NT

The next 4 macros typeset the non-bid calls

<code>\pass</code>	<code>ML \pass</code>	Pass
<code>\allpass</code>	<code>ML \allpass</code>	All pass
<code>\double</code>	<code>ML \double</code>	Double
<code>\redouble</code>	<code>ML \redouble</code>	ReDouble

The next 6 macros typeset the directions that identify dealer (or player) and the axis that is vulnerable.

<code>\north</code>	<code>ML \north</code>	North
---------------------	------------------------	--------------

<code>\east</code>	<code>ML \east</code>	East
--------------------	-----------------------	-------------

<code>\south</code>	<code>ML \south</code>	South
---------------------	------------------------	--------------

<code>\west</code>	<code>ML \west</code>	West
--------------------	-----------------------	-------------

<code>\northsouth</code>	<code>ML \northsouth</code>	North–South
--------------------------	-----------------------------	--------------------

<code>\eastwest</code>	<code>ML \eastwest</code>	East–West
------------------------	---------------------------	------------------

The next 4 macros typeset the *unit* of the points when valuing a bridge hand. If one or more of these items do not appear in your bridge world, don't argue them, Just don't use them! They are meant to specify *High Card Points*, *High Card + Length Points*, *Length Points*, *Distribution Points* and *Total Points* respectively.

<code>\HCP</code>	<code>ML \HCP</code>	HCP
-------------------	----------------------	------------

<code>\HLP</code>	<code>ML \HLP</code>	HLP
-------------------	----------------------	------------

<code>\LP</code>	<code>ML \LP</code>	LP
------------------	---------------------	-----------

<code>\DP</code>	<code>ML \DP</code>	DP
------------------	---------------------	-----------

<code>\TP</code>	<code>ML \TP</code>	TP
------------------	---------------------	-----------

The next 5 commands produce the abbreviations for several forcing expressions.

<code>\GF</code>	<code>ML \GF</code>	GF
------------------	---------------------	-----------

<code>\SF</code>	<code>ML \SF</code>	SF
------------------	---------------------	-----------

<code>\NMF</code>	<code>ML \NMF</code>	NMF
<code>\TSF</code>	<code>ML \TSF</code>	3rd SF
<code>\FSF</code>	<code>ML \FSF</code>	4th SF

4.2.1 Header of the Bidding Table

<code>\namesNS</code>	<code>\namesNS{<N-name>}{<S-name>}</code>
	Defines the real world names for the North and the South player. If they are defined, they appear in the N - and the S -column of the bidding diagram.
<code>\namesEW</code>	<code>\namesEW{<E-name>}{<W-name>}</code>
	Same as <code>\namesNS</code> but now for the East and the West player.

4.3 The Card Diagrams

4.3.1 The hands of the players

Before a card diagram can be shown, one must specify the cards that each player holds. With the various `<player>hand`-commands one can do this. The suits they define are only shown when a `show`-command is issued after they have been defined. The `show`-commands are discussed on page 17.

<code>\northhand</code>	<code>\northhand[<v-offset>]{<Sp>}{<He>}{<Di>}{<Cl>}</code>
<code>\easthand</code>	<code>\easthand[<h-offset>]{<Sp>}{<He>}{<Di>}{<Cl>}</code>
<code>\southhand</code>	<code>\southhand[<v-offset>]{<Sp>}{<He>}{<Di>}{<Cl>}</code>
<code>\westhand</code>	<code>\westhand[<h-offset>]{<Sp>}{<He>}{<Di>}{<Cl>}</code>

The command `\northhand` defines the cards (all 4 suits) for the **N**-player. `\easthand`, `\southhand` and `\westhand` do that for the **E**-, **S**- or

W-player. These commands have 4 mandatory arguments in which the cards of the 4 suits are specified. In all suit commands where card ranks are issued, one must use **T** to denote the value **10**. On output, some kerning takes care that the output looks like **10** and not like a **1** followed by a **0**. So `\suit{AKJT8}` produces **AKJ108**. It is also possible to use a small 'x' as symbol for a spot card: `\suit{AKxxx}` produces **AKxxx**.

These commands also have an optional argument, an offset which by default is 0pt. This offset is meant to finetune the layout of the hands in the card diagrams. They change the distance between a hand and the compass. `\northhand` and `\southhand` have a *vertical* offset, whereas `\easthand` and `\westhand` have a *horizontal* one. A positive value moves away from the compass.

4.3.2 The single hand

```
\hand *!- [pos]{Sp}{He}{Di}{Cl}
```

This macro typesets the cards of one single hand, either vertically or horizontally. There are 4 mandatory arguments defining the 4 suits. With 2 optional tokens '*' resp. '!' one can typeset the hand with some special features:

- `\hand*` typesets a hand horizontally, centered
- `\hand!` typesets a hand vertically, left aligned
- `\hand*!` typesets a hand vertically, centered

For vertical hands the optional argument `pos` (default= `c`) controls the horizontal alignment. Without a token, the hand is typeset horizontally, left aligned: The call `\hand{AK2}{T85}{AQ16}{A42}` typesets the hand horizontally like:

```
♠ AK2 ♥ 1085 ♦ AQ16 ♣ A42
```

whereas the `\hand!` version produces

```
♠ AK2
♥ 1085
♦ AQ16
♣ A42
```

The third optional token, a '-' suppresses all output and saves the stuff for later use. This is used e.g. in `\expertquiz`.

4.3.3 Only one suit

`\onesuitAll` **#!** `{\langle N \rangle}{\langle S \rangle}{\langle E \rangle}{\langle W \rangle}`

Typesets the cards of 1 suit for all players. This command has 4 mandatory arguments defining the cards of the 4 players. There are 2 optional tokens. With `\onesuitAll*` the output is centered, with `onesuitAll!` the cards are placed around a NESW compass. Without the '!'-token a small box (\square) is used instead. Thus the macro call

`\onesuitAll{AQ6}{J3}{T54}{K2}` produces

A Q 6
K 2 \square 10 5 4
J 3

.

Please note the order of the players in the arguments: the first two denote the **North** and **South** hand. The last 2 denote the **East** and **West** hand. We choose it this way so you can easily cut and paste one pair from the `\onesuitAll`, or extend `\onesuitNS` to showing all hands.

`\onesuitNS` **#!** `{\langle N \rangle}{\langle S \rangle}`

`\onesuitEW` **#!** `{\langle E \rangle}{\langle W \rangle}`

`\onesuitNE` **#!** `{\langle N \rangle}{\langle E \rangle}`

`\onesuitNW` **#!** `{\langle N \rangle}{\langle W \rangle}`

These commands are similar to `\onesuitAll` but have only 2 mandatory arguments. The command `\onesuitNS{AQ3}{JT9}` typesets

and `\onesuitEW{8764}{K2}` will produce

A Q 3
\square
J 10 9

K 2 \square 8 7 6 4

.

Please note that at the latter the cards for the **East** hand appear in the first argument. Finally `\onesuitNE{AQ3}{8764}` produces

A Q 3
\square 8 7 6 4

 and

`\onesuitNW{AQ3}{K2}`

A Q 3
K 2 \square

.

`\suit` `[{\langle suit symbol \rangle}]{\langle cards \rangle}`

This command has 1 optional argument denoting a suit symbol and

1 mandatory argument, defining the cards of the suit. `\suit{AQJ7}` by default produces **AQJ7**. When the German language is active it would produce **ADB7**. Using the optional argument like in `\suit[\Di]{AQJ7}` will produce: **♦ AQJ7**.

4.3.4 Showing Card Diagrams

`\showAll` **++** [*pos*]

`\showNS` **++** [*pos*] (*N/S*)

`\showEW` **++** [*pos*] (*E/W*)

`\showNE` **++** [*pos*]

`\showNW` **++** [*pos*]

All `show`-commands have two optional tokens, a '*' which centers the output and a '+' which also displays a bidding diagram next to the card diagram. This bidding diagram must have been defined before, see page 28. They also have one optional argument that defines the aligning. Its default is `c`. `\showAll` typesets a card diagram with the NESW compass with **N** in top and the hands of the 4 players surrounding it. These hands must have been defined before by calling `\northhand` etc. Hands that are not defined are left empty. Optionally some conditions (like the dealer or who is vulnerable etc.) can be added to the diagram by using the commands described in section *Diagram Conditions*. Please note that when the **North** or **South** hand contains a long suit that extends beyond the NESW compass, this might collide with these extra texts. You can correct that with the optional offset parameter of the condition commands (see page 18).

The other commands are similar to `\showAll` but typeset only the hands of the players that are represented in the name of the command: **N-S**, **E-W**, **N-E** and **N-W**.

The commands `\showNS` and `\showEW` have an extra optional argument with which selectively only one of the two hands can be displayed. E.g. `\showNS(S)` will display only the **South**-hand.

4.3.5 Showing Card Diagrams with Bidding

4.3.6 Diagram Conditions

`\headlinetext` `\headlinetext{text}`

`\footlinetext` `\footlinetext{text}`

These commands have 1 mandatory argument: the text that defines the annotation that is to be added to a card diagram. The text can be on more than one line, just separate them with a `\par` or `\newline`⁷. `\headlinetext` places the annotation above the diagram, `\footlinetext` below it.

`\leftupper` `\leftupper[h-offset]{line1}{line2}{line3}`

`\leftlower` `\leftlower[h-offset]{line1}{line2}{line3}`

`\rightupper` `\rightupper[h-offset]{line1}{line2}{line3}`

`\rightlower` `\rightlower[h-offset]{line1}{line2}{line3}`

These commands have 1 optional argument (default 0pt) with which you can add some extra horizontal space if hand and legend collide, and 3 mandatory arguments: the lines of text that are added as conditions to the card diagram. Both `\leftupper` (`\rightupper`) place their text in the left- (right-) upper corner of the diagram. The top line will be aligned with the (inner) top of the diagram. `\leftlower` (`\rightlower`) are similar, but place their text at the lower corner of the diagram. The last line is aligned with the (inner) bottom of the diagram. For an empty line you must issue an empty argument. With a positive offset, `\leftupper` and `\leftlower` shift to the left whereas `\rightupper` and `\rightlower` shift to the right. I.e. they shift away from their neighbouring hand.

`\dealer` `\dealer[text]`

`\vulner` `\vulner[text]`

⁷Using `''\` instead produces a misleading error: **! Missing } inserted...**

Both commands have 1 optional argument. If present it sets (and prints) the internal corresponding variable to this value, otherwise it only outputs the value of this internal variable.

`\dealertext` `\dealertext[<text>]`

`\vulnertext` `\vulnertext[<text>]`

These commands have also 1 optional argument. If present e.g. `\dealertext[\North*]` this text is output in the form `Dealer: North`. If the German language is active then the call `\dealertext[\South*]` produces the text `Teiler: Süd`. Calling `\dealertext` without an argument outputs the predefined text, which can be set with `\dealer`. Example: `\dealer[Jacob]\dealertext` produces `Dealer: Jacob`.

`\boardnr` `\boardnr{<Nr>}`

The macro `\boardnr` has 1 mandatory argument. If it is a number, it is considered to be the board number. The dealer and which side is vulnerable is then calculated from it and stored in the appropriate variables. If it is not a positive integer, it is considered user-defined text and will be stored and used *as is*.

`\boardtext` `ML \boardtext *`

The macro `\boardtext` has 1 token and no arguments. `\boardtext` retrieves only the board number (stored by calling `\boardnr`). `\boardtext*` outputs the board number with some additional (multilingual) text. `\boardnr{23}\boardtext` produces `23` whereas `\boardtext*` would produce `Board: 23`. Note that `\boardnr` can have a non-integer argument. `\boardnr{Fun}\boardtext` produces `Fun` and with `\boardtext*` it would produce `Board: Fun`.

4.3.7 Sizing and Fonts

`\handskip` `\handskip{<length description>}`

This command has 1 mandatory argument: a **text** describing a length. `\handskip` enlarges the distance (default 1em) between the rightmost hand and the bidding diagram. A negative value diminishes the distance.

<code>\bidderfont</code>	<code>\bidderfont{}</code>
<code>\compassfont</code>	<code>\compassfont{}</code>
<code>\gamefont</code>	<code>\gamefont{}</code>
<code>\legendfont</code>	<code>\legendfont{}</code>
<code>\namefont</code>	<code>\namefont{}</code>
<code>\otherfont</code>	<code>\otherfont{}</code>

These commands all have 1 mandatory argument: a **description** of the font to be used. In the list below the command names are typeset in their default font.

- **bidderfont**: Used for the player-names in the bidding diagram. The default is `\mdseries\sffamily`.
- **compassfont**: Used for the directions and the "midvalue" in the compass. The default is `\mdseries\sffamily`.
- **gamefont**: Used for card diagrams, hands and suits. The default is `\bfseries\sffamily`.
- **legendfont**: Used for the conditions in card diagrams. The default is `\mdseries\rmfamily`.
- **namefont**: Used for the real world names in bidding diagrams. The default is `\mdseries\slshape`.
- **otherfont**: Used for the other bridge expressions, also outside diagrams. The default is `\bfseries\sffamily`.

If a new font is defined, all relevant dimensions of the card diagrams (including the `NESW` compass, the bidding diagram etc.) will be recalculated. Some examples for setting the `gamefont` to a new value are:

- `\gamefont{\sffamily\bfseries\HUGE}` to get *HUGE* diagrams. Refer to the documentation of package `moresize` for details.
- `\legendfont{\smaller}` to diminish the text in the card diagram conditions a little. Refer to the documentation of package `relsize` for details.

- `\gamefont{\sffamily\scafont{3}}`⁸ to typeset real big diagrams for overhead sheets

4.4 Misc

Many of the text producing macros have in common that they can produce 4 different versions of the text they represent. Normally, without any token, they produce the lowercase text. With the token `*` they produce the capitalised text. With the token `!` they produce some abbreviation of the text (if available). Finally with both tokens `*!` they produce the capitalised abbreviation of the text. What exactly is produced, is shown in the macro descriptions. In some cases it seems rather strange to have the code for an abbreviated form, i.e. `\Lead[*!]`, because it produces only the variants **lead** and **Lead**. But remember that we also support automatic translations into other languages and that in another language an abbreviation might be feasible: With the german language active `\Lead*` and `\Lead*!` produce **Ausspiel** and **Aussp.** respectively.

At the other hand it seems peculiar to let `\Ace!` produce **a** for an Ace. But we do not foresee which modern novelist might want to produce this. That's why they are defined, but 'normal' writers probably will never use it.

The short versions are primarily meant to be used within diagrams, although it is possible to get the long forms there too. Refer to page 27 for details. We show the output of such a macro `\Macro` (note the capital M!) in the form:

<code>\Macro *!</code>	<code>\Macro (\Macro*, \Macro!, \Macro*!)</code>
------------------------	--

In addition to each macro `\Macro` with its 4 variants, we also create a macro `\macro` which is defined to output the most used variant of `\Macro`:

<code>\macro</code>	most used variant of <code>\Macro</code>
---------------------	--

It is very easy to redefine `\macro`. As an example we take the macro `\ace`. Its definition is:

```
\def\ace{\Ace*!}
```

so calling `\ace` will produce **A**. If somewhere in your document you

⁸needs package `scafont`

redefine `\ace` to be

```
\def\ace{\Ace*}
```

then `\ace` will produce `\Ace` rather than `\A`.

4.4.1 Honor Cards

<code>\Ace</code>	<code>ML \Ace *!</code>	<code>ace (Ace, a, A)</code>
<code>\ace</code>	<code>ML \ace</code>	<code>A</code>
<code>\King</code>	<code>ML \King *!</code>	<code>king (King, k, K)</code>
<code>\king</code>	<code>ML \king</code>	<code>K</code>
<code>\Queen</code>	<code>ML \Queen *!</code>	<code>queen (Queen, q, Q)</code>
<code>\queen</code>	<code>ML \queen</code>	<code>Q</code>
<code>\Jack</code>	<code>ML \Jack *!</code>	<code>jack (Jack, j, J)</code>
<code>\jack</code>	<code>ML \jack</code>	<code>J</code>

These commands produce the language dependent names for the honor cards. To be used primarily when adding a lead to a card diagram.

4.4.2 Variants of Basic Symbols

In section 4.2 we already described the macros for the main variant. Here we introduce the macros that handle 4 variants with a combination of the tokens `*` and `!`.

We start with the variants for `\NT`

<code>\nt</code>	<code>ML \nt *!</code>	<code>no trump (No Trump, nt, NT)</code>
------------------	------------------------	--

Next we show the 4 macros for the non-bid calls:

<code>\Pass</code>	<code>ML \Pass *!</code>	<code>pass (Pass, p, P)</code>
<code>\Allpass</code>	<code>ML \Allpass *!</code>	<code>all pass (All pass, ap, AP)</code>
<code>\Double</code>	<code>ML \Double *!</code>	<code>double (Double, X, Dbl)</code>
<code>\Redouble</code>	<code>ML \Redouble *!</code>	<code>redouble (ReDouble, XX, ReDbl)</code>

Next come the 6 macros for the symbolic names of players and axes.

<code>\North</code>	<code>ML \North *!</code>	<code>north (North, n, N)</code>
<code>\East</code>	<code>ML \East *!</code>	<code>east (East, e, E)</code>
<code>\South</code>	<code>ML \South *!</code>	<code>south (South, s, S)</code>
<code>\West</code>	<code>ML \West *!</code>	<code>west (West, w, W)</code>
<code>\NorthSouth</code>	<code>ML \NorthSouth *!</code>	<code>north–south (North–South, n–s, N–S)</code>
<code>\EastWest</code>	<code>ML \EastWest *!</code>	<code>east–west (East–West, e–w, E–W)</code>

The 4 commands that typeset the *point-unit*

<code>\hpts</code>	<code>ML \hpts *!</code>	<code>high card points (High Card Points, hcp, HCP)</code>
<code>\tpts</code>	<code>ML \tpts *!</code>	<code>total points (Total Points, tp, TP)</code>
<code>\lpts</code>	<code>ML \lpts *!</code>	<code>length points (Length Points, lp, LP)</code>
<code>\dpts</code>	<code>ML \dpts *!</code>	<code>distribution points (Distribution Points, dp, DP)</code>

The forcing terms

<code>\gforce</code>	<code>ML \gforce *!</code>	game forcing (Game Forcing, gf, GF)
<code>\sforce</code>	<code>ML \sforce *!</code>	semi forcing (Semi Forcing, sf, SF)
<code>\nmforce</code>	<code>ML \nmforce *!</code>	new minor forcing (New Minor Forcing, nmf, NMF)
<code>\tsforce</code>	<code>ML \tsforce *!</code>	third suit forcing (Third Suit Forcing, 3rd sf, 3rd SF)
<code>\fsforce</code>	<code>ML \fsforce *!</code>	fourth suit forcing (Fourth Suit Forcing, 4th sf, 4th SF)

4.4.3 Conditions in Diagrams

These commands produce the language dependent expressions for *All*, *None*, *by*, *Board* etc. To be used primarily in card diagrams.

<code>\All</code>	<code>ML \All *!</code>	all (All, all, All)
<code>\all</code>	<code>ML \all</code>	All
<code>\None</code>	<code>ML \None *!</code>	none (None, none, None)
<code>\none</code>	<code>ML \none</code>	None
<code>\by</code>	<code>ML \by</code>	by
<code>\Board</code>	<code>ML \Board *!</code>	board (Board, brd, Brd)
<code>\board</code>	<code>ML \board</code>	Board
<code>\Contract</code>	<code>ML \Contract *!</code>	contract (Contract, contr, Contr)

<code>\contract</code>	<code>ML \contract</code>	Contract
<code>\Declarer</code>	<code>ML \Declarer *!</code>	declarer (Declarer, decl, Decl)
<code>\declarer</code>	<code>ML \declarer</code>	Declarer
<code>\Deal</code>	<code>ML \Deal *!</code>	deal (Deal, deal, Deal)
<code>\deal</code>	<code>ML \deal</code>	Deal
<code>\Lead</code>	<code>ML \Lead *!</code>	lead (Lead, lead, Lead)
<code>\lead</code>	<code>ML \lead</code>	Lead

4.4.4 Annotations in Bidding Diagrams

<code>\alert</code>	<code>\alert</code>	*
<code>\announce</code>	<code>\announce</code>	A

These macros have no argument. With `\alert` one can mark a call that must be alerted with an asterisk (*) e.g. a weak **2NT** opening with `2\NT\alert`. It produces $\boxed{2\text{NT}^*}$. With `\announce` one can mark a bid with an 'A' (**A**) where an announcement is obligatory, e.g. `2\He\announce` produces $\boxed{2\heartsuit^A}$.

<code>\markit</code>	<code>\markit</code>
<code>\explainit</code>	<code>\explainit{\langle explanation \rangle}</code>

These commands are to be used to mark a call in the bidding diagram and explain it with a kind of footnote-like mechanism, directly below the bidding diagram. Both `\markit` and `\explainit` step a counter for associating the explanation with the mark. `\markit` has no arguments;

`\explainit` has one mandatory argument: the text to be displayed as explanation. `\explainit` should be called in the description part of bidding diagrams (or `expertquiz`). The text of the explanation is then typeset under the bidding diagram and has the same width. You can use newline (`\`, or `\newline`) in your text to force an new line in the explanation⁹.

4.4.5 Specials

`\expertquiz`

```
\expertquiz *! [<comment>]{<award>}
```

This command displays a hand, a bidding sequence and some additional stuff. It is designed after the *Expert Quiz* column in the *Bridge Magazin* of the DBV¹⁰, the German Bridge Union.

It has 2 optional tokens: a `'*` for centering and a `'!` which 1) forces the bidding diagram to appear on a new line, and 2) shifts the hand a bit to the right. Next there is one optional argument with which some commentary information can be added. And finally there is 1 mandatory argument that describes the awards for certain solutions. Both the hand and the bidding must be defined before calling `\expertquiz`. One can do that by calling e.g. `\hand-` which suppresses the output of the hand.

4.4.6 Re-Initialization

`\newgame`

```
\newgame
```

This command resets most bridge diagram data and can be used to start a new series of bridge diagrams. It is however not necessary to use `\newgame` if one enters new cards for the **North** etc. hands. The list below shows which items are reset by calling `\newgame`

- resets `\boardnr{0}`.
- resets `headlinetext` and `footlinetext`.
- resets `LeftUpperText`, `LeftLowerText`, `RightUpperText` and `RightLowerText`.
- resets `northhand`, `easthand`, `southhand` and `westhand`.

⁹Using `\par` produces the error: **Runaway argument?...**

¹⁰Deutsche Bridge Verband

- resets `namesNS` and `namesEW`.
- resets the checks for Spades, Hearts, Diamonds and Clubs.

```
\setdefaults *{\key1=val1},\key2=val2},...
```

key	font	key	value
bidder	bidderfont	compline	$\langle factor \rangle$
compass	compassfont	compshow	<u>off</u> <u>on</u>
game	gamefont	compsize	$\langle factor \rangle$
legend	legendfont	compturn	<u>off</u> <u>on</u>
name	namefont		
other	otherfont		

key	value	key	value
colors	0 1 <u>2</u> 3 4A 4B	bidders	<u>off</u> <u>on</u>
warn	<u>off</u> <u>on</u>	bidfirst	N E S <u>W</u>
err	<u>off</u> <u>on</u>	bidline	<u>off</u> <u>on</u>
		bidlong	<u>off</u> <u>on</u>

The macro `\setdefaults` uses a $\langle key \rangle = \langle val \rangle$ mechanism. The tables above show which keys are available. The underlined key values are defaults. The available keys with respect to fonts are: `bidder`, `compass`, `game`, `legend`, `name` and `other`. In upper the table to the left the association between key and font is shown. A call e.g. `\setdefaults{name=\bfseries\scriptsize}` will set the default for the namefont to the value specified. The starred form `\setdefaults*` will also call `\resetfonts`, which effectuates any change in a new default font immediately.

In the lower table to the left you'll find the three keys that are also possible as package options. They have been described already in section 3.1.3.

The keys with respect to the compass are: `compline`, `compshow`, `compsize` and `compturn`. They are shown in the upper table to the right. With the first key one can set the linethickness of the compass frame, the default is 0.1em. The second key controls whether the compass is shown or not. The third key controls the size of the compass, which per default is 2.5em. With the fourth key one can rotate the letters for the **E-W** direction over 90°. The multiplication- $\langle factor \rangle$, which defaults to the value 1, can have any non-negative real value.

The keys with respect to the bidding diagram are: `bidders`, `bidfirst`, `bidline` and `bidlong`. They are shown in the lower table

to the right. With the first key one can suppress the bidders in the bidding header. With the second key one can set which bidder appears in the first column of the diagram. The default is `w`. The third key controls whether an `\hline` is printed below the header. The fourth key switches between the long or short form of the non-bid calls, like `Pass` or `p`.

For the key-value `on` we have the synonyms `true` and `1`, for the key-value `off` we have the synonyms `false` and `0`.

`\resetfonts`

```
\resetfonts
```

When calling this macro, all fonts are set back to their default value. This is the value that was explicitly set by a previous call of `\setdefaults`, or to the intrinsic default value if `\setdefaults` has not been called before.

4.5 Environments

In the first place we must warn the user for a peculiarity of the package `collcell`, which is used to do some special processing in these 3 environments: *The last row must either end with a newline (`\`) or with an empty cell.*

The advantage of the `collcell` processing is that within the `bidding` and `play` diagrams we can use shorthands for the suit symbols: rather than writing `3\Sp` in a `bidding` diagram we can also write `3S` to obtain `3♠`. In the `play` diagram we could write `HA` instead of `\He\`, `\Ace` and get `♥A` as output. In `bidding` diagrams some non-bid calls may appear in short or in long form, controlled by calling `\setdefaults{bidlong=on}`, which switches to the long form, or `\setdefaults{bidlong=off}` which switches to the short form. These non-bid calls are coded as follows: A small `p` yields short form `p` or long form `Pass`. A capital `P` yields short form `AP` or long form `All pass`. A capital `x` yields short form `X` and long form `Double`. A capital `R` yields short form `XX` and long form `ReDouble`.¹¹

`bidding`

```
ML \begin{bidding} *!- [pos](description)... \end{bidding}
```

This environment has 3 tokens. The `'*` centers the `bidding` diagram. The `'!` outputs the short form: e.g. `N` rather than the long form `North` in the table heading. The `'-` completely suppresses the output. The

¹¹due to the fact that `x` denotes a spotcard, we had to implement it this way

data is stored in a savebox and can be used in other macros, e.g. in all `\showXX` macros. Next come 2 optional parameters. The first one controls the alignment (default `c`) and the second one adds the list of annotations below the `bidding` diagram. With `\setdefaults` one can fine tune the look of the `bidding` diagram. Refer to page 27 for details. For example

```
\begin{bidding}(\explainit{WeakTwo}\explainit{preemptive})
  2H\markit & p & 3H\markit & p \\
\end{bidding}
\quad
\setdefaults{bidlong=off}
\begin{bidding}!
  2H & p & 3H & p \\
\end{bidding} \\
```

produces

West	North	East	South		W	N	E	S
2♥ ^a	Pass	3♥ ^b	Pass		2♥	p	3♥	p
^a WeakTwo								
^b preemptive								

In addition to the shorthands we mentioned before, one can also use macros in the bidding and play diagrams. As long as these macros appear as 1 single token in the diagram, no special care has to be taken. But if a macro appears as multiple tokens, e.g. like the call `\Pass*!`, which consists of the 3 tokens `\Pass`, `*` and `!`, then it **must** be enclosed in braces `{...}` to make it act as 1 token. Without the braces `\Pass*!` will produce `pass*!`, with them `{\Pass*!}` will produce `P`. Note that although `\pass` expands to `\Pass*`, it will produce the correct `Pass`.

`biddingpair` `ML \begin{biddingpair} *!- [pos] (<descr>)...\end{biddingpair}`

This environment is essentially the same as the `bidding` environment, but here the `bidding` diagram has only two columns rather than four.

`play` `ML \begin{play} * {<Lead>} [<Trump>]...\end{play}`

This environment has 1 token, a `'*`, which controls the centering of the table; 1 mandatory argument, denoting the opening lead and 1 optional argument which specifies the trump suit, with default: **NT**. It typesets a diagram with the sequence of playing tricks, by producing a table with 8 columns: the running number; the player who had had

the lead; the 4 cards played in this trick and finally the 2 columns that show how many tricks **N-S** and **E-W** have won so far. The user has to specify only the columns 3-6, i.e. the cards that were played. Columns 1-2 and 7-8 are constructed by **ONEDOWN**. Just as in `bidding` diagrams, also here one can denote the suit with S, H, D, or C. The winning card is automatically detected, taking into account if a suit contract or **NT** is played. The winning tricks counters are then updated automatically. Concerning consistency it is just as with hands and deals: If a card occurs more than one time, an error is raised and for missing cards a warning is issued.

```
\begin{play}{W} \\
  D3 & 2 & Q & K \\
  HA & 5 & 4 & J \\
  C3 & 5 & K & A \\
\end{play}
```

N ^o	Lead	2nd	3rd	4th	N S	E W
1	W: ♠3	2	Q	K	1	0
2	S: ♥A	5	4	J	2	0
3	S: ♣3	5	K	A	2	1

```
\begin{play}*{W} \\
  D3 & 2 & Q & K \\
  HA & 5 & 4 & J \\
  C3 & 5 & K & A \\
\end{play}
```

produces the same diagram, but this time horizontally centered.

N ^o	Lead	2nd	3rd	4th	N S	E W
1	W: ♠3	2	Q	K	1	0
2	S: ♥A	5	4	J	2	0
3	S: ♣3	5	K	A	2	1

5 Final Remarks

5.1 Known Bugs

- Some dictionaries have questionable translations.
- The boxes in `\northhand` and `\southhand` lead to problems with shifting the **West** hand. See file `Legends` in the example bundle.

5.2 ToDo

- Add the High Points to card diagrams (like Turnier in KA)
- Check if `\def\xspace{}` is also needed in bidding...?
- Make a template for showing 16 hands on 1 DIN A4 (3x5+1 or 4x4 landscape) e.g as handout for the hands on slides.
- Read source files in PBN format. Example: <http://new.bridgekosice.sk/bridzove-diagramy-vykrelene-pomocou-tex/>

5.3 Acknowledgements

This package is based on (ideas from) the style files:

- `bridge-i v1.0` (1995/04/16) by René Steiner and Thomas Hof.
- `kibitzer v1.0` (1995/04/16) by René Steiner and Thomas Hof.
- `bridge v0.1` (2012/03/18) by Antony Lee.
- `grbbridge v2.2` (2013/12/24) by Gordon Bower.

The style file `bridge-i` is based on the style file `bridge v1.7c` (1994/12/20) by J.L. Braams, which on itself was based on an article by Kees van der Laan in TUGboat (Vol 11 (1990), No 2: p265ff.

Last but not least I want to thank all those wonderful people down there in the Internet who spent their time in answering silly questions and solving difficult problems. If I had imagined the difficulties I would encounter, then I would not have started this work. And without the help of all these, to me unknown, people, this package would not exist.

6 Implementation

6.1 Preliminaries

6.1.1 Packages we depend upon

```
1 % Warn if a too old expl3 package is used.
2 \RequirePackage{expl3}[2019/09/21]%   needed for LaTeX3 packages (xparse)
3 \RequirePackage{%
4   xcolor,%           colorizing symbols \Sp etc.
5   textcomp,%        for the numbersign in environment play.
6   moresize,%        add \HUGE and \ssmall to font-sizes
7   relsize,%         relative font-sizes. (e.g. \smaller)
8   makecmds,%        needed for provideenvironment
9 % expl3,%           needed for LaTeX3 packages (xparse)
```

```

10  xparse,%      optional params and starred commands
11  xspace,%     handling of spacing behind a command
12  calc,%       makes calculations and lengths easier
13  ifthen,%     easy booleans, tests and loops
14  adjustbox,%  stacked boxes in L-/R-Lower captions
15  translator,% auto-translate terms (e.g. East->Ost)
16  array,%      actions for tabular column cells
17  collcell,%   macro calls for tabular column cells
18  pgfopts,%    for keyval opts, loads also pgfkeys
19  environ,%    for handling bidding environments
20  xstring,%    for easy string processing.
21  tracklang,%  for iterating over loaded languages
22  pict2e,%     for drawing the compass
23 }

```

Add exceptions for xspace

```

24 \xspaceaddexceptions{%
25   = \markit \, \suit \translate
26   2 3 4 5 6 7 8 9 T J Q K A
27 }

```

6.1.2 Options

We use the pgf `<key>=<val>` system for our options: `colors`, `warn` and `err`.

```

28 \pgfkeys{/ODw/.is family}
29 \def\ODw@set#1{\pgfkeys{/ODw,#1}}
30 \ODw@set{colors/.is choice,}
31 \ODw@set{warn/.is choice,}
32 \ODw@set{err/.is choice,}

```

The details for option `colors` are on page 39ff and those for option `warn` and `err` on page 100.

6.1.3 Misc

```

\, \,
\thinspace \thinspace

```

We redefine `\thinspace` (originally $\frac{1}{6}$ em) to a smaller amount. That makes denominations like `3\Sp` (3♠) look better. The code is from: <https://tex.stackexchange.com/questions/181003/multiply-fine-tuning-with-a-thinspace>

This code however doesn't work when coded within an own package, unless we use `\AtBeginDocument`.

```

33 \AtBeginDocument{%
34   \renewcommand{\,}[1][1]{%
35     \ifmmode\mskip#1\thinmuskip%
36     \else\thinspace[#1]\fi%
37   }%
38   \renewcommand{\thinspace}[1][1]{%
39     \kern#1\dimexpr0.16667em\relax%
40   }%
41 }% AtBeginDocument

```

`\ODw@gsetlength`

```
\ODw@gsetlength{\len-name}{\len-value}
```

We need to store the length of certain objects that are within a group (the group is needed to keep the font-size changes local). Therefore we define macro `\ODw@gsetlength` that works globally. The code is based upon a solution on LaTeX StackExchange: (<https://tex.stackexchange.com/questions/406015/defining-macro-gsetlength-as-global-setlength-reliable>)

```

42 \gdef\ODw@gsetlength#1#2{%
43   \begingroup
44   \setlength\skip@{#2}% local assign to scratch reg.
45   \global#1=\skip@%    global assignment to #1;
46   \endgroup%          restore \skip@ by endgroup.
47 }% \ODw@gsetlength

```

`\ODw@append`

```
\ODw@append{\tokens}
```

In the environment `play` we need to calculate the winning tricks for **N-S** and **E-W**¹². We store this information as a string in `\ODw@Scratch` and use `\ODw@append` to accumulate them.

```

48 \gdef\ODw@append#1{%
49   \bgroup%
50   \edef\tmp{\the\ODw@Scratch #1}%
51   \global\ODw@Scratch=\expandafter{\tmp}%
52   \egroup%
53 }% \ODw@append

```

¹²For details see page 91

6.1.4 Variables

In this package we do use font relative sizing. That means that widths and skips are defined in terms of `em`, `ex` and `baselineskip`. On the other hand there are e.g. the real world names of the bidders, that must be recorded. Some of these the user should be able to control. Rather than forcing the user to do that directly with `\def` or `\renewcommand` we store all this information in internal variables, by defining a constant command. These variables can be set by calling a user command, that is associated with the variable. E.g. the variable `\ODw@Skipwidth` gets set by the command `\handskip`. The variables we use are:

<code>\ODw@BidderFont</code>	Controlled by <code>\bidderfont</code> : the font used for the players (<code>bidding</code>).
<code>\ODw@CompassFont</code>	Controlled by <code>\compassfont</code> : the font used for the compass.
<code>\ODw@GameFont</code>	Controlled by <code>\gamefont</code> : the font used for all card diagrams
<code>\ODw@LegendFont</code>	Controlled by <code>\legendfont</code> : the font used for the annotations.
<code>\ODw@NameFont</code>	Controlled by <code>\namefont</code> : the font used for the real world names.
<code>\ODw@OtherFont</code>	Controlled by <code>\otherfont</code> : the font used for other bridge expressions, also outside diagrams.
<code>\ODw@BidderDefault</code>	These contain the default values for the fonts
<code>\ODw@CompassDefault</code>	
<code>\ODw@GameDefault</code>	
<code>\ODw@LegendDefault</code>	
<code>\ODw@NameDefault</code>	
<code>\ODw@OtherDefault</code>	
<code>\ODw@North@Name</code>	Controlled by <code>\namesNS</code> : hold the real world names of the North and the South player.
<code>\ODw@South@Name</code>	
<code>\ODw@East@Name</code>	Controlled by <code>\namesEW</code> : hold the real world names of the East and the West player. These names are typeset using the font specified with <code>\namefont</code> .
<code>\ODw@West@Name</code>	
<code>ODw@BoardText</code>	Controlled by <code>\boardtext</code> : holds the board number and extra text.
<code>ODw@HeaderText</code>	Controlled by <code>\headlinetext</code> : holds header information for card diagrams.
<code>ODw@FooterText</code>	Controlled by <code>\footlinetext</code> : holds footer information for card diagrams.
<code>ODw@Last</code>	Used in command <code>\ODw@Tricks</code> : to store the player that had the lead ¹³ .
<code>\ODw@Skipwidth</code>	Controlled by <code>\handskip</code> : holds the distance between hand and bidding diagram.

¹³see page 89

```

54 \def\ODw@CompSize{1}% factor to enlarge the compass
55 \def\ODw@CompLine{}% thickness of compass frame
56 \def\ODw@Skipwidth{1em}%

```

6.1.5 Booleans, Saveboxes, Lengths, Counters and Registers

6.1.5.1 Booleans

`\ifODw@description` `\ifODw@description` is used in the bidding environments and the command `\expertquiz` to test if there is an annotation that should be written.

`\ifODw@short` `\ifODw@short` is used in the bidding environments and the command `\expertquiz` to denote if a short form of the diagram header is to be used.

`\ifODw@monochrome` `\ifODw@monochrome` flags the case that the user specified `colors=0` or `colors=1`, i.e. just black and white. In this situation we will not print the vulnerable side in red, but use italics instead.

`\ifODw@CardSkip` `\ifODw@CardSkip` determines whether we need some extra space between card ranks (i.e. in suit descriptions) or not (i.e. in bidding or play diagrams).

`\ifODw@Bidders` `\ifODw@Bidders` suppresses the bidders in the bidding header.

`\ifODw@BidLine` `\ifODw@BidLine` draw a `\hline` below the bidding header.

`\ifODw@LongCalls` `\ifODw@LongCalls` determines whether to use the short (like **N**) or the long form (like **North**) for calls in the bidding diagram.

`\ifODw@CompShow` With `\ifODw@CompShow` one can suppress drawing a compass within card diagrams completely.

`\ifODw@CompTurn` With `\ifODw@CompTurn` one rotates letters E and W in the compass 90°.

```

57 \newif\ifODw@description% must typeset an annotation
58 \newif\ifODw@short%      short form in bidding header
59 \newif\ifODw@monochrome% no colors wanted
60 \newif\ifODw@CardSkip%   skip between ranks needed

```

The next booleans are directly controlled by `\setdefaults`.

```

61 \newif\ifODw@Bidders%    suppress bidders in bidding header
62 \newif\ifODw@BidLine%   draw \hline below bidding header
63 \newif\ifODw@LongCalls% switch between long/short calls
64 \newif\ifODw@CompShow%  show compass or not
65 \newif\ifODw@CompTurn%  turn E-W letters 90°

```

`\ODw@EmptyHeader` Since there seems to be a problem in using `\ifthenelse`¹⁴ in particular places, these booleans are set by calling `ODw@TestIfEmpty` (which uses

`\ODw@EmptyFooter`

¹⁴`\ifthenelse` bites `multicolumn!`

an ordinary `\ifthenelse`) *outside* the dangerous places, and then use e.g. `\ifODw@EmptyHeader` as a test whether the header is empty or not.

```
66 \newboolean{ODw@EmptyHeader}% = 'header is empty'
67 \newboolean{ODw@EmptyFooter}% = 'footer is empty'
```

6.1.5.2 Saveboxes

`\ODw@Diagram@Box` `\ODw@Diagram@Box` is to store the actual card diagram (the compass with the hands) in order to calculate its width.

`\ODw@Bid@Box` `\ODw@Bid@Box` stores a bidding diagram.

`\ODw@Hand@Box` `\ODw@Hand@Box` stores a hand with the 4 suits.

```
68 \newsavebox\ODw@Diagram@Box
69 \newsavebox\ODw@Hand@Box
70 \newsavebox\ODw@Bid@Box
```

6.1.5.3 Lengths

`\ODw@Bid@Width` Is used to store the actual width of the bidding diagram.

`\ODw@Card@Skip` Defines space between adjacent cards in suits.

`\ODw@Diagram@Width` Defines the width of compass plus (**E-W**) hands.

`\ODw@Skip@Width` Defines the distance between the card diagram and the bidding diagram.

`\ODw@Tmp@Len`

`\ODw@Tmp@Width` Auxiliary lengths, used in several calculations.

```
71 \newlength\ODw@Compasssize% the size of the compass.
72 \newlength\ODw@Diagram@Width
73 \newlength\ODw@Card@Skip
74 \setlength\ODw@Card@Skip{.15em}% space between cards
75 \newlength\ODw@Bid@Width
76 \newlength\ODw@Skip@Width
77 \setlength\ODw@Skip@Width{\ODw@Skipwidth}
78 \newlength\ODw@Tmp@Len% temp var for computations
79 \newlength\ODw@Tmp@Width% temp var for computations
```

6.1.5.4 Counters

`ODw@Nr` Counts lines (in play diagrams) and explanations (in bidding diagrams).

```
80 \newcounter{ODw@Nr}
```

`ODw@Cnt` Auxiliary counter, used in several calculations.

```
81 \newcounter{ODw@Cnt}
```

`ODw@PlayerNr` Set to the player that won the trick in environment `play`.

82 `\newcounter{ODw@PlayerNr}`

`ODw@NSCnt` Holds the number of N-S tricks in environment `play`.

83 `\newcounter{ODw@NSCnt}`

`ODw@EWCnt` Holds the number of E-W tricks in environment `play`.

84 `\newcounter{ODw@EWCnt}`

6.1.5.5 Registers

`ODw@Scratch` Tempory store for winning tricks in environment `play`.

85 `\newtoks{\ODw@Scratch}`

6.1.6 Fonts

6.1.6.1 Text Fonts

Here we merely define the commands to set the default fonts. At the end of this `.sty` file they are set to their value. Refer to section [6.10](#) for details.

`\bidderfont` The font used to indicate the symbolic player (N, E, S, W) in bidding diagrams. The default is `\mdseries\sffamily`.

`\ODw@BidderFont`

86 `\newcommand\bidderfont[1]{\gdef\ODw@BidderFont{#1}}`

`\compassfont` The font used to indicate the directions (N, E, S, W) in the compass. The default is `\mdseries\sffamily`.

`\ODw@CompassFont`

87 `\newcommand\compassfont[1]{\gdef\ODw@CompassFont{#1}}`

`\namefont` The font used for the real world names of the players in bidding diagrams. The default id `\mdseries\slshape`.

`\ODw@NameFont`

88 `\newcommand\namefont[1]{\gdef\ODw@NameFont{#1}}`

`\legendfont` The font used for the conditions in card diagrams. The default is `\mdseries\rmfamily`.

`\ODw@LegendFont`

89 `\newcommand\legendfont[1]{\gdef\ODw@LegendFont{#1}}`

`\otherfont` The font used for the other bridge expressions like `\north`, `\pass` or `\double`. The default is `\bfseries\sffamily`.

`\ODw@OtherFont`

90 `\newcommand\otherfont[1]{\gdef\ODw@OtherFont{#1}}`

`\gamefont` The font for the hands and calls. It sets the general font-size/widths for the game. The default is `\bfseries\sffamily`.

`\ODw@GameFont` We use widths that are dynamically adjusted at font changes and store the 'value' as text in `\ODw@GameFont`.

`\ODw@GameSize` `\ODw@GameSize` recalculates these sizes and is called in all show- and bid-diagrams.

```

91 \newcommand\gamefont[1]{%
92   \gdef\ODw@GameFont{#1}%
93   \gdef\ODw@GameSize{% recalculate dimens for the new font
94     \ODw@GameFont%
95     \setlength\ODw@Skip@Width{\ODw@Skipwidth}%
96   }%
97 }% gamefont

```

6.1.6.2 Symbol Font

We need special symbols to get solid colored ♥ and ♦, rather than ♡ and ◇. We use those from `stix`. As the shape of the 'normal' black suits differ from the red ones we also take the black suits from the font `stix`. First we define the symbols and font. As we do not want to load the complete package, we only use the relevant piece of code found in `txfont.sty`:

```

98 \fontencoding{T1}\fontfamily{stix}
99 \fontseries{m}\fontshape{n}\selectfont
100 %
101 % Code stolen from txfonts.sty.
102 % It works smoothly: thank you guys!
103 % Because of an interference with package newtxmath I had to rename
104 % symbols into ODw@symbols and symbolsC into ODw@symbolsC
105 %
106 \DeclareSymbolFont{ODw@symbols}{OMS}{txsy}{m}{n}
107 \SetSymbolFont{ODw@symbols}{bold}{OMS}{txsy}{bx}{n}
108 \DeclareFontSubstitution{OMS}{txsy}{m}{n}
109 \DeclareSymbolFont{ODw@symbolsC}{U}{txsyc}{m}{n}
110 \SetSymbolFont{ODw@symbolsC}{bold}{U}{txsyc}{bx}{n}
111 \DeclareFontSubstitution{U}{txsyc}{m}{n}

```

`\ODw@spadesuit`

```
112 \DeclareMathSymbol{\ODw@spadesuit}{\mathord}{ODw@symbols}{127}
```

`\ODw@varheart`

```
113 \DeclareMathSymbol{\ODw@varheart}{\mathord}{ODw@symbolsC}{114}
```

`\ODw@vardiamond`

```
114 \DeclareMathSymbol{\ODw@vardiamond}{\mathord}{ODw@symbolsC}{113}
```

```
\ODw@clubsuit
```

```
115 \DeclareMathSymbol{\ODw@clubsuit}{\mathord}{\ODw@symbols}{124}
```

6.2 Bridge Basic Terms

6.2.1 Suit Symbols

First we supply shorthands for the ‘five’ suits (\clubsuit , \diamondsuit , \heartsuit , \spadesuit and **NT**) that are used in the game of bridge. We define the international version with the English shortcuts. We use the `xcolor` package to colorize the suit symbols. The color can be set as an `<key>=<val>` option when loading the package. The option `colors=0` means mono-color (black only), synonyms of key 0 are `mono` and `black`. `colors=1` means black and white, a synonym is `b+w`. `colors=2` means bi-color (black and red), with synonym `b+r`. `colors=3` means grey, with synonyms `gray` and `grey`. This ‘color’ is meant for special effects, e.g. for making unimportant parts less visible `colors=4A` gives qua-color (green, orange, red and blue); synonyms are `fourA` and `4a`. Finally `colors=4B` defines the second qua-color (black, orange, red and green) with synonyms `fourB` and `4b`.

We precede all the suit symbols with a ‘very-thin-space’ (`\, [0.3]`) which is 0.3 the size of a normal `\thinspace`.

In order to test which suit (`\Cl,...`) was encountered in `\ODw@translate`¹⁵ we *must* define the suits with a `renewrobustcommand`. So we must `\def` them first.

```
116 % \def them first and then use renewrobustcmd!
117 \def\Cl{\def\Di{\def\He{\def\Sp{}}%
118 \ODw@set{%
119   colors/0/.code={%
120     \ODw@monochrometrue%
121     \renewrobustcmd\Cl{\textcolor{black}%
122       {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
123     \renewrobustcmd\Di{\textcolor{black}%
124       {\, [0.3]\ensuremath{\ODw@vardiamond}}\xspace}%
125     \renewrobustcmd\He{\textcolor{black}%
126       {\, [0.3]\ensuremath{\ODw@varheart}}\xspace}%
127     \renewrobustcmd\Sp{\textcolor{black}%
128       {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
129   }%
130 }
131 \ODw@set{colors/mono/.code={\pgfkeys{/ODw/colors=0}}}
```

¹⁵see page 45

```

132 \ODw@set{colors/black/.code={\pgfkeys{/ODw/colors=0}}}
133 %
134 \ODw@set{%
135 colors/1/.code={%
136   \ODw@monochrometrue%
137   \renewrobustcmd\Ci{\textcolor{black}%
138     {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
139   \renewrobustcmd\Di{\, [0.3]\ensuremath{\diamondsuit}\xspace}%
140   \renewrobustcmd\He{\, [0.3]\ensuremath{\heartsuit}\xspace}%
141   \renewrobustcmd\Sp{\textcolor{black}%
142     {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
143 }%
144 }
145 \ODw@set{colors/b+w/.code={\pgfkeys{/ODw/colors=1}}}
146 %
147 \ODw@set{%
148 colors/2/.code={%
149   \ODw@monochromefalse%
150   \renewrobustcmd\Ci{\textcolor{black}%
151     {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
152   \renewrobustcmd\Di{\textcolor{red}%
153     {\, [0.3]\ensuremath{\ODw@vardiamond}}\xspace}%
154   \renewrobustcmd\He{\textcolor{red}%
155     {\, [0.3]\ensuremath{\ODw@varheart}}\xspace}%
156   \renewrobustcmd\Sp{\textcolor{black}%
157     {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
158 }%
159 }
160 \ODw@set{colors/b+r/.code={\pgfkeys{/ODw/colors=2}}}
161 %
162 \ODw@set{%
163 colors/3/.code={%
164   \ODw@monochromefalse%
165   \renewrobustcmd\Ci{\textcolor{gray}%
166     {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
167   \renewrobustcmd\Di{\textcolor{gray}%
168     {\, [0.3]\ensuremath{\ODw@vardiamond}}\xspace}%
169   \renewrobustcmd\He{\textcolor{gray}%
170     {\, [0.3]\ensuremath{\ODw@varheart}}\xspace}%
171   \renewrobustcmd\Sp{\textcolor{gray}%
172     {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
173 }%
174 }

```



```

175 \ODw@set{colors/grey/.code={\pgfkeys{/ODw/colors=3}}}
176 \ODw@set{colors/gray/.code={\pgfkeys{/ODw/colors=3}}}
177 %
178 \ODw@set{%
179 colors/4A/.code={%
180 \ODw@monochromefalse%
181 \renewrobustcmd\C1{\textcolor{green}%
182   {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
183 \renewrobustcmd\Di{\textcolor{orange}%
184   {\, [0.3]\ensuremath{\ODw@vardiamond}}\xspace}%
185 \renewrobustcmd\He{\textcolor{red}%
186   {\, [0.3]\ensuremath{\ODw@varheart}}\xspace}%
187 \renewrobustcmd\Sp{\textcolor{blue}%
188   {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
189 }%
190 }
191 \ODw@set{colors/fourA/.code={\pgfkeys{/ODw/colors=4A}}}
192 \ODw@set{colors/4a/.code={\pgfkeys{/ODw/colors=4A}}}
193 %
194 \ODw@set{%
195 colors/4B/.code={%
196 \ODw@monochromefalse%
197 \renewrobustcmd\C1{\textcolor{black}%
198   {\, [0.3]\ensuremath{\ODw@clubsuit}}\xspace}%
199 \renewrobustcmd\Di{\textcolor{orange}%
200   {\, [0.3]\ensuremath{\ODw@vardiamond}}\xspace}%
201 \renewrobustcmd\He{\textcolor{red}%
202   {\, [0.3]\ensuremath{\ODw@varheart}}\xspace}%
203 \renewrobustcmd\Sp{\textcolor{green}%
204   {\, [0.3]\ensuremath{\ODw@spadesuit}}\xspace}%
205 }%
206 }
207 \ODw@set{colors/fourB/.code={\pgfkeys{/ODw/colors=4B}}}
208 \ODw@set{colors/4b/.code={\pgfkeys{/ODw/colors=4B}}}

```

\nt Because some languages use a different symbol for **NT (No Trump)**
\NT we must look it up in the dictionary to find e.g. **SA (Sans Atout)** for German.

```

209 \NewDocumentCommand\nt{s t!}{%
210   \bgroup%
211     \ODw@OtherFont%
212     \IfBooleanTF#1{%
213       \IfBooleanTF#2

```

```

214     {\,[0.3]\translate{NT-(ODw)}}}%
215     {\translate{No Trump-(ODw)}}}%
216   }{%
217   \IfBooleanTF#2
218     {\,[0.3]\translate{nt-(ODw)}}}%
219     {\translate{no trump-(ODw)}}}%
220   }%
221   \egroup%
222   \xspace%
223 }% nt

```

Define a practical shorthand to produce **NT** without the need to add a token.

```
224 \def\NT{\nt*!}
```

`\ODw@SetRank` `\ODw@SetRank{\langle card rank \rangle}`

`\ODw@SetRank` stores the rank of the card played in `\ODw@Rank`. This is essentially the intrinsic rank of the card (*2* for a 2, *14* for an Ace), but there are special cases:

- Spot cards (denoted with x) always get rank 0
- Discards always get rank 0
- Trump cards get 15 (*15* to ensure that a spot trump card defeats an Ace) added to the intrinsic rank, to make sure that:
 - A trump card will defeat all other cards
 - The highest trump card will win the trick

We first define three variables, one to store the suit of the actual card, the second one to store which suit was led and the last variable to store which suit is the trump suit, all initialized with the '**NT**-suit'.

```

225 \gdef\ODw@SuitPlayed{N}
226 \gdef\ODw@SuitLead{N}
227 \gdef\ODw@TrumpSuit{N}
228 \newcounter{ODw@Rank}

229 \def\ODw@SetRank#1{%
230   \ifthenelse{\equal{\ODw@SuitPlayed}{\ODw@SuitLead}}%
231   % if a suit is followed, store the intrinsic rank
232   {\setcounter{ODw@Rank}{#1}}%
233   {% else, if a suit is not followed then ...
234   % (at NoTrump, the trumpsuit is coded 'N' and will
235   % never match a real suit (coded C, D, H and S))
236   % thus avoiding that trump cards are detected
237   \ifthenelse{\equal{\ODw@SuitPlayed}{\ODw@TrumpSuit}}%

```

```

238 % if it is a trump card, increase the rank
239     {\setcounter{ODw@Rank}{#1}%
240      \addtocounter{ODw@Rank}{15}}%
241 % if it is a discard, set the rank to 0 to make
242 % sure it will never win
243     {\setcounter{ODw@Rank}{0}}%
244 }% ifthen
245 % If the card was of another suit,
246 % then ODw@SuitPlayed was changed.
247 % If we encounter 'unsuited' cards,
248 % then we must reestablish the
249 % original ODw@SuitPlayed.
250 \global\edef\ODw@SuitPlayed{\ODw@SuitLead}% org. suit
251 }% ODw@SetRank

```

`\ODw@Xfer`

```
\ODw@Xfer{<tokens>}
```

This macro gets called by `ODw@Tfer` which is automatically called in the environments `play`, `bidding` and `biddingpair` by means of `columnstype P` and `columnstype B` to convert at one hand the shorthand suit code in suit symbols and at the other hand to translate card honors into the active language. It also converts the card value `T` into `10` and a hyphen into an en-dash. It calls `\ODw@translate` to do the work.

The following code was contributed on StackExchange by egreg, see https://tex.stackexchange.com/questions/417731/problem-with-xstring-ifeqcase-case-falls-thru/417788?noredirect=1#comment1045001_417788

```

252 \ExplSyntaxOn
253 % NB: now all spaces are ignored, use '~' if needed.
254 \NewDocumentCommand{\ODw@Xfer}{m}{
255   \bgroup
256   % we do not want spaces here
257   \def\xspace{}
258   \tl_map_function:nN {#1} \ODw@translate:n
259   \egroup
260 }% ODw@Xfer

```

`\ODw@translate`

```
\ODw@translate{<tokens>}
```

`\ODw@translate` processes a (relatively short) string of tokens that determine an entry in `bidding` or `play` diagrams, and also in all situations where suits are defined. It gets called by `\ODw@Xfer`.

It expects bridge stuff describing strings like AKT54 to produce the suit **AK1052**, `2H\alert` to produce the call $2\heartsuit^*$ in the bidding diagram, or DA to produce $\heartsuit A$ as entry in the play diagram to show that the ace of diamonds was played. Please note that constructs like `\textit{DA}` or `\frame{2H}` are not allowed and will produce rather misleading errors like:

```
! Argument of \ODw@translate:n has an extra }.
```

or

```
! Missing number,treated as zero.
```

Even clever people who use `\frame{2H}` will get disappointed, because they'll get $2H$ rather than the wanted $2\heartsuit$. But the very clever people can reach their goal by using `\frame{2\He}` or `\textit{\Di A}`.

```

261 \cs_new_protected:Nn \ODw@translate:n
262 {
263   \setcounter{ODw@Rank}{0}
264   \str_case:nnTF {#1}
265     { % Store the suit of the card played
266       % needed to determine the winner
267       % and for checking for multiple cards
268       {C}{\Cl\gdef\ODw@SuitPlayed{C}}
269       {D}{\Di\gdef\ODw@SuitPlayed{D}}
270       {H}{\He\gdef\ODw@SuitPlayed{H}}
271       {S}{\Sp\gdef\ODw@SuitPlayed{S}}
272       {N}{\NT\gdef\ODw@SuitPlayed{N}}
273   % Translate a hyphen into an en-dash
274     {-}{--}
275   %
276   % 1: translate the honour cards,
277   % 2: store the played cards for checking
278   % 3: and set their rank. This must be done last, because
279   % \ODw@SetRank resets \ODw@SuitPlayed to \ODw@SuitLead
280   % Honour Cards
281   %           1           2
282   {A}{\Ace*!\ODw@AppendCard{\ODw@SuitPlayed}{A}
283     \ODw@SetRank{14}}% 3
284   {K}{\King*!\ODw@AppendCard{\ODw@SuitPlayed}{K}
285     \ODw@SetRank{13}}
286   {Q}{\Queen*!\ODw@AppendCard{\ODw@SuitPlayed}{Q}
287     \ODw@SetRank{12}}
288   {J}{\Jack*!\ODw@AppendCard{\ODw@SuitPlayed}{J}
289     \ODw@SetRank{11}}
290   {T}{\kern-0.1em1\kern-0.1em0% massage 1 and 0 a bit

```

```

291         \ODw@AppendCard{\ODw@SuitPlayed}{T}\ODw@SetRank{10}}
292 % Numeral Cards
293     {9}{9\ODw@AppendCard{\ODw@SuitPlayed}{9}\ODw@SetRank{9}}
294     {8}{8\ODw@AppendCard{\ODw@SuitPlayed}{8}\ODw@SetRank{8}}
295     {7}{7\ODw@AppendCard{\ODw@SuitPlayed}{7}\ODw@SetRank{7}}
296     {6}{6\ODw@AppendCard{\ODw@SuitPlayed}{6}\ODw@SetRank{6}}
297     {5}{5\ODw@AppendCard{\ODw@SuitPlayed}{5}\ODw@SetRank{5}}
298     {4}{4\ODw@AppendCard{\ODw@SuitPlayed}{4}\ODw@SetRank{4}}
299     {3}{3\ODw@AppendCard{\ODw@SuitPlayed}{3}\ODw@SetRank{3}}
300     {2}{2\ODw@AppendCard{\ODw@SuitPlayed}{2}\ODw@SetRank{2}}
301 % A spot card has rank 0
302     {x}{x\ODw@SetRank{0}}
303 %
304 % Non cards (bidding only)
305     {1}{1}% this enables e.g. 1\He in biddings
306     {p}{\ifODw@LongCalls\Pass*\else\Pass!\fi}
307     {P}{\ifODw@LongCalls\Allpass*\else\Allpass!\fi}
308     {X}{\ifODw@LongCalls\Double*\else\Double!\fi}
309     {R}{\ifODw@LongCalls\Redouble*\else\Redouble!\fi}
310 }% case
311 {% if matched (case T(true))
312     \ifODw@CardSkip\hspace{\ODw@Card@Skip}\fi
313 % suit of 1st card (ODw@SuitLead) is ODw@SuitPlayed
314     \if\theODw@PlayerNr1
315         \global\edef\ODw@SuitLead{\ODw@SuitPlayed}
316     \fi
317 }

```

We offer the possibility that one can use also `\He` in bidding and play diagrams rather than just the abbreviation `H`. Therefore we must test which suit was given and set `\ODw@SuitPlayed` accordingly. To make this test work, we had to redefine the suit macros with an `\renewrobustcmd`. Here we also issue `\expandafter{#1}` rather than just `#1`. Otherwise, among others, the coloring of the suit symbol would extend behind it. Curiously enough the phenomena does not occur anymore. I leave the `expandafter` in, until this is cleared.

```

318     {% if not matched (case F(false))
319         \ifx#1\C1\gdef\ODw@SuitPlayed{C}\fi
320         \ifx#1\D1\gdef\ODw@SuitPlayed{D}\fi
321         \ifx#1\He\gdef\ODw@SuitPlayed{H}\fi
322         \ifx#1\Sp\gdef\ODw@SuitPlayed{S}\fi
323         \expandafter{#1}% enables e.g. 1\He
324     % suit of 1st card (ODw@SuitLead) is ODw@SuitPlayed

```

```

325     \if\theODw@PlayerNr1
326         \global\edef\ODw@SuitLead{\ODw@SuitPlayed}
327     \fi
328 }
329 }% ODw@translate
330 \ExplSyntaxOff

```

\ODw@AppendCard

```
\ODw@AppendCard{<su<i>t</i>>}{<card>}
```

In order to do a simple consistency check in play diagrams, we need to store the cards that were played. We do that for each suit in the variable \ODw@<su<i>t</i>>. This macro is called in \ODw@translate, i.e. for all situations where cards are to be manipulated. But the result of \ODw@AppendCard is used only within play diagrams. The macro \ODw@appendcard appends 1 character to a string.

```

331 %
332 \newcommand{\ODw@appendcard}[2]{\xdef#1{#1#2}}
333
334 \newcommand\ODw@AppendCard[2]{%
335     \IfEqCase{#1}{%
336         {C}{\ODw@appendcard{\ODw@Clubs}{#2}}%
337         {D}{\ODw@appendcard{\ODw@Diamonds}{#2}}%
338         {H}{\ODw@appendcard{\ODw@Hearts}{#2}}%
339         {S}{\ODw@appendcard{\ODw@Spades}{#2}}%
340     }%
341 }% ODw@AppendCard

```

\ODw@PTfer

```
\ODw@PTfer{<tokens>}
```

This macro is called within play diagrams where we can write HA and get ♥A. Also all relevant symbols get translated into the active language. We use the counter ODw@PlayerNr to determine the column in the play diagrams with the winning card, and from this we can compute which player won the trick. \ODw@PTfer is essentially called for each entry in all columns of the play diagram through the column definition:

```
\newcolumnstype{P}{>{\collectcell\ODw@PTfer}c<{\endcollectcell}}
```

We first define two counters, both initially set to zero.

```

342 \newcounter{ODw@Highest}%     the highest rank until now
343 \setcounter{ODw@Highest}{0}
344 \newcounter{ODw@WinningNr}%   player with the highest rank

```

```

345 \setcounter{ODw@WinningNr}{0}
346
347 \def\ODw@PTfer#1{%
348   \stepcounter{ODw@PlayerNr}%
349   \ODw@Xfer{#1}% ODw@Rank = the rank for this card
350   \ifthenelse{\value{ODw@Rank} > \value{ODw@Highest}}{%
351     {% This rank is higher than previous highest one
352       \setcounter{ODw@WinningNr}{\theODw@PlayerNr}%
353       \setcounter{ODw@Highest}{\theODw@Rank}%
354     }%
355   }%
356   \ifthenelse{\value{ODw@PlayerNr} = 4}%
357     {% last player: Process the winning trick:
358       \stepcounter{ODw@Nr}% Start new row with new player
359       \ODw@AccTricks%      Accumulate tricks for N-S/E-W
360     }%
361   }%
362 }

```

\ODw@FTfer

```
\ODw@FTfer{<tokens>}
```

\ODw@FTfer is called for the first column of the play diagram TableII. In \ODw@Tricks it just resets ODw@PlayerNr and \ODw@Last and writes the player who leads. Finally it processes the entry of the first column by calling \ODw@PTfer. \ODw@FTfer is essentially called for the entries in the first column of TableII in the play diagram through the column definition:

```

\newcolumnstype{F}{>{\collectcell\ODw@FTfer}c<{\endcollectcell}}
363 \def\ODw@FTfer#1{%
364   \ODw@Tricks%
365   \ODw@PTfer{#1}%
366 }% ODw@FTfer

```

\ODw@BTfer

```
\ODw@BTfer{<tokens>}
```

This macro is called within bidding diagrams and enables us to type 1C\announce and get 1♣^A. The symbols get translated into the active language. \ODw@BTfer is essentially called for each entry in the bidding diagrams through:

```
\newcolumnstype{B}{>{\collectcell\ODw@BTfer}c<{\endcollectcell}}
```

As there is no special processing for the bidding entries, we call \ODw@Xfer right away to do the job.

```

367 \def\ODw@BTfer#1{%
368   \ODw@Xfer{#1}%
369 }

```

6.2.2 Names of Directions and Axes

\North

```
\North *!
```

\north

```
\north
```

```

370 \NewDocumentCommand\North{s t!}{%
371   \bgroup%
372   \ODw@OtherFont%
373   \IfBooleanTF#1{%
374     \IfBooleanTF{#2}{\ODw@N*}{\ODw@North*}%
375   }{%
376     \IfBooleanTF{#2}{\ODw@N}{\ODw@North}%
377   }% TF#1
378 \egroup%
379 \xspace%
380 }% North
381 %
382 \def\north{\North*}

```

\East

```
\East *!
```

\east

```
\east
```

```

383 \NewDocumentCommand\East{s t!}{%
384   \bgroup%
385   \ODw@OtherFont%
386   \IfBooleanTF#1{%
387     \IfBooleanTF{#2}{\ODw@E*}{\ODw@East*}%
388   }{%
389     \IfBooleanTF{#2}{\ODw@E}{\ODw@East}%
390   }% TF#1
391 \egroup%
392 \xspace%
393 }% East
394 %
395 \def\east{\East*}

```


\South

\South *!

\south

\south

```
396 \NewDocumentCommand\South{s t!}{%
397   \bgroup%
398     \ODw@OtherFont%
399     \IfBooleanTF#1{%
400       \IfBooleanTF{#2}{\ODw@S*}{\ODw@South*}%
401     }{%
402       \IfBooleanTF{#2}{\ODw@S}{\ODw@South}%
403     }% TF#1
404   \egroup%
405   \xspace%
406 }% South
407 %
408 \def\south{\South*}
```

\West

\West *!

\west

\west

```
409 \NewDocumentCommand\West{s t!}{%
410   \bgroup%
411     \ODw@OtherFont%
412     \IfBooleanTF#1{%
413       \IfBooleanTF{#2}{\ODw@W*}{\ODw@West*}%
414     }{%
415       \IfBooleanTF{#2}{\ODw@W}{\ODw@West}%
416     }% TF#1
417   \egroup%
418   \xspace%
419 }% West
420 %
421 \def\west{\West*}
```

\NorthSouth

\NorthSouth *!

\northsouth

\northsouth

```
422 \NewDocumentCommand\NorthSouth{s t!}{%
423   \bgroup%
```

```

424 \ODw@OtherFont%
425 \IfBooleanTF#1{%
426   \IfBooleanTF{#2}{\North*!--\South*!}{\North*--\South*}%
427 }{%
428   \IfBooleanTF{#2}{\North!--\South!}{\North--\South}%
429 }%
430 \egroup%
431 \xspace%
432 }% NorthSouth
433 %
434 \def\northsouth{\NorthSouth*}

```

\EastWest

```
\EastWest *!
```

\eastwest

```
\eastwest
```

```

435 \NewDocumentCommand\EastWest{s t!}{%
436   \bgroup%
437   \ODw@OtherFont%
438   \IfBooleanTF#1{%
439     \IfBooleanTF{#2}{\East*!--\West*!}{\East*--\West*}%
440   }{%
441     \IfBooleanTF{#2}{\East!--\West!}{\East--\West}%
442   }%
443   \egroup%
444   \xspace%
445 }% EastWest
446 %
447 \def\eastwest{\EastWest*}

```

Next we define macros that translate the short form of the directions into the active language.

```

\ODw@N
\ODw@E 448 \def\ODw@N{%
\ODw@S 449   \@ifstar{\translate{N-(ODw)}}%
\ODw@W 450   {\translate{n-(ODw)}}%
451 }
452 \def\ODw@E{%
453   \@ifstar{\translate{E-(ODw)}}%
454   {\translate{e-(ODw)}}%
455 }
456 \def\ODw@S{%

```

```

457 \@ifstar{\translate{S-(ODw)}}%
458         {\translate{s-(ODw)}}%
459 }
460 \def\ODw@W{%
461   \@ifstar{\translate{W-(ODw)}}%
462           {\translate{w-(ODw)}}%
463 }

```

\ODw@NS

```

\ODw@EW 464 \def\ODw@NS{\ODw@N--\ODw@S}
465 \def\ODw@EW{\ODw@E--\ODw@W}

```

\ODw@North

```

\ODw@East 466 \def\ODw@North{%
\ODw@South 467   \@ifstar{\translate{North-(ODw)}}%
\ODw@West 468           {\translate{north-(ODw)}}%
469 }
470 \def\ODw@East{%
471   \@ifstar{\translate{East-(ODw)}}%
472           {\translate{east-(ODw)}}%
473 }
474 \def\ODw@South{%
475   \@ifstar{\translate{South-(ODw)}}%
476           {\translate{south-(ODw)}}%
477 }
478 \def\ODw@West{%
479   \@ifstar{\translate{West-(ODw)}}%
480           {\translate{west-(ODw)}}%
481 }

```

6.2.3 Non-Bid Calls

\Pass

```
\Pass *!
```

\pass

```
\pass
```

```

482 \NewDocumentCommand\Pass{s t!}{%
483   \bgroup%
484   \ODw@OtherFont%
485   \IfBooleanTF{#1}{%
486     \IfBooleanTF{#2}%
487     {\translate{Pass!-(ODw)}}%

```

```

488     {\translate{Pass-(ODw)}}}%
489   }{%
490   \IfBooleanTF{#2}%
491     {\translate{pass!-(ODw)}}}%
492     {\translate{pass-(ODw)}}}%
493   }% TF#1
494 \egroup%
495 \xspace%
496 }% Pass
497 %
498 \def\pass{\Pass*}

```

\Allpass

\Allpass *!

\allpass

\allpass

```

499 \NewDocumentCommand\Allpass{s t!}{%
500   \bgroup%
501   \ODw@OtherFont%
502   \IfBooleanTF#1{%
503     \IfBooleanTF{#2}%
504     {\translate{AP-(ODw)}}}%
505     {\translate{All pass-(ODw)}}}%
506   }{%
507   \IfBooleanTF{#2}%
508     {\translate{ap-(ODw)}}}%
509     {\translate{all pass-(ODw)}}}%
510   }% TF#1
511 \egroup%
512 \xspace%
513 }% Allpass
514 %
515 \def\allpass{\Allpass*}

```

\Double

\Double *!

\double

\double

```

516 \NewDocumentCommand\Double{s t!}{%
517   \bgroup%
518   \ODw@OtherFont%
519   \IfBooleanTF#1{%

```

```

520     \IfBooleanTF{#2}%
521     {\translate{Db1-(ODw)}}}%
522     {\translate{Double-(ODw)}}}%
523   }{%
524     \IfBooleanTF{#2}%
525     {X}%
526     {\translate{double-(ODw)}}}%
527   }% TF#1
528 \egroup%
529 \xspace%
530 }% Double
531 %
532 \def\double{\Double*}

```

\Redouble

```
\Redouble *!
```

\redouble

```
\redouble
```

```

533 \NewDocumentCommand\Redouble{s t!}{%
534   \bgroup%
535   \ODw@OtherFont%
536   \IfBooleanTF#1{%
537     \IfBooleanTF{#2}%
538     {\translate{ReDb1-(ODw)}}}%
539     {\translate{ReDouble-(ODw)}}}%
540   }{%
541     \IfBooleanTF{#2}%
542     {\mbox{X\kern-0.1em X}}}%
543     {\translate{redouble-(ODw)}}}%
544   }% TF#1
545 \egroup%
546 \xspace%
547 }% Redouble
548 %
549 \def\redouble{\Redouble*}

```

6.2.4 Bidding Diagrams

\ODw@FirstBidCol

```
\ODw@FirstBidCol{N|S|E|W}
```

\ODw@FirstBidCol determines which player starts in the first bidding column. We also take care that the real world name of the players are

kept associated with the columns. For `\ODw@BidderX` ($X=I,II,III,IV$) we define two macros: `\ODw@BidderX` and `\ODw@BidderX*` in one. When these macros are called in the bidding diagram, the starred version writes the short notation whereas the unstarred one writes the long version.

```

550 \newcommand\ODw@FirstBidCol[1]{%
551 \IfEqCase{#1}{%
552   {N}{%
553     \def\ODw@BidderI{\@ifstar{\ODw@North*}{\ODw@N*}}%
554     \def\ODw@BidderII{\@ifstar{\ODw@East*}{\ODw@E*}}%
555     \def\ODw@BidderIII{\@ifstar{\ODw@South*}{\ODw@S*}}%
556     \def\ODw@BidderIV{\@ifstar{\ODw@West*}{\ODw@W*}}%
557     \def\ODw@NameI{\ODw@North@Name}%
558     \def\ODw@NameII{\ODw@East@Name}%
559     \def\ODw@NameIII{\ODw@South@Name}%
560     \def\ODw@NameIV{\ODw@West@Name}%
561   }%
562   {E}{%
563     \def\ODw@BidderI{\@ifstar{\ODw@East*}{\ODw@E*}}%
564     \def\ODw@BidderII{\@ifstar{\ODw@South*}{\ODw@S*}}%
565     \def\ODw@BidderIII{\@ifstar{\ODw@West*}{\ODw@W*}}%
566     \def\ODw@BidderIV{\@ifstar{\ODw@North*}{\ODw@N*}}%
567     \def\ODw@NameI{\ODw@East@Name}%
568     \def\ODw@NameII{\ODw@South@Name}%
569     \def\ODw@NameIII{\ODw@West@Name}%
570     \def\ODw@NameIV{\ODw@North@Name}%
571   }%
572   {S}{%
573     \def\ODw@BidderI{\@ifstar{\ODw@South*}{\ODw@S*}}%
574     \def\ODw@BidderII{\@ifstar{\ODw@West*}{\ODw@W*}}%
575     \def\ODw@BidderIII{\@ifstar{\ODw@North*}{\ODw@N*}}%
576     \def\ODw@BidderIV{\@ifstar{\ODw@East*}{\ODw@E*}}%
577     \def\ODw@NameI{\ODw@South@Name}%
578     \def\ODw@NameII{\ODw@West@Name}%
579     \def\ODw@NameIII{\ODw@North@Name}%
580     \def\ODw@NameIV{\ODw@East@Name}%
581   }%
582   {W}{%
583     \def\ODw@BidderI{\@ifstar{\ODw@West*}{\ODw@W*}}%
584     \def\ODw@BidderII{\@ifstar{\ODw@North*}{\ODw@N*}}%
585     \def\ODw@BidderIII{\@ifstar{\ODw@East*}{\ODw@E*}}%
586     \def\ODw@BidderIV{\@ifstar{\ODw@South*}{\ODw@S*}}%
587     \def\ODw@NameI{\ODw@West@Name}%

```

```

588 \def\ODw@NameII{\ODw@North@Name}%
589 \def\ODw@NameIII{\ODw@East@Name}%
590 \def\ODw@NameIV{\ODw@South@Name}%
591 }%
592 }% IfEqCase
593 }% ODw@FirstBidCol

```

Next we define the real world names for the **N-S** and the **E-W** bidders. We use `\ODw@All@Names` as variable to test if we have names for bidders at all: If it is empty, then no names were defined.

`\namesNS`

```
\namesNS{<N-name>}{<S-name>}
```

```

594 \newcommand\namesNS[2]{%
595 \gdef\ODw@North@Name{#1}%
596 \gdef\ODw@South@Name{#2}%
597 \gdef\ODw@All@Names{#1#2\ODw@East@Name\ODw@West@Name}%
598 }% namesNS

```

`\namesEW`

```
\namesEW{<E-name>}{<W-name>}
```

```

599 \newcommand\namesEW[2]{%
600 \gdef\ODw@East@Name{#1}%
601 \gdef\ODw@West@Name{#2}%
602 \gdef\ODw@All@Names{#1#2\ODw@North@Name\ODw@South@Name}%
603 }% namesEW

```

6.2.5 Diagram Hands

Here we implement commands that store the cards that each player holds. E.g. `\northhand` defines the complete hand, i.e. all 4 suits for the **North**-player. They have an optional parameter (an offset, default `opt`) to finetune the distance to the compass. In `\ODw@Nhand` and `\ODw@Shand` we use a makebox to prevent that its width goes beyond the **NESW** compass (but this can interfere with the Right-U/L-Legend). Within these macros, we check the consistency of the cards in the hand and also store card-information to check a complete deal later.

`\northhand`

```
\northhand[<v-offset>]{<Sp>}{<He>}{<Di>}{<Cl>}
```

```

\northhand[v-offset]{Sp}{He}{Di}{Cl}
      1      2      3      4      5
604 \newcommand\northhand[5][opt]{%
605 % check that north has 13 cards

```

```

606 \ODw@ChkNrOfCards{#2#3#4#5}{\north}%
607 \gdef\ODw@NSpades{#2}% Save norths cards
608 \gdef\ODw@NHearts{#3}% of all suits
609 \gdef\ODw@NDiamonds{#4}% for later
610 \gdef\ODw@NClubs{#5}% checking

```

\ODw@Nhand We fit the **North** hand in a box to avoid that a very long suit shifts the **East** hand to the right

```

611 \gdef\ODw@Nhand{%
612   \makebox[\ODw@Compasssize + 2ex][l]{%
613     \ODw@hand{t}{#2}{#3}{#4}{#5}%
614   }%
615   \vspace{#1}%
616 }%

617 }% northhand

```

\easthand `\easthand[h-offset]{Sp}{He}{Di}{Cl}`

```

\easthand[h-offset]{Sp}{He}{Di}{Cl}
      1      2      3      4      5

618 \newcommand\easthand[5][Opt]{%
619 % check that east has 13 cards
620 \ODw@ChkNrOfCards{#2#3#4#5}{\east}%
621 \gdef\ODw@ESpades{#2}% Save easts cards
622 \gdef\ODw@EHearts{#3}% of all suits
623 \gdef\ODw@EDiamonds{#4}% for later
624 \gdef\ODw@EClubs{#5}% checking

```

\ODw@Ehand

```

625 \gdef\ODw@Ehand{%
626   \hspace{#1}%
627   \ODw@hand{c}{#2}{#3}{#4}{#5}%
628 }%

629 }% easthand

```

\southhand `\southhand[v-offset]{Sp}{He}{Di}{Cl}`

```

\southhand[v-offset]{Sp}{He}{Di}{Cl}
      1      2      3      4      5

630 \newcommand\southhand[5][Opt]{%
631 % check that south has 13 cards
632 \ODw@ChkNrOfCards{#2#3#4#5}{\south}%

```



```

633 \gdef\ODw@SSpades{#2}% Save souths cards
634 \gdef\ODw@SHearts{#3}% of all suits
635 \gdef\ODw@SDiamonds{#4}% for later
636 \gdef\ODw@SClubs{#5}% checking

```

`\ODw@Shand` We fit the **South** hand in a box to avoid that a very long suit shifts the **East** hand to the right

```

637 \gdef\ODw@Shand{%
638   \parbox[b]{\ODw@Compasssize + 2ex}{%
639     \vspace*{#1}\par%
640     \makebox[Opt][1]{%
641       \ODw@hand{b}{#2}{#3}{#4}{#5}%
642     }%
643   }%
644 }%

645 }% southhand

```

`\westhand` `\westhand[h-offset]{Sp}{He}{Di}{Cl}`

```

\westhand[h-offset]{Sp}{He}{Di}{Cl}
      1      2  3  4  5

646 \newcommand\westhand[5][Opt]{%
647 % check that west has 13 cards
648 \ODw@ChkNrOfCards{#2#3#4#5}{\west}%
649 \gdef\ODw@WSpades{#2}% Save wests cards
650 \gdef\ODw@WHearts{#3}% of all suits
651 \gdef\ODw@WDiamonds{#4}% for later
652 \gdef\ODw@WClubs{#5}% checking

```

`\ODw@Whand`

```

653 \gdef\ODw@Whand{%
654   \ODw@hand{c}{#2}{#3}{#4}{#5}%
655   \hspace*{#1}%
656 }%

657 }% westhand

```

6.2.6 A Single Hand

Sometimes we want to show only (the cards of) one single hand.

`\hand *!- [pos]{Sp}{He}{Di}{Cl}`

```

658 \NewDocumentCommand\hand{s t! t- 0{c}mmmm}{%

```

`\hand* ! -[pos]{Sp}{He}{Di}{Cl}`
 1 2 3 4 5 6 7 8

This macro has 3 optional tokens that act as follows:

Naked mode: Displays the hand horizontally, left aligned.

* mode: Displays the hand horizontally, centered.

! mode: Displays the hand vertically, left aligned.

*! mode: Displays the hand vertically, centered.

- mode: Output is suppressed, it is stored in a savebox for later use.

The 4th argument is the aligning (used only in case of a vertical hand):
 default `c`. The rest of the arguments denote the cards of the suits.

We first check that this hand has 13 cards. Then we check for each suit that there are no multiples. Finally we store the hand in `\ODw@Hand@Box`. If output is not suppressed, we write the hand with `\usebox\ODw@Hand@Box`.

```

659 \ODw@ChkNrOfCards{#5#6#7#8}{Hand}%
660 \ODw@ChkSameCards{#5}{\Sp}%
661 \ODw@ChkSameCards{#6}{\He}%
662 \ODw@ChkSameCards{#7}{\Di}%
663 \ODw@ChkSameCards{#8}{\Cl}%
664 \global\sbox{\ODw@Hand@Box}{%
665   \bgroup%
666   \ODw@GameSize%
667   \IfBooleanTF{#2}%
668     {\ODw@vhand[#4]{#5}{#6}{#7}{#8}}%
669     {\ODw@hhand{#5}{#6}{#7}{#8}}%
670   \egroup%
671 }% sbox
672 \IfBooleanTF{#3}{-}{%
673   \IfBooleanTF{#1}%
674     {\centering \usebox{\ODw@Hand@Box}\par}}%
675     {\usebox{\ODw@Hand@Box}}%
676 }%
677 }% hand

```

`\ODw@hhand`

```
\ODw@hhand{<Sp>}{<He>}{<Di>}{<Cl>}
```

Displays a hand horizontally (e.g. ♠xxxx ♥xxx ♦xxx ♣xxx). We use a `tabular` with 4 columns in 1 row to print the 4 suits.

```

678 \newcommand\ODw@hhand[4]{%
679   \bgroup%
680   \def\xspace{}% undo xspace locally
681   % it screws the distance between suit and cards

```


```

682 \setlength\tabcolsep{1\ODw@Card@Skip}%
683 \begin{tabular}{lllll}
684 % we can't use \suit here: it would cause double checks!
685 \Sp\hspace{0.3em}\ODw@Cards{#1} &%
686 \He\hspace{0.3em}\ODw@Cards{#2} &%
687 \Di\hspace{0.3em}\ODw@Cards{#3} &%
688 \Cl\hspace{0.3em}\ODw@Cards{#4} \\
689 \end{tabular}%
690 \egroup%
691 }% ODw@hhand

```

\ODw@vhand

```
\ODw@vhand[\langle pos \rangle]{\langle Sp \rangle}{\langle He \rangle}{\langle Di \rangle}{\langle Cl \rangle}
```

Display a hand vertically (e.g. ). We call `\ODw@hand` to do the job.

```

692 \newcommand\ODw@vhand[5][c]{%
693 \ODw@hand{#1}{#2}{#3}{#4}{#5}%
694 }% ODw@vhand

```

\ODw@hand

```
\ODw@hand[\langle pos \rangle]{\langle Sp \rangle}{\langle He \rangle}{\langle Di \rangle}{\langle Cl \rangle}
```

We put a hand and some spacing in a `tabular` by reading the cards for each suit, making the lines more tense with the `\\[-0.5ex]`. Empty hands are discarded completely.

```

695 \newcommand\ODw@hand[5]{%
ODw@hand{pos}{spades}{hearts}{diamonds}{clubs}
          1      2      3      4      5

```

First we test if the hand is completely empty; only if not, we output something.

```

696 \ifthenelse{\equal{#2#3#4#5}{}}{}{%
697 \setlength\tabcolsep{\ODw@Card@Skip}%
698 %JW \ODw@GameSize% XYX JW 30.04.2018
699 \begin{tabular}[#1]{ll}%
700 % we can't use \suit here: it would cause double checks!
701 \Sp & \ODw@Cards{#2} \\[-0.5ex]
702 \He & \ODw@Cards{#3} \\[-0.5ex]
703 \Di & \ODw@Cards{#4} \\[-0.5ex]
704 \Cl & \ODw@Cards{#5} \\
705 \end{tabular}%
706 }% ifthenelse
707 }% ODw@hand

```

6.2.7 Suits

In some cases, we need only a collection of cards, without a suit symbol.

```
\onesuitAll \onesuitAll *! {\langle N \rangle}{\langle S \rangle}{\langle E \rangle}{\langle W \rangle}
```

Display the cards of one suit in a **NS–EW** diagram, with the **N**-, **E**-, **S**- and **W**-hand.

```
708 \NewDocumentCommand\onesuitAll{s t! mmmm}{%
onesuitAll* !{N-hand}{S-hand}{E-hand}{W-hand}
          1 2 3 4 5 6
```

Naked version: Use a

* version: Display the diagram centered

! version: use the NESW compass

First we test that we have no multiple cards in the suit. Then we use a tabular to place the cards around a compass or around a box.

```
709 \ODw@ChkSameCards{#3#4#5#6}{}%
710 \IfBooleanTF#1{\begin{center}}{}%
711 \bgroup%
712 \ODw@GameSize%
713 \setlength\tabcolsep{0em}%
714 \begin{tabular}{@{}r@{ }c@{ }l@{}}%
715 %\begin{tabular}{@{}rcl@{}}%
716 & \ODw@Cards{#3} \IfBooleanTF#2{\ll[-0.2em]}{\ll}%
717 \ODw@Cards{#6} & \IfBooleanTF#2{\ODw@Compass}{\ODw@Box}%
718 & \ODw@Cards{#5} \IfBooleanTF#2{\ll[-0.2em]}{\ll}%
719 & \ODw@Cards{#4}\ll
720 \end{tabular}%
721 \egroup%
722 \IfBooleanTF#1{\end{center}}{}%
723 }% onesuitAll
```

```
\onesuitNS \onesuitNS *! {\langle N \rangle}{\langle S \rangle}
```

Display a suit as **NS**-diagram. Similar to `\onesuitAll` but with only **N**- and **S**-hand.

```
724 \NewDocumentCommand\onesuitNS{s t! mm}{%
onesuitNS* !{N-hand}{S-hand}
          1 2 3 4
```

Naked version: Use a

```

* version: Display the diagram centered
! version: use the NESW compass

725 \ODw@ChkSameCards{#3#4}{}%
726 \IfBooleanTF#1{\begin{center}}{}%
727 \bgroup%
728 \ODw@GameSize%
729 \begin{tabular}{@{}c@{}}%
730 \ODw@Cards{#3}\IfBooleanTF#2{\[-0.2em]}{\}%
731 \IfBooleanTF#2{\ODw@Compass}\[-0.2em]}{\ODw@Box}\}%
732 \ODw@Cards{#4}%
733 \end{tabular}%
734 \egroup%
735 \IfBooleanTF#1{\end{center}}{}%
736 }% onesuitNS

```

\onesuitEW

```
\onesuitEW *! {\langle E \rangle}{\langle W \rangle}
```

Display a suit as EW diagram. Similar to \onesuitA11 but with only E- and W-hand.

```

737 \NewDocumentCommand\onesuitEW{s t! mm}{%
onesuitEW* !{E-hand}{W-hand}

```

```

1 2 3 4

```

Naked version: Use a

```

* version: Display the diagram centered
! version: use the NESW compass

```

```

738 \ODw@ChkSameCards{#3#4}{}%
739 \IfBooleanTF#1{\begin{center}}{}%
740 \bgroup%
741 \ODw@GameSize%
742 \begin{tabular}{@{}r@{ }c@{ }l@{}}%
743 \ODw@Cards{#4} &%
744 \IfBooleanTF#2{\ODw@Compass}{\ODw@Box} &%
745 \ODw@Cards{#3} \\\%
746 \end{tabular}%
747 \egroup%
748 \IfBooleanTF#1{\end{center}}{}%
749 }% onesuitEW

```

\onesuitNE

```
\onesuitNE *! {\langle N \rangle}{\langle E \rangle}
```

Display a suit as NE diagram. Similar to \onesuitA11 but with only N- and E-hand.

```

750 \NewDocumentCommand\onesuitNE{s t! mm}{%
onesuitNE* ![N-hand]{E-hand}
      1 2 3 4
Naked version: Use a  $\square$ 
* version: Display the diagram centered
! version: use the NESW compass

751 \ODw@ChkSameCards{#3#4}{}%
752 \IfBooleanTF#1{\begin{center}}{ }%
753 \bgroup%
754 \ODw@GameSize%
755 \begin{tabular}[b]{c@{ }l@{ }}%
756 \ODw@Cards{#3} \\\%
757 \IfBooleanTF#2{\ODw@Compass}{\ODw@Box} &%
758 \ODw@Cards{#4} \\\%
759 \end{tabular}%
760 \egroup%
761 \IfBooleanTF#1{\end{center}}{ }%
762 }% onesuitEW

```

`\onesuitNW` $\text{!} \{ \langle N \rangle \} \{ \langle W \rangle \}$

Display a suit as NW diagram. Similar to `\onesuitAll` but with only N- and W-hand.

```

763 \NewDocumentCommand\onesuitNW{s t! mm}{%
onesuitNW* ![N-hand]{W-hand}
      1 2 3 4
Naked version: Use a  $\square$ 
* version: Display the diagram centered
! version: use the NESW compass

764 \ODw@ChkSameCards{#3#4}{}%
765 \IfBooleanTF#1{\begin{center}}{ }%
766 \bgroup%
767 \ODw@GameSize%
768 \begin{tabular}[b]{r@{ }c}%
769 & \ODw@Cards{#3} \\\%
770 \ODw@Cards{#4} & \IfBooleanTF#2{\ODw@Compass}{\ODw@Box} \\\%
771 \end{tabular}%
772 \egroup%
773 \IfBooleanTF#1{\end{center}}{ }%
774 }% onesuitEW

```

`\suit` $\text{[} \langle \textit{suit symbol} \rangle \text{]} \{ \langle \textit{cards} \rangle \}$

Command for displaying the cards of a suit. With the optional argument one can add a suit symbol to the suit.

```

775 \newcommand\suit[2] []{%
\suit[suit]{cards}
      1      2
776 \ODw@ChkNrOfCards{#2}{suit}%
777 \ODw@ChkSameCards{#2}{#1}%
778 \bgroup% keep font change local
779 %JW
780 \ODw@GameSize%
781 #1\ODw@Cards{#2}%
782 \egroup%
783 \xspace%
784 }% suit

```

\ODw@Cards

```
\ODw@Cards{\langle cards \rangle}
```

This macro gets called by `\suit` and all commands that process hands. It processes the ranks of the cards. Between ranks some space is typeset. Care is taken that `\tau` becomes **10** and honor cards are translated into the active language.

```
785 \newcommand{\ODw@Cards}[1]{%
```

We enable the `cardskip` and call `\ODw@Xfer` to do the job

```

786 \ODw@CardSkiptrue%
787 \ODw@Xfer{#1}%
788 }% \ODw@Cards

```

6.2.8 Card Diagrams

Next we define several diagrams with hands around the compass. `\ODw@GameSize` sets the size of the compass, the directions and the hands, according to the actual font or font-size. `\ODw@LeftUpperText` etc. displays extra text, that appear in the left upper, etc. corner of the diagram. If `\headline` (`\footline`) is the empty string, we set boolean `\ODw@EmptyHeader` (`\ODw@EmptyFooter`) to true¹⁶. This value is used in `\ODw@ProcessHeader` to conditionally span the 3 columns with the headline- (`\footline`)text. We first store the diagram in a box, so we can calculate its width and use that as a size to limit the header/footer texts.

¹⁶see page 35, last paragraph why this is necessary

\showAll

```
\showAll *+ [pos]
```

Define the diagram, showing the cards for **All** hands.

```
789 \NewDocumentCommand\showAll{s t+ O{c}}{%
```

```
showAll* + [pos]
```

```
1 2 3
```

Display the NS--EW diagram, defined by \northhand, etc

* Version: Display the diagram centered

+ Version: Also display a bidding diagram

pos: aligning, default= c

First, for all suits we store all cards of all sides together. Next we check the consistency of all complete suits. The individual suits of each player have already been checked as we defined the hands. Finally we print the hands around the compass using a `tabular`, taking care of the additions above, below and in the corners of the diagram. Before we write the diagram, we store it in an `sbox` to calculate its width, so we can use that in other places.

```
790 \gdef\ODw@Spades{% store all Spades together
791   \ODw@NSpades\ODw@ESpades\ODw@SSpades\ODw@WSpades%
792 }%
793 \gdef\ODw@Hearts{% store all Hearts together
794   \ODw@NHearts\ODw@EHearts\ODw@SHearts\ODw@WHearts%
795 }%
796 \gdef\ODw@Diamonds{% store all Diamonds together
797   \ODw@NDiamonds\ODw@EDiamonds\ODw@SDiamonds\ODw@WDiamonds%
798 }%
799 \gdef\ODw@Clubs{% store all Clubs together
800   \ODw@NClubs\ODw@EClubs\ODw@SClubs\ODw@WClubs%
801 }%
802 % check for multiple and nr. of cards
803 \ODw@ChkNrOfCards{\ODw@Spades}{\Sp}%
804 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
805 \ODw@ChkNrOfCards{\ODw@Hearts}{\He}%
806 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
807 \ODw@ChkNrOfCards{\ODw@Diamonds}{\Di}%
808 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
809 \ODw@ChkNrOfCards{\ODw@Clubs}{\Cl}%
810 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
811 \IfBooleanTF#1{\begin{center}}{\}%
812 \bgroup%
813   \setlength\tabcolsep{0em}%
```



```

814 \ODw@GameSize%
815 \ODw@TestIfEmpty{\ODw@HeaderText}{ODw@EmptyHeader}%
816 \ODw@TestIfEmpty{\ODw@FooterText}{ODw@EmptyFooter}%
817 % sbox1 necessary to calc. |Compasssize| for |Nhand|
818 \sbox1{\ODw@Compass}%
819 \sbox0{%
820 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
821 \ODw@LeftUpperText & \ODw@Nhand & \ODw@RightUpperText\\
822 \ODw@Whand & \usebox{1} & \ODw@Ehand\\
823 \ODw@LeftLowerText & \ODw@Shand & \ODw@RightLowerText\\
824 \end{tabular}}%
825 }% sbox
826 \ODw@gsetlength{\ODw@Diagram@Width}{\wd0}%
827 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
828 \ODw@ProcessHeader{3}% span 3 columns
829 \ODw@LeftUpperText & \ODw@Nhand & \ODw@RightUpperText\\
830 \ODw@Whand & \usebox{1} & \ODw@Ehand\\
831 \ODw@LeftLowerText & \ODw@Shand & \ODw@RightLowerText\\
832 \ODw@ProcessFooter{3}% span 3 columns
833 \end{tabular}}%
834 \IfBooleanTF#2{%
835 % needed for \ODw@CondNewLine
836 \setlength{\ODw@Bid@Width}{\wd\ODw@BidBox}%
837 \ODw@CondNewLine%
838 \usebox{\ODw@BidBox}%
839 }{}%
840 \egroup%
841 \IfBooleanTF#1{\end{center}}{}%
842 }% showAll

```

\showNS

```
\showNS *+ [(pos)] (<N/S>)
```

Define the diagram, showing the cards for the **N-S** hands.

```
843 \NewDocumentCommand\showNS{s t+ 0{c} d()}{%
```

showNS* +[pos] (N/S)

1 2 3 4

Display the NS diagram, defined by \northhand, etc

* Version: Display the diagram centered

+ Version: Also display a bidding diagram

pos: aligning, default c

N/S: only N-hand (or S-hand) is to be displayed

Description: similar to \showAll

```
844 % For all suits store all cards of north and south together
845 \gdef\ODw@Spades{\ODw@NSpades\ODw@SSpades}%
846 \gdef\ODw@Hearts{\ODw@NHearts\ODw@SHearts}%
847 \gdef\ODw@Diamonds{\ODw@NDiamonds\ODw@SDiamonds}%
848 \gdef\ODw@Clubs{\ODw@NClubs\ODw@SClubs}%
849 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
850 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
851 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
852 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
853 \IfBooleanTF#1{\begin{center}}{}%
854 \bgroup%
855 \setlength\tabcolsep{0em}%
856 \ODw@GameSize%
857 \ODw@TestIfEmpty{\ODw@HeaderText}{\ODw@EmptyHeader}%
858 \ODw@TestIfEmpty{\ODw@FooterText}{\ODw@EmptyFooter}%
```

Here we store the width of the diagram **without** the header and footer.
So we can limit their width to the diagramwidth.

```
859 % sbox1 necessary to calc. |Compasssize| for |Nhand|
860 \sbox1{\ODw@Compass}%
861 \sbox0{%
862 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
863 % Display the N-hand only with token 'N', or no token at all
864 & \IfNoValueTF{#4}{\ODw@Nhand}{\ifthenelse{\equal{#4}{N}}{\ODw@Nhand}} & \
865 & \usebox{1} & \
866 % Display the S-hand only with token 'S', or no token at all
867 & \IfNoValueTF{#4}{\ODw@Shand}{\ifthenelse{\equal{#4}{S}}{\ODw@Shand}} & \
868 \end{tabular}}%
869 }% sbox0
870 \ODw@gsetlength\ODw@Diagram@Width{\wd0}%
871 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
872 \ODw@ProcessHeader{3}% span 3 columns
873 % Display the N-hand only with token 'N', or no token at all
874 & \IfNoValueTF{#4}{\ODw@Nhand}{\ifthenelse{\equal{#4}{N}}{\ODw@Nhand}} & \
875 & \usebox{1} & \
876 % Display the S-hand only with token 'S', or no token at all
877 & \IfNoValueTF{#4}{\ODw@Shand}{\ifthenelse{\equal{#4}{S}}{\ODw@Shand}} & \
878 \ODw@ProcessFooter{3}% span 3 columns
879 \end{tabular}}%
880 \IfBooleanTF#2{%
881 % necessary for \ODw@CondNewLine
882 \setlength{\ODw@Bid@Width}{\wd\ODw@BidBox}%
883 \ODw@CondNewLine%
```

```

884     \usebox{\ODw@BidBox}%
885   }{}%
886 \egroup%
887 \IfBooleanTF#1{\end{center}}{}%
888 }% showNS

```

\showEW

```
\showEW *+ [pos] (<E/W>)
```

Define the diagram, showing the cards for the **E–W** hands.

```
889 \NewDocumentCommand\showEW{s t+ 0{c} d()}{}%
```

```
showEW* + [pos] (E/W)
```

```
1 2 3 4
```

Display the EW diagram, defined by \easthand, etc

* Version: Display the diagram centered

+ Version: Also display a bidding diagram

pos: aligning, default c

E/W: only E-hand (or W-hand) is to be displayed

Description: similar to \showAll

```
890 % For all suits put all cards of east and west together
```

```
891 \gdef\ODw@Spades{\ODw@ESpades\ODw@WSpades}%
```

```
892 \gdef\ODw@Hearts{\ODw@EHearts\ODw@WHearts}%
```

```
893 \gdef\ODw@Diamonds{\ODw@EDiamonds\ODw@WDiamonds}%
```

```
894 \gdef\ODw@Clubs{\ODw@EClubs\ODw@WClubs}%
```

```
895 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
```

```
896 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
```

```
897 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
```

```
898 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
```

```
899 \IfBooleanTF#1{\begin{center}}{}%
```

```
900 \bgroup%
```

```
901 \setlength\tabcolsep{0em}%
```

```
902 \ODw@GameSize%
```

```
903 \ODw@TestIfEmpty{\ODw@HeaderText}{\ODw@EmptyHeader}%
```

```
904 \ODw@TestIfEmpty{\ODw@FooterText}{\ODw@EmptyFooter}%
```

```
905 \sbox0{%
```

```
906 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
```

```
907 % Display the W-hand only with token 'W', or no token at all
```

```
908 \IfNoValueTF{#4}{\ODw@Whand}{\ifthenelse{\equal{#4}{W}}{\ODw@Whand}{}}
```

```
909 & \ODw@Compass &
```

```
910 % Display the E-hand only with token 'E', or no token at all
```

```
911 \IfNoValueTF{#4}{\ODw@Ehand}{\ifthenelse{\equal{#4}{E}}{\ODw@Ehand}{}} \
```

```
912 \end{tabular}%
```

```

913 }% sbox
914 \ODw@gsetlength{\ODw@Diagram@Width}{\wd0}%
915 \begin{tabular}[#3]{@{}r@{}c@{}l@{}}%
916 \ODw@ProcessHeader{3}% span 3 columns
917 & & & \\\
918 % Display the W-hand only with token 'W', or no token at all
919 \IfNoValueTF{#4}{\ODw@Whand}{\ifthenelse{equal{#4}{W}}{\ODw@Whand}{}}
920 & \ODw@Compass &
921 % Display the E-hand only with token 'E', or no token at all
922 \IfNoValueTF{#4}{\ODw@Ehand}{\ifthenelse{equal{#4}{E}}{\ODw@Ehand}{}} \\\
923 \ODw@ProcessFooter{3}% span 3 columns
924 \end{tabular}%
925 \IfBooleanTF#2{%
926 % necessary for \ODw@CondNewLine
927 \setlength{\ODw@Bid@Width}{\wd\ODw@BidBox}%
928 \ODw@CondNewLine%
929 \usebox{\ODw@BidBox}%
930 }{}%
931 \egroup%
932 \IfBooleanTF#1{\end{center}}{}%
933 }% showEW

```

\showNE

```
\showNE *+ [(pos)]
```

Define the diagram, showing the cards for the **N-E** hands.

```
934 \NewDocumentCommand\showNE{s t+ O{c}}{%
```

```
showNE* +[pos]
```

```
1 2 3
```

Display the NE diagram, defined by \northhand, etc

* Version: Display the diagram centered

+ Version: Also display a bidding diagram

pos: aligning, default c

Description: similar to \showAll

```

935 % For all suits put all cards of north and east together
936 \gdef\ODw@Spades{\ODw@NSpades\ODw@ESpades}%
937 \gdef\ODw@Hearts{\ODw@NHearts\ODw@EHearts}%
938 \gdef\ODw@Diamonds{\ODw@NDiamonds\ODw@EDiamonds}%
939 \gdef\ODw@Clubs{\ODw@NClubs\ODw@EClubs}%
940 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
941 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
942 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
943 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
944 \IfBooleanTF#1{\begin{center}}{}%

```

```

945 \bgroup%
946   \setlength\tabcolsep{0em}%
947   \ODw@GameSize%
948   \ODw@TestIfEmpty{\ODw@HeaderText}{ODw@EmptyHeader}%
949   \ODw@TestIfEmpty{\ODw@FooterText}{ODw@EmptyFooter}%
950 % sbox1 necessary to calc. |Compasssize| for |Nhand|
951   \sbox1{\ODw@Compass}%
952   \sbox0{%
953     \begin{tabular}[#3]{@{}c@{}l@{}}%
954       \ODw@Nhand & \ODw@RightUpperText\\
955       \usebox{1} & \ODw@Ehand\\
956     \end{tabular}%
957   }% sbox
958   \ODw@gsetlength{\ODw@Diagram@Width}{\wd0}%
959   \begin{tabular}[#3]{@{}c@{}l@{}}%
960     \ODw@ProcessHeader{2}% span 2 columns
961     \ODw@Nhand & \ODw@RightUpperText\\
962     \usebox{1} & \ODw@Ehand\\
963     \ODw@ProcessFooter{2}% span 2 columns
964   \end{tabular}%
965   \IfBooleanTF#2{%
966 % necessary for \ODw@CondNewLine
967   \setlength{\ODw@Bid@Width}{\wd\ODw@BidBox}%
968   \ODw@CondNewLine%
969   \usebox{\ODw@BidBox}%
970 }{}%
971 \egroup%
972 \IfBooleanTF#1{\end{center}}{}%
973 }% showNE

```

\showNW

```
\showNW *+ [pos]
```

Define the diagram, showing the cards for the **N–W** hands.

```
974 \NewDocumentCommand\showNW{s t+ 0{c}}{%
```

```
showNW* +[pos]
```

```
1 2 3
```

Display the NW diagram, defined by \northhand, etc

* Version: Display the diagram centered

+ Version: Also display a bidding diagram

pos: aligning, default c

Description: similar to \showAll

```
975 % For all suits put all cards of north and west together
```

```
976 \gdef\ODw@Spades{\ODw@NSpades\ODw@WSpades}%
```

```

977 \gdef\ODw@Hearts{\ODw@NHearts\ODw@WHearts}%
978 \gdef\ODw@Diamonds{\ODw@NDiamonds\ODw@WDiamonds}%
979 \gdef\ODw@Clubs{\ODw@NClubs\ODw@WClubs}%
980 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
981 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
982 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
983 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
984 \IfBooleanTF#1{\begin{center}}{}%
985 \bgroup%
986 \setlength\tabcolsep{0em}%
987 \ODw@GameSize%
988 \ODw@TestIfEmpty{\ODw@HeaderText}{ODw@EmptyHeader}%
989 \ODw@TestIfEmpty{\ODw@FooterText}{ODw@EmptyFooter}%
990 % sbox1 necessary to calc. |Compasssize| for |Nhand|
991 \sbox1{\ODw@Compass}%
992 \sbox0{%
993 \begin{tabular}[#3]{@{}c@{}l@{}}%
994 \ODw@LeftUpperText & \ODw@Nhand\\
995 \ODw@Whand & \usebox{1}\\
996 \end{tabular}%
997 }% sbox
998 \ODw@gsetlength{\ODw@Diagram@Width}{\wd0}%
999 \begin{tabular}[#3]{@{}r@{}c@{}}%
1000 \ODw@ProcessHeader{2}% span 2 columns
1001 \ODw@LeftUpperText & \ODw@Nhand\\
1002 \ODw@Whand & \usebox{1}\\
1003 \ODw@ProcessFooter{2}% span 2 columns
1004 \end{tabular}%
1005 \IfBooleanTF#2{%
1006 % necessary for \ODw@CondNewLine
1007 \setlength{\ODw@Bid@Width}{\wd\ODw@BidBox}%
1008 \ODw@CondNewLine%
1009 \usebox{\ODw@BidBox}%
1010 }{}%
1011 \egroup%
1012 \IfBooleanTF#1{\end{center}}{}%
1013 }% showNW

```

6.2.9 The Compass

When displaying the compass, the square with N-S and E-W axes, we try to achieve several things:

1. Making the size font-size dependent

2. Put both `N` and `S` horizontally centered
3. Put both `E` and `W` vertically centered
4. Print the vulnerable side in red if in colored mode
5. Underline the dealer (we *overline* `S` for better clarity)

We use the mapping as shown in the tables below. The `U` stands for undefined. This reflects the situation where neither `\vulner` nor `\dealer` have been called, and also there is no board number known.

Player	
N	0
E	1
S	2
W	3

Vulner	
none	0
all	1
N-S	2
E-W	3
U	-1

Dealer	
N	0
E	1
S	2
W	3
U	-1

`\ODw@Compass`

```
\ODw@Compass
```

```
1014 \newcommand{\ODw@Compass}{%
```

The codes for dealership (`\ODw@D`) and vulnerability (`\ODw@V`) are used in `\ODw@Print`. We initialize them with the value `-1` to denote the undefined state. `\@ODw` acts as a local temp variable in order to make a smooth comparison.

```
1015 \begingroup
1016 \def\ODw@V{-1}\def\ODw@D{-1}%
1017 %
1018 % Set the code for vulnerability
1019 %
1020 \def\@ODw{\none}\ifx\ODw@Vulner\@ODw\def\ODw@V{0}\fi%
1021 \def\@ODw{\none*}\ifx\ODw@Vulner\@ODw\def\ODw@V{0}\fi%
1022 \def\@ODw{\none!}\ifx\ODw@Vulner\@ODw\def\ODw@V{0}\fi%
1023 \def\@ODw{\none*!}\ifx\ODw@Vulner\@ODw\def\ODw@V{0}\fi%
1024 \def\@ODw{\all}\ifx\ODw@Vulner\@ODw\def\ODw@V{1}\fi%
1025 \def\@ODw{\all*}\ifx\ODw@Vulner\@ODw\def\ODw@V{1}\fi%
1026 \def\@ODw{\all!}\ifx\ODw@Vulner\@ODw\def\ODw@V{1}\fi%
1027 \def\@ODw{\all*!}\ifx\ODw@Vulner\@ODw\def\ODw@V{1}\fi%
1028 \def\@ODw{\NorthSouth}\ifx\ODw@Vulner\@ODw\def\ODw@V{2}\fi%
1029 \def\@ODw{\NorthSouth*}\ifx\ODw@Vulner\@ODw\def\ODw@V{2}\fi%
1030 \def\@ODw{\NorthSouth!}\ifx\ODw@Vulner\@ODw\def\ODw@V{2}\fi%
1031 \def\@ODw{\NorthSouth*!}\ifx\ODw@Vulner\@ODw\def\ODw@V{2}\fi%
1032 \def\@ODw{\EastWest}\ifx\ODw@Vulner\@ODw\def\ODw@V{3}\fi%
```

```

1033 \def\@ODw{\EastWest*}\ifx\ODw@Vulner\@ODw\def\ODw@V{3}\fi%
1034 \def\@ODw{\EastWest!}\ifx\ODw@Vulner\@ODw\def\ODw@V{3}\fi%
1035 \def\@ODw{\EastWest*!}\ifx\ODw@Vulner\@ODw\def\ODw@V{3}\fi%
1036 %
1037 % Set the code for dealership
1038 %
1039 \def\@ODw{\North}\ifx\ODw@Dealer\@ODw\def\ODw@D{0}\fi%
1040 \def\@ODw{\North*}\ifx\ODw@Dealer\@ODw\def\ODw@D{0}\fi%
1041 \def\@ODw{\North!}\ifx\ODw@Dealer\@ODw\def\ODw@D{0}\fi%
1042 \def\@ODw{\North*!}\ifx\ODw@Dealer\@ODw\def\ODw@D{0}\fi%
1043 \def\@ODw{\East}\ifx\ODw@Dealer\@ODw\def\ODw@D{1}\fi%
1044 \def\@ODw{\East*}\ifx\ODw@Dealer\@ODw\def\ODw@D{1}\fi%
1045 \def\@ODw{\East!}\ifx\ODw@Dealer\@ODw\def\ODw@D{1}\fi%
1046 \def\@ODw{\East*!}\ifx\ODw@Dealer\@ODw\def\ODw@D{1}\fi%
1047 \def\@ODw{\South}\ifx\ODw@Dealer\@ODw\def\ODw@D{2}\fi%
1048 \def\@ODw{\South*}\ifx\ODw@Dealer\@ODw\def\ODw@D{2}\fi%
1049 \def\@ODw{\South!}\ifx\ODw@Dealer\@ODw\def\ODw@D{2}\fi%
1050 \def\@ODw{\South*!}\ifx\ODw@Dealer\@ODw\def\ODw@D{2}\fi%
1051 \def\@ODw{\West}\ifx\ODw@Dealer\@ODw\def\ODw@D{3}\fi%
1052 \def\@ODw{\West*}\ifx\ODw@Dealer\@ODw\def\ODw@D{3}\fi%
1053 \def\@ODw{\West!}\ifx\ODw@Dealer\@ODw\def\ODw@D{3}\fi%
1054 \def\@ODw{\West*!}\ifx\ODw@Dealer\@ODw\def\ODw@D{3}\fi%
1055 %

```

We use a picture environment and set its size to $2.5em \times 2.5em$ by setting the `\PicSize` to 500 and the `unitlength` to $0.005em$. Doing this enables us to avoid floating point arithmetic in the calculations of positions. Both `\PicSize` and `\MidSize` are local to `\ODw@Compass` and skipped from indexing. The same goes for `\Hoffset` and `\Voffset`.

```

1056 \ODw@CompassDefault% use the compass font
1057 \def\PicSize{500}%
1058 \def\MidSize{250}%
1059 % Multiply unitlength=0.005em with CompSize (default= 1)
1060 \setlength\unitlength{0.005em * \real{\ODw@CompSize}}%
1061 \ODw@gsetlength\ODw@Compasssize{\unitlength * \PicSize + 2ex}%
1062 \def\Hoffset{30}% distance between W (E) and frame
1063 \def\Voffset{30}% distance between N (S) and frame
1064 \setlength\ODw@Tmp@Width{0.1em * \real{\ODw@CompLine}}%
1065 \linethickness{\ODw@Tmp@Width}%
1066 % leave 1ex space on all sides
1067 \parbox[c][\ODw@Compasssize]{\ODw@Compasssize}{%
1068 \centering%
1069 \begin{picture}(\PicSize,\PicSize)%
1070 \ifODw@CompShow%

```



```

1071     % the frame
1072     \moveto(0,0)
1073     \if\ODw@CompLine0% must do it this way, because
1074     \else% linethickness zero does not suppress the line
1075     \lineto(0,\PicSize)\lineto(\PicSize,\PicSize)
1076     \lineto(\PicSize,0)\closepath\strokepath
1077     \fi%
1078     % the cardinal points
1079     \put(\MidSize,\the\numexpr \PicSize - \Voffset)%
1080     {\makebox[Opt]{\raisebox{-\height}{\ODw@Print{0}}}}% N
1081     \put(\MidSize,\Voffset){\makebox[Opt]{\ODw@Print{2}}}% S
1082     \put(\Hoffset,\MidSize){%
1083     \makebox[Opt][l]{%
1084     \ifODw@CompTurn%
1085     \raisebox{-0.5\height}{%
1086     \rotatebox[origin=t]{90}{\ODw@Print{3}}%
1087     }%
1088     \else%
1089     \raisebox{-0.5\height}{\ODw@Print{3}}%
1090     \fi%
1091     }% makebox
1092     }% W
1093     \put(\the\numexpr \PicSize - \Hoffset,\MidSize)%
1094     {\makebox[Opt][r]{%
1095     \raisebox{-0.5\height}{%
1096     \ifODw@CompTurn%
1097     \rotatebox[origin=c]{90}{\ODw@Print{1}}%
1098     \else%
1099     \ODw@Print{1}%
1100     \fi%
1101     }%
1102     }% makebox
1103     }% E
1104     % the center
1105     \put(\MidSize,\MidSize){\makebox(0,0){\ODw@mid}}
1106     \fi%
1107     \end{picture}%
1108     }% parbox
1109     \endgroup
1110 }% ODw@Compass

```

\ODw@Print \ODw@Print{*player-code*}

\ODw@Print prints N, E, S and W in the compass. The side that is vul-

nerable is printed in red (or italics if we are monochrome), otherwise in black. The dealer is under- or overlined.

The `\ifcase` distinguishes between the players. Then dealership and vulnerability are tested. `\ODw@PrintColor` is called to actually print the player.

```

1111 \newcommand\ODw@Print[1]{%
\ODw@Print{player-code}
      1 (player-code=0-3)
1112 \bgroup
1113 \smaller\smaller%
1114 \ifcase#1% #1=0: print N
1115 \ifboolexpr{ test {\ifnumcomp{\ODw@D}{=}{0}}}%
1116   {% dealer = N
1117     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1118                 test {\ifnumcomp{\ODw@V}{=}{2}} }%
1119     {\ODw@PrintColor{\underline{\ODw@N*}}}% Vul
1120     {\underline{\ODw@N*}}% not Vul
1121   }{% dealer <> N
1122     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1123                 test {\ifnumcomp{\ODw@V}{=}{2}} }%
1124     {\ODw@PrintColor{\ODw@N*}}% Vul
1125     {\ODw@N*}% not Vul
1126   }%
1127 \or% #1=1: print E
1128 \ifboolexpr{ test {\ifnumcomp{\ODw@D}{=}{1}}}%
1129   {% dealer E
1130     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1131                 test {\ifnumcomp{\ODw@V}{=}{3}} }%
1132     {\ODw@PrintColor{\underline{\ODw@E*}}}% Vul
1133     {\underline{\ODw@E*}}% not Vul
1134   }{% dealer <> E
1135     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1136                 test {\ifnumcomp{\ODw@V}{=}{3}} }%
1137     {\ODw@PrintColor{\ODw@E*}}% Vul
1138     {\ODw@E*}% not Vul
1139   }%
1140 \or% #1=2: print S
1141 \ifboolexpr{ test {\ifnumcomp{\ODw@D}{=}{2}}}%
1142   {% dealer S
1143     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1144                 test {\ifnumcomp{\ODw@V}{=}{2}} }%
1145     {\ODw@PrintColor{%

```

```

1146         \ensuremath{\overline{\mbox{\ODw@S*}}}%
1147     }% Vul
1148     \ensuremath{\overline{\mbox{\ODw@S*}}}% not Vul
1149 }{% dealer <> S
1150     \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1151                 test {\ifnumcomp{\ODw@V}{=}{2}} }%
1152         {\ODw@PrintColor{\ODw@S*}}% Vul
1153         {\ODw@S*}% not Vul
1154     }%
1155 \or% #1=3: print W
1156     \ifboolexpr{ test {\ifnumcomp{\ODw@D}{=}{3}}}%
1157     {% dealer W
1158         \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1159                     test {\ifnumcomp{\ODw@V}{=}{3}} }%
1160         {\ODw@PrintColor{\underline{\ODw@W*}}}% Vul
1161         {\underline{\ODw@W*}}% not Vul
1162     }{% dealer <> W
1163         \ifboolexpr{ test {\ifnumcomp{\ODw@V}{=}{1}} or%
1164                     test {\ifnumcomp{\ODw@V}{=}{3}} }%
1165         {\ODw@PrintColor{\ODw@W*}}% Vul
1166         {\ODw@W*}% not Vul
1167     }%
1168 \fi% (ifcase#1)
1169 \egroup%
1170 }% ODw@Print

```

`\ODw@PrintColor` `\ODw@PrintColor{|N|E|S|W}`

`\ODw@PrintColor` checks if we are in monochrome mode. If that is the case we print N, E, S or W in italics, otherwise in color.

```

1171 \newcommand\ODw@PrintColor[1]{%
1172   \ifODw@monochrome\textit{#1}\else\textcolor{red}{#1}\fi%
1173 }% ODw@PrintColor

```

`\ODw@mid` Hook to write something in the middle of the compass. We write what is stored in `\ODw@CompMid` a bit smaller than the N-S and E-W letters. `\ODw@mid` writes the contents of `\ODw@CompMid` in the middle of the compass. As there is only very limited room, this text should be very short. It is primarily meant to write just a board number in the compass.

`\ODw@CompMid`

```

1174 \def\ODw@CompMid{}
1175 \def\ODw@mid{{\smaller\smaller\smaller\ODw@CompMid}}

```

```

\ODw@Box Displays a  $\square$ 
1176 \newcommand{\ODw@Box}{%
1177   \bgroup
1178   \setlength{\fboxsep}{0pt}%
1179   \setlength{\fboxrule}{0.1em}%
1180   \fbox{\rule{0mm}{0.7em}\rule{0.7em}{0mm}}%
1181   \egroup
1182 }% \ODw@Box

```

6.2.10 Diagram Conditions

Here we implement several macros that add some board information to the card diagram. `\ODw@BoardText` serves as a variable to store the user-defined or (by means of the board number) auto-generated text, concerning the board. The macro `\boardnr` has 1 mandatory argument. If it is a number, it is considered to be the board number. The dealer and which side is vulnerable is then calculated from it and stored by calling `\dealer` resp. `\vulner`. If it is not a positive integer, it is considered user-defined text which is stored 'as is' in `\ODw@BoardText`. The contents can be retrieved by the user by calling `\boardtext` to actually print the board information.

```
\ODw@BoardText
```

```
1183 \def\ODw@BoardText{}
```

```
\boardtext
```

```
\boardtext *
```

`\boardtext` has only 1 token and no arguments. The unstarred form outputs only the text stored in `\ODw@BoardText`, this is normally a board number. `\boardtext*` outputs something like 'Board: 23'.

```

1184 \NewDocumentCommand\boardtext{s}{%
1185   \IfBooleanTF#1{\translate{Board-(ODw)}:\,\ODw@BoardText}%
1186   {\ODw@BoardText}%
1187 }% \boardtext

```

```
\boardnr
```

```
\boardnr{Nr}
```

`\boardnr{Nr}` sets the dealership and vulnerability according to `Nr`. As the system repeats itself after the 16th board, we canonize `Nr` to the range of 1–16. We also set `\ODw@BoardText` accordingly. For the association between boardnumber and dealer/vulner, see file *Compass* of the *onedown-example* collection.

```
1188 \newcommand{\boardnr}[1]{%
```

```

1189 \IfInteger{#1}{%
1190   \gdef\ODw@BoardText{%
1191     \bgroup%
1192     \ODw@OtherFont%
1193 %     \translate{Board-(ODw)} #1%
1194     #1%
1195     \egroup%
1196   }%
1197   \setcounter{ODw@Cnt}{#1}%
1198   \whiledo{\theODw@Cnt > 16}{%
1199     \addtocounter{ODw@Cnt}{-16}%
1200   }% whiledo, now 1 <= Cnt <= 16
1201   \IfEqCase{\theODw@Cnt}{% set dealer/vulner
1202 % Board 0 = no board: mark dealer and vulner undefined
1203     {0}{\gdef\ODw@BoardText{\vulner[-1]\dealer[-1]}
1204     {1}{\vulner[\none]\dealer[\North*!]}
1205     {2}{\vulner[\NorthSouth*!]\dealer[\East*!]}
1206     {3}{\vulner[\EastWest*!]\dealer[\South*!]}
1207     {4}{\vulner[\all]\dealer[\West*!]}
1208     {5}{\vulner[\NorthSouth*!]\dealer[\North*!]}
1209     {6}{\vulner[\EastWest*!]\dealer[\East*!]}
1210     {7}{\vulner[\all]\dealer[\South*!]}
1211     {8}{\vulner[\none]\dealer[\West*!]}
1212     {9}{\vulner[\EastWest*!]\dealer[\North*!]}
1213     {10}{\vulner[\all]\dealer[\East*!]}
1214     {11}{\vulner[\none]\dealer[\South*!]}
1215     {12}{\vulner[\NorthSouth*!]\dealer[\West*!]}
1216     {13}{\vulner[\all]\dealer[\North*!]}
1217     {14}{\vulner[\none]\dealer[\East*!]}
1218     {15}{\vulner[\NorthSouth*!]\dealer[\South*!]}
1219     {16}{\vulner[\EastWest*!]\dealer[\West*!]}
1220   }% IfEqCase
1221 }{\gdef\ODw@BoardText{#1}}% otherwise take #1
1222 }% boardnr

```

The next macros are used to add some game information above resp. below the card diagram. \ODw@HeaderText and \ODw@FooterText are used as variables to store the user-defined text.

```
\headlinetext \headlinetext{<text>}
```

```
\ODw@HeaderText
```

```

1223 \newcommand\headlinetext[1]{\gdef\ODw@HeaderText{#1}}
1224 \headlinetext{}

```

```
\footlinetext    \footlinetext{<text>}
```

```
\ODw@FooterText
```

```
1225 \newcommand\footlinetext [1]{\gdef\ODw@FooterText{#1}}
1226 \footlinetext{}
```

The next macros are used to add some game information in the corners of the card diagram. We use a `tabular` with 1 column and 3 lines to do so.

We redefine the (originally empty) `Left[Upper/Lower]-` and `Right[Upper/Lower]Text`, and set it to the wanted value. The first (optional) parameter defines some horizontal extra space if the hand and a condition text collide. The commands have 3 mandatory arguments, each for 1 of the 3 condition lines, which may be empty.

```
\leftupper    \leftupper[<h-offset>]{<line1>}{<line2>}{<line3>}
```

```
\ODw@LeftUpperText
```

```
1227 \def\ODw@LeftUpperText{}
1228 \newcommand\leftupper [4] [Opt]{%
1229    \gdef\ODw@LeftUpperText{%
1230     \hspace{-#1}%
1231     \begin{tabular}[t]{l}{#2}\#3\#4\end{tabular}
1232    }%
1233 }% leftupper
```

```
\leftlower    \leftlower[<h-offset>]{<line1>}{<line2>}{<line3>}
```

```
\ODw@LeftLowerText
```

```
1234 \def\ODw@LeftLowerText{}
1235 \newcommand\leftlower [4] [Opt]{%
1236    \gdef\ODw@LeftLowerText{%
1237     \hspace{-#1}%
1238     \begin{tabular}[b]{l}{#2}\#3\#4\end{tabular}
1239    }%
1240 }% leftlower
```

```
\rightupper    \rightupper[<h-offset>]{<line1>}{<line2>}{<line3>}
```

```
\ODw@RightUpperText
```

```
1241 \def\ODw@RightUpperText{}
```

```

1242 \newcommand\rightupper[4] [Opt]{%
1243   \gdef\ODw@RightUpperText{%
1244     \hspace{#1}%
1245     \begin{tabular}[t]{t}{l}#2\\#3\\#4\\\end{tabular}
1246   }%
1247 }% rightupper

```

```
\rightlower \rightlower[h-offset]{line1}{line2}{line3}
```

\ODw@RightLowerText

```

1248 \def\ODw@RightLowerText{}
1249 \newcommand\rightlower[4] [Opt]{%
1250   \gdef\ODw@RightLowerText{%
1251     \hspace{#1}%
1252     \begin{tabular}[b]{b}{l}#2\\#3\\#4\\\end{tabular}
1253   }%
1254 }% rightlower

```

\ODw@ProcessHeader

```
\ODw@ProcessHeader{N}
```

\ODw@ProcessHeader[N] puts HeaderText in a multicolumn which spans N columns.

```
1255 \newcommand{\ODw@ProcessHeader}[1]{%
```

Programmers note:

\ODw@TestIfEmpty cannot be called in here. The \ifthenelse called within the tabular environment leads to the error:

```
! Misplaced \omit. \multispan ->\omit \@multispan
```

So the test of the emptiness of Header- and FooterText is done out of the tabular. Why is (La)TeX always causing unexpected problems?:-(We set the headline/footerline text to the width of the diagram with a solution found at:

<https://tex.stackexchange.com/questions/125005/how-to-create-a-table-where-one-cell-spans-all-the-columns-and-the-text-wraps-pr>

```

1256   \ifODw@EmptyHeader% Must be this way (StackExchange)
1257   \else%               |\ifthenelse| bites |\multicolumn|!
1258     \multicolumn{#1}{%
1259       p{\dimexpr\ODw@Diagram@Width-%
1260         2\tabcolsep-2\arrayrulewidth}%
1261       }{\ODw@LegendFont\ODw@HeaderText}}\
1262   \fi%
1263 }% ODw@ProcessHeader

```

`\ODw@TestIfEmpty`

`\ODw@TestIfEmpty{<Str>}{<Bool>}`

Tests the emptiness of a string.

```
1264 \newcommand{\ODw@TestIfEmpty}[2]{%
\ODw@TestIfEmpty{Str}{Bool}
  sets boolean Bool to true if string Str is empty
1265   \ifthenelse{\equal{#1}{}}{%
1266     \setboolean{#2}{true}}{%
1267     \setboolean{#2}{false}}%
1268   }%
1269 }% \ODw@TestIfEmpty
```

`\ODw@ProcessFooter`

`\ODw@ProcessFooter{<N>}`

`\ODw@ProcessFooter[N]` puts FooterText in a multicolumn which spans N columns.

```
1270 \newcommand{\ODw@ProcessFooter}[1]{%
1271   \ifODw@EmptyFooter% Must be this way (StackExchange)
1272   \else%           |\ifthenelse| bites |\multicolumn|!
1273     \multicolumn{#1}{%
1274       p{\dimexpr\ODw@Diagram@Width%
1275         -2\tabcolsep-2\arrayrulewidth}%
1276       }{\ODw@LegendFont\ODw@FooterText}}\
1277   \fi%
1278 }% \ODw@ProcessFooter
```

`\handskip`

`\handskip{<length>}`

`\handskip` adds <length> to SkipWidth, i.e. the distance between the card diagram (with or without the east hand) and the bidding diagram.

```
1279 \newcommand\handskip[1]{%
1280   \def\ODw@Skipwidth{1em + #1}% recalculate the new Skipwidth
1281   \setlength\ODw@Skip@Width{\ODw@Skipwidth}%
1282 }% \handskip
```

`\ODw@DealerText`

Typesets the string *Dealer*.

```
1283 \def\ODw@DealerText{%
1284   \bgroup%
1285     \ODw@OtherFont\translate{Dealer-(ODw)}%
1286   \egroup%
1287 }% \ODw@DealerText
```


\ODw@VulnerText

\ODw@VulnerText *! vulnerable (Vulnerable, vul, Vul)

Typesets the string *vulnerable* or *Vul*.

```
1288 \NewDocumentCommand\ODw@VulnerText{s t!}{%
1289   \bgroup%
1290     \ODw@OtherFont%
1291     \IfBooleanTF#1{%
1292       \IfBooleanTF#2{\translate{Vul-(ODw)}}%
1293       {\translate{Vulnerable-(ODw)}}%
1294     }{%
1295       \IfBooleanTF#2{\translate{vul-(ODw)}}%
1296       {\translate{vulnerable-(ODw)}}%
1297     }%
1298   \egroup%
1299   \xspace%
1300 }% ODw@VulnerText
```

\ODw@AwardText

Typesets the text to be put as 'title' in the *award* part of a expert quiz.

```
1301 \def\ODw@AwardText{\textsf{\translate{Award-(ODw)}}}
```

\dealer

\dealer[*dealer*]

If #1 = empty then set \ODw@Dealer to #1 else output \ODw@Dealer

```
1302 \newcommand\dealer[1] []{%
1303   \ifthenelse{\equal{#1}{}}%
1304     {\ODw@Dealer}%
1305     {\gdef\ODw@Dealer{#1}}%
1306 }% dealer
```

\ODw@Dealer

set **North** as default

```
1307 \def\ODw@Dealer{\North*}
```

\vulner

\vulner[*vulner*]

If #1 = empty then set \ODw@Vulner to #1 else output \ODw@Vulner

```
1308 \newcommand\vulner[1] []{%
1309   \ifthenelse{\equal{#1}{}}%
1310     {\ODw@Vulner}%
1311     {\gdef\ODw@Vulner{#1}}%
1312 }% vulner
```

\ODw@Vulner

set **north-south** as default

```
1313 \def\ODw@Vulner{\NorthSouth}
```

```

\dealertext
1314 \newcommand\dealertext[1][\ODw@Dealer]{\ODw@DealerText:\,#1}

\vulnertext
1315 \newcommand\vulnertext[1][\ODw@Vulner]{%
1316   \ifODw@LongCalls%
1317     \ODw@VulnerText*%
1318   \else%
1319     \ODw@VulnerText*!%
1320   \fi%
1321   :\,#1%
1322 }

\alert
1323 \newcommand{\alert}{{}\ensuremath{\textbf{\textasteriskcentered}}}}

\announce
1324 \newcommand{\announce}{{}\ensuremath{\textbf{\smaller A}}}}

\markit
\markit

1325 \newcommand\markit{%
Sets markers a, b etc. To be used only in bidding diagrams.
1326   \stepcounter{ODw@Nr}%
1327   \footnotemark[\theODw@Nr]%
1328 }% markit

\explainit
\explainit{\textit{}}

1329 \newcommand\explainit[1]{%
Explains the marked items. To be used only in bidding diagrams. The
counter ODw@Nr associates the marker with the explanation.
1330   \stepcounter{ODw@Nr}%
1331   \ensuremath{{}\texttrm{\smaller\alpha{ODw@Nr}}}\,#1%
1332 }% explainit

```

6.3 The Bidding Environments

6.3.1 Special Columntypes

`\newcolumntype` In order to automatically apply a macro call on all cell contents of a column (translate/convert; step a counter) in `bidding` and `play` diagrams, we define `newcolumntypes`, made possible by loading package `collectcell`. We define next `columntypes`:

B: Transfers suits and cards (bidding and play)
 F: sets First column in play diagrams
 P: Transfers suits and cards, accumulates won tricks (play)

```

1333 \newcolumnntype{B}{% for Biddings
1334   >{\collectcell\ODw@BTfer}c<{\endcollectcell}}
1335 \newcolumnntype{F}{% for First column in |play| diagrams
1336   >{\collectcell\ODw@FTfer}c<{\endcollectcell}}
1337 \newcolumnntype{P}{% for |Play| diagrams (2nd--4th column)
1338   >{\collectcell\ODw@PTfer}c<{\endcollectcell}}

```

6.3.2 The Hidden Implementation

As explained before, in the bidding diagrams we convert S to the spade symbol ♠, etc. We do need packages array and collectcell for this and define the columnntype B. In order to avoid these conversions in the top row, where names are displayed, we use the \cci-trick to suppress expansion of the cell macro. Curiously the command \cci (from collcell, v0.5, 2011/02/27) sometimes produces unwanted characters. The reason is unknown to me. We use a space (" ") as first character in \cci to avoid this.

We store the bidding diagram without the explanations in a box, so we can calculate the width of the bidding diagram and make our explanation part exactly as wide. In the first row we write the bidders: North etc. (or N, if it has to be short). In the second row we write the real world names of the bidders, if given. If any description is given, we make a multicolumn over the 4 rows with the previously stored width to write the explanations. We also use this width to calculate whether the bidding diagram will fit on the actual line. If not, we put it on a new line.

ODw@Bidding

```

\ODw@Bidding[<pos>](<description>)... \endODw@Bidding

```

```

1339 \NewEnviron{ODw@Bidding}[2][t]{%
1340   \def\xspace{}%
1341   \setlength\tabcolsep{0.2em}%
1342   \sbox{0}{%
1343     \begin{tabular}[#1]{BBBB}% 1st column
1344     \ifODw@Bidders%
1345       \cci{ % there MUST be a ' ' (space)
1346         \ODw@BidderFont%
1347         \ifODw@short\ODw@BidderI%
1348         \else\ODw@BidderI*%

```

```

1349         \fi%
1350     } &%                2nd column
1351     \ccif % there MUST be a ' ' (space)
1352         \ODw@BidderFont%
1353         \ifODw@short\ODw@BidderII%
1354         \else\ODw@BidderII*%
1355         \fi%
1356     } &%                3rd column
1357     \ccif % there MUST be a ' ' (space)
1358         \ODw@BidderFont%
1359         \ifODw@short\ODw@BidderIII%
1360         \else\ODw@BidderIII*%
1361         \fi%
1362     } &%                4th column
1363     \ccif % there MUST be a ' ' (space)
1364         \ODw@BidderFont%
1365         \ifODw@short\ODw@BidderIV%
1366         \else\ODw@BidderIV*%
1367         \fi%
1368     } \\\% end of 1st row
1369     \if\ODw@All@Names\empty%
1370     \else% 2nd row
1371         \ccif \ODw@NameFont\ODw@NameI} &% please
1372         \ccif \ODw@NameFont\ODw@NameII} &% mind
1373         \ccif \ODw@NameFont\ODw@NameIII} &% the
1374         \ccif \ODw@NameFont\ODw@NameIV} \\\% spaces!
1375     \fi%
1376     \ifODw@BidLine\hline\fi%
1377     \fi%
1378     \BODY%
1379 %%%
1380     \end{tabular}%
1381 }% sbox0
1382 \setcounter{ODw@Nr}{0}%
1383 \setlength{\ODw@Bid@Width}{\wd0}%
1384 \global\sbox\ODw@BidBox{%
1385     \begin{tabular}[#1]{BBBB}% 1st column
1386     \ifODw@Bidders%
1387     \ccif % there MUST be a ' ' (space)
1388         \ODw@BidderFont%
1389         \ifODw@short\ODw@BidderI%
1390         \else\ODw@BidderI*%
1391         \fi%

```

```

1392 } &% 2nd column
1393 \cci{ % there MUST be a ' ' (space)
1394 \ODw@BidderFont%
1395 \ifODw@short\ODw@BidderII%
1396 \else\ODw@BidderII*%
1397 \fi%
1398 } &% 3rd column
1399 \cci{ % there MUST be a ' ' (space)
1400 \ODw@BidderFont%
1401 \ifODw@short\ODw@BidderIII%
1402 \else\ODw@BidderIII*%
1403 \fi%
1404 } &% 4th column
1405 \cci{ % there MUST be a ' ' (space)
1406 \ODw@BidderFont%
1407 \ifODw@short\ODw@BidderIV%
1408 \else\ODw@BidderIV*%
1409 \fi%
1410 } \\\% end of 1st row
1411 \if\ODw@All@Names\empty%
1412 \else% 2nd row
1413 \cci{ \ODw@NameFont\ODw@NameI} &% please
1414 \cci{ \ODw@NameFont\ODw@NameII} &% mind
1415 \cci{ \ODw@NameFont\ODw@NameIII} &% the
1416 \cci{ \ODw@NameFont\ODw@NameIV} \\\% spaces!
1417 \fi%
1418 \ifODw@BidLine\hline\fi%
1419 \fi
1420 \BODY%
1421 %%% Until here the same code as in the sbox!
1422 \ifODw@description%
1423 % Add the description, if not empty
1424 \hline%
1425 \multicolumn{4}{% span explanations over 4 cols...
1426 p{\dimexpr\ODw@Bid@Width-2\tabcolsep}%
1427 }{% ...with the right width
1428 \setcounter{ODw@Nr}{0}%
1429 \raggedright%
1430 \smaller\smaller#2%
1431 }\\%
1432 \fi%
1433 \end{tabular}%
1434 }% sbox ODw@BidBox

```

1435 }% ODw@Bidding

ODw@Biddingpair

```
\ODw@Biddingpair[{pos}]({description})...\endODw@Biddingpair
```

Decription: Similar to ODw@Bidding

```
1436 \NewEnviron{ODw@Biddingpair}[2][t]{%
1437   \def\xspace{}%
1438   \setlength\tabcolsep{0.2em}%
1439   \sbox{0}{%
1440     \begin{tabular}[#1]{BB}% 1st column
1441     \ifODw@Bidders%
1442       \ccif % there MUST be a ' ' (space)
1443       \ODw@BidderFont%
1444       \ifODw@short\ODw@BidderI%
1445       \else\ODw@BidderI*%
1446       \fi%
1447     } &% 2nd column
1448     \ccif % there MUST be a ' ' (space)
1449     \ODw@BidderFont%
1450     \ifODw@short\ODw@BidderIII%
1451     \else\ODw@BidderIII*%
1452     \fi%
1453   } \\% end of 1st row
1454   \if\ODw@All@Names\empty%
1455   \else% 2nd row
1456     \ccif \ODw@NameFont\ODw@NameI} &% please mind
1457     \ccif \ODw@NameFont\ODw@NameIII} \\% the spaces!
1458     \fi%
1459     \ifODw@BidLine\hline\fi%
1460     \fi%
1461     \BODY%
1462 %%%}%
1463   \end{tabular}%
1464 }% sbox0
1465 \setcounter{ODw@Nr}{0}%
1466 \setlength{\ODw@Bid@Width}{\wd0}%
1467 \global\sbox\ODw@BidBox{%
1468   \begin{tabular}[#1]{BB}% 1st column
1469   \ifODw@Bidders%
1470     \ccif % there MUST be a ' ' (space)
1471     \ODw@BidderFont%
1472     \ifODw@short\ODw@BidderI%
1473     \else\ODw@BidderI*%
1474     \fi%
```

```

1475     } &% 2nd column
1476     \ccif % there MUST be a ' ' (space)
1477         \ODw@BidderFont%
1478         \ifODw@short\ODw@BidderIII%
1479         \else\ODw@BidderIII*%
1480         \fi%
1481     } \\\% end of 1st row
1482     \if\ODw@All@Names\empty%
1483     \else% 2nd row
1484         \ccif \ODw@NameFont\ODw@NameI} &% please mind
1485         \ccif \ODw@NameFont\ODw@NameIII} \\\% the spaces!
1486         \fi%
1487         \ifODw@BidLine\hline\fi%
1488         \fi%
1489         \BODY%
1490     %%% Until here the same code as in the sbox!
1491     \ifODw@description%
1492     % Add the description, if not empty
1493         \hline%
1494         \multicolumn{2}{%
1495             p{\dimexpr\ODw@Bid@Width-2\tabcolsep}%
1496         }{%
1497             \setcounter{ODw@Nr}{0}%
1498             \raggedright%
1499             \smaller\smaller#2%
1500         } \\\%
1501         \fi%
1502     \end{tabular}%
1503 }% sbox ODw@BidBox
1504 }% ODw@Biddingpair

```

6.4 The User Environments

6.4.1 Bidding

The `bidding` environments have 2 optional arguments: an alignment [*pos*] and an annotation (*description*). There are also 3 tokens: the `*` centers the bidding diagram, the `+` forces the short notation, i.e. **N** rather than **North** and the `-` suppresses all output.

`\ODw@GameSize` takes care of the font dependent sizing of the diagram. We locally redefine `\thefootnote` and reset the (general) counter `ODw@Nr`, which is stepped in `\markit` and `\explainit` to make the annotations correspond. In the end code we define a `multicolumn` over all 4

(2) columns and write the annotation given in argument #2. With `p{...\ODw@Bid@Width...}` care is taken to limit this text to the width of the diagram.

bidding

```
\begin{bidding} *!- [pos](description)...\end{bidding}
```

```
1505 \NewDocumentEnvironment{bidding}{s t! t- 0{c}d()}{%
```

```
\begin{bidding}* ! -[pos](description)
```

```
1 2 3 4 5
```

The 1st token (*) centers the environment; the 2nd token (!) switches to the short notation in the table header; the 3rd token (-) suppresses the output. Argument 4 regulates the alignment of the table (default is `c` and the 5th argument contains the annotations of the bidding.

```
1506 \ODw@GameSize%
```

```
1507 \renewcommand{\thefootnote}{\alph{footnote}}%
```

```
1508 \setcounter{ODw@Nr}{0}%
```

```
1509 \IfBooleanTF#1{\center}{}% "*" detected
```

```
1510 \IfBooleanTF#2{\ODw@shorttrue}{}% "!" detected
```

```
1511 \IfValueTF#5{\ODw@descriptiontrue}{\ODw@descriptionfalse}%
```

```
1512 \ODw@Bidding[#4]{#5}%
```

```
1513 }{%
```

```
1514 \endODw@Bidding%
```

```
1515 \IfBooleanTF#3{%
```

```
1516 {\rule{0pt}{0pt}}%
```

```
1517 % +---without this, pdflatex aborts compilation!
```

```
1518 {\usebox{\ODw@BidBox}}% "-" detected
```

```
1519 \IfBooleanTF#1{\endcenter}{}%
```

```
1520 }% bidding
```

biddingpair

```
\begin{biddingpair} *!- [pos](description)...\end{biddingpair}
```

```
1521 \NewDocumentEnvironment{biddingpair}{s t! t- 0{c}d()}{%
```

```
\begin{biddingpair}* ! -[pos](description)
```

```
1 2 3 4 5
```

The same as with environment `bidding`, only with 2 columns instead of 4.

```
1522 \def\xspace{}%
```

```
1523 \ODw@GameSize%
```

```
1524 \renewcommand{\thefootnote}{\alph{footnote}}%
```

```
1525 \setcounter{ODw@Nr}{0}%
```

```
1526 \IfBooleanTF#1{\center}{}% "*" detected
```



```

1527 \IfBooleanTF{#2}{\ODw@shorttrue}{}% "+" detected
1528 \IfValueTF{#5}{\ODw@descriptiontrue}{\ODw@descriptionfalse}%
1529 \ODw@Biddingpair[#4]{#5}%
1530 }{%
1531 \endODw@Biddingpair%
1532 \IfBooleanTF{#3}%
1533   {\rule{0pt}{0pt}}%
1534 %      +---without this, pdflatex aborts compilation!
1535   {\usebox{\ODw@BidBox}}% "-" detected
1536 \IfBooleanTF#1{\endcenter}{}%
1537 }% biddingpair

```

6.4.2 Play

Environment `play` displays the sequence of playing tricks. It uses 2 `newcolumn`types:

F to increment and display the current row.

P to translate an convert suits/ranks.

`\ODw@AccTricks` `\ODw@AccTricks` calculates and shows the accumulated tricks in `play` that N-S and E-W has won. The winning card is detected automatically and `\ODw@LastTrick` is called to process the winning trick for whichever side won it (N-S or E-W) by stepping the counter for the winning side. The counter `ODw@Player` denotes the player who won the trick. From the player who leads and the position that wins we calculate the winning player and step the counter for his side.

```

1538 \def\ODw@AccTricks{%
1539   \ODw@LastTrick{\ODw@Last}{\theODw@WinningNr}%
1540   \ODw@append{\theODw@NSCnt,\theODw@EWCnt,}% store counters
1541   \setcounter{ODw@Highest}{0}% reset for next trick
1542 }% ODw@AccTricks

```

`\ODw@Tricks` `\ODw@Tricks`

`\ODw@Last` This macro is automatically called in `TableII` for column 1. This column displays the player who had the lead. It essentially 1) resets the `ODw@PlayerNr` which will be stepped for each next column in search for the winning card 2) remembers in `\ODw@Last` who had the lead. From these two values we can later calculate who won this trick. The stepping of `ODw@PlayerNr` occurs in `\ODw@Tfer`.

```

1543 \def\ODw@Tricks{%

```

```

1544 \setcounter{ODw@PlayerNr}{0}%
1545 \gdef\ODw@Last{\ODw@NextLead}%
1546 \expandafter\translate%
1547 \expandafter{\ODw@NextLead-(ODw)}:\,%
1548 % write a colon and a thin space in the table,
1549 % as separator between lead player and lead card.
1550 }% ODw@Tricks

```

\ODw@LastTrick

```
\ODw@LastTrick{<Player>}{<Pos>}
```

This macro is called by \ODw@AccTricks. It computes who won the last trick and steps the corresponding counter.

```
1551 \newcommand\ODw@LastTrick[2]{%
```

```
\ODw@LastTrick{Player}{Pos}
```

```

1          2--- Seat Nr that won the trick
+----- Player (N,E,S,W) who has led

```

Consider the following table, where the seats are in horizontal direction, starting with the player who leads in seat 1. Vertically, in the first column, we have an initial value, stored in ODw@Cnt which is associated with the leading player.

	1	2	3	4
0	W ¹	N ²	E ³	S ⁴
1	N ²	E ³	S ⁴	W ⁵
2	E ³	S ⁴	W ⁵	N ⁶
3	S ⁴	W ⁵	N ⁶	E ⁷

If we add this initial value to the *seat* where the trick is won, then the result gives us the *player* who won the trick. Suppose that e.g. S had the lead, so ODw@Cnt = 3. Suppose also that seat number 3 wins the trick. The sum equals 6 and this is the seat of N. For clarity these sums are displayed in the table as superscripts to the players

First we set the counter ODw@Cnt to the player who has the lead and add the seat number (#2) to it. We then store who has the next lead in \ODw@NextLead and increment the counter of the winning side.

```

1552 \IfEqCase{#1}{%
1553   {W}{\setcounter{ODw@Cnt}{0}}%
1554   {N}{\setcounter{ODw@Cnt}{1}}%
1555   {E}{\setcounter{ODw@Cnt}{2}}%
1556   {S}{\setcounter{ODw@Cnt}{3}}%
1557 }% IfEqCase
1558 \addtocounter{ODw@Cnt}{#2}%
1559 \IfEqCase{\theODw@Cnt}{%

```

```

1560 {1}{\gdef\ODw@NextLead{W}\stepcounter{ODw@EWCnt}}
1561 {2}{\gdef\ODw@NextLead{N}\stepcounter{ODw@NSCnt}}
1562 {3}{\gdef\ODw@NextLead{E}\stepcounter{ODw@EWCnt}}
1563 {4}{\gdef\ODw@NextLead{S}\stepcounter{ODw@NSCnt}}
1564 {5}{\gdef\ODw@NextLead{W}\stepcounter{ODw@EWCnt}}
1565 {6}{\gdef\ODw@NextLead{N}\stepcounter{ODw@NSCnt}}
1566 {7}{\gdef\ODw@NextLead{E}\stepcounter{ODw@EWCnt}}
1567 }% IfEqCase
1568 }% ODw@LastTrick

```

```

play \begin{play}*{\Lead}[\Trump]...\end{play}

```

Finally we define environment `play`. It consists primarily of these 3 tables, the middle one with the special newcolumn type **P**. We reset the counter for the running line (=trick) `ODw@Nr` and the winning trick counters for N-S and E-W: `ODw@NSCnt` and `ODw@EWCnt`. As usual `\ODw@GameSize` takes care for the correct sizing. The first row (the title row) is displayed using the `\cci` method. The 3th table gets a stacked N/S and E/W title.

```

1569 \NewDocumentEnvironment{play}{s m0{N}}{%
1570 % #1 --> s center
1571 % #2 --> m lead
1572 % #3 --> 0 trumpsuit (default NoTrump)
1573 \def\ODw@TrumpSuit{#3}
1574 \gdef\ODw@NextLead{#2}%
1575 \setcounter{ODw@Nr}{0}%
1576 \setcounter{ODw@NSCnt}{0}
1577 \setcounter{ODw@EWCnt}{0}%
1578 \ODw@GameSize%
1579 \ODw@Scratch{}% make empty
1580 \let\ODw@Clubs\empty%
1581 \let\ODw@Diamonds\empty%
1582 \let\ODw@Hearts\empty%
1583 \let\ODw@Spades\empty%
1584 %
1585 % We need some data which is calculated in TableII
1586 % to create TableI and TableIII. So we put TableII in a
1587 % box and display it later at due time
1588 %
1589 \def\ODw@EXtra{0.9em}% white space in title
1590 %
1591 \provideenvironment{TableII}{%

```

```

1592 \begin{tabular}[b]{FPPP}%
1593 \multicolumn{1}{c}{\translate{Lead!-(ODw)}} & %
1594 \cci{\translate{2nd-(ODw)}} & %
1595 \cci{\translate{3rd-(ODw)}} & %
1596 \cci{\translate{4th-(ODw)}} \\\[0.3em]\hline%
1597 \multicolumn{4}{c}{\}\[-\ODw@EXtra]%
1598 }{%
1599 \end{tabular}%
1600 }% TableII
1601 %
1602 \begin{lrbox}{0}% save TableII for later
1603 \begin{TableII}
1604 }{%
1605 \end{TableII}
1606 \end{lrbox}%
1607 % Check consistency of the played cards
1608 \ODw@ChkSameCards{\ODw@Spades}{\Sp}%
1609 \ODw@ChkSameCards{\ODw@Hearts}{\He}%
1610 \ODw@ChkSameCards{\ODw@Diamonds}{\Di}%
1611 \ODw@ChkSameCards{\ODw@Clubs}{\Cl}%
1612 %
1613 \setcounter{ODw@Cnt}{0}%
1614 \IfBooleanTF#1{\begin{center}}{}%
1615 %

```

We need TableI to show the running trick number. When constructing Table II, the total number of tricks that were actually displayed is available in counter `ODw@Nr`. So we just need to loop `\theODw@Nr` times and write the local counter value `\theODw@Cnt`. To avoid *the extra line problem* we use the solution from:

<https://tex.stackexchange.com/questions/50296/problem-with-using-loop-inside-the-tabular-environment/142562#142562>

```

1616 %
1617 \begin{tabular}[b]{r}% TableI
1618 \cci{\translate{Nr-(ODw)}}\\\[0.3em]
1619 \hline\[-\ODw@EXtra]%
1620 \setcounter{ODw@Cnt}{1}%
1621 \whiledo{\theODw@Cnt<\theODw@Nr}{%
1622 \theODw@Cnt\
1623 \stepcounter{ODw@Cnt}%
1624 }%
1625 \theODw@Cnt\% MUST be outside the loop
1626 % (the extra line problem)!

```

```

1627 \end{tabular}%
1628 %
1629 % TableII showing the cards played in the tricks
1630 %
1631 \usebox{0}%
1632 %

We use TableIII to show the winning trick counts. These are already
stored in a CSV-list \ODw@Scratch, implemented as a token register. To
process this list we use \docsvlist and must only define our \do. As
this table has 2 columns, we check with ODw@Nr that after an item is
read, we put an & and after the next item a \\.

1633 %
1634 \setcounter{ODw@Nr}{0}%
1635 \renewcommand*\do}[1]{%
1636   \ifnumequal{\value{ODw@Nr}}{2}{\setcounter{ODw@Nr}{0}}{%
1637   \stepcounter{ODw@Nr}%
1638   ##1
1639   \ifnumequal{\value{ODw@Nr}}{2}{&}}%
1640 }%
1641 \begin{tabular}[b]{|cc}% TableIII
1642   \multicolumn{1}{|c}%
1643     \cci{\scriptsize\shortstack[c]{\North*!\South*!}} &%
1644   \multicolumn{1}{c}%
1645     \cci{\scriptsize\shortstack[c]{\East*!\West*!}}%
1646   }\\ \hline \[-\ODw@Extra]%
1647   \expandafter\docsvlist\expandafter{\the\ODw@Scratch}%
1648 \end{tabular}%
1649 \IfBooleanTF#1{\end{center}}{}%
1650 %
1651 }% play

```

6.5 Card Diagrams with Bidding

\ODw@CondNewLine

\ODw@CondNewLine[*offset*]

```
1652 \NewDocumentCommand\ODw@CondNewLine{0{Oem}}{%
```

\ODw@CondNewLine forces a newline if the bidding diagram does not fit on the line, taking into account the width of the card diagram and the width of the bidding diagram. Otherwise the bidding diagram appears to the right of the card diagram at distance \ODw@Skip@Width. We call the global macro \ODw@Diagram@Width that contains the width of the

card diagram. The optional parameter of `\ODw@CondNewLine` is used to add some extra offset if needed.

```

1653 {\ODw@GameFont% needed to relate skips to the font-size
1654 % : \the\ODw@Skip@Width:% JW XXX
1655 \setlength{\ODw@Tmp@Len}{\ODw@Bid@Width}%
1656 \addtolength{\ODw@Tmp@Len}{\ODw@Diagram@Width}%
1657 \addtolength{\ODw@Tmp@Len}{#1}%
1658 \addtolength{\ODw@Tmp@Len}{\ODw@Skip@Width}%
1659 \ifthenelse{\lengthtest{\ODw@Tmp@Len > \textwidth}}{%
1660 \ [1em]}{%
1661 \hspace{\ODw@Skip@Width}%
1662 }%
1663 }%
1664 }% ODw@CondNewLine

```

6.6 The Expert Quiz

`\expertquiz`

```
\expertquiz *! [comment]{award}
```

The macro `\expertquiz` displays a hand, a bidding diagram and the award for the answers. Optionally a description can be added. The hand and the bidding have to be defined before. This is done to avoid having 4 more arguments, needed for specifying the hand. The token `'*` centers the whole and the token `'!` forces that the bidding diagram appears on a new line and that the hand shifts a bit to the right. The last parameter defines the award. In order to limit the width of the award we use the known widths of the bidding diagram and the hand and set the parbox accordingly to display the award.

```

1665 \NewDocumentCommand\expertquiz{st! O{}m}{%
1666 %
1667 %
1668 %
1669 %
1670 %
1671 %
1672 %
1673 %

```

```

\expertquiz* !comment{award}
1 2 3 4

```

```

1667 \noindent%
1668 \IfBooleanTF#1{\begin{center}}% "*" detected
1669 {\par\vspace{0.5\baselineskip}}%
1670 \bgroup% keep font changes local (e.g. "\smaller").
1671 \ODw@LegendFont%
1672 \ifx#3\empty\else#3\par\fi%
1673 \egroup%

```

```

1674 \IfBooleanTF{#2}{~\hspace*{2em}}{}% !" detected
1675 \usebox{\ODw@Hand@Box}% display the saved hand
1676 \IfBooleanTF{#2}{}{\quad}% no !" detected
1677 \setlength\ODw@Tmp@Width{\wd\ODw@BidBox + 1em}%
1678 \IfBooleanTF{#2}%
1679 {\}%
1680 {\addtolength\ODw@Tmp@Width{\wd\ODw@Hand@Box}}%
1681 \usebox{\ODw@BidBox}% display the saved bidding
1682 \par\vspace{0.3em}%\noindent%
1683 {% keep legendfont and "smaller" local
1684 \ODw@LegendFont%
1685 \smaller%
1686 \IfBooleanTF#1{\bgroup\centering}{}%
1687 \parbox[t]{\ODw@Tmp@Width}{%
1688 \textbf{\ODw@AwardText: }%
1689 \raggedright#4%
1690 }% parbox
1691 \IfBooleanTF#1{\egroup}{}%
1692 }%
1693 % \fi%
1694 \IfBooleanTF#1{\end{center}}{}%
1695 }% expertquiz

```

6.7 Resetting the Game

We use `pgfkeys` with its `<key>=<val>` system to specify the fonts and other things that we want to have as defaults, rather than the initial **ONEDOWN** values. Therefore we first define the keys and the store for it.

```

1696 \pgfkeys{%
1697 /ODw/.is family, /ODw,
1698 % fonts
1699 bidder/.store in = \ODw@BidderDefault,
1700 compass/.store in = \ODw@CompassDefault,
1701 game/.store in = \ODw@GameDefault,
1702 legend/.store in = \ODw@LegendDefault,
1703 name/.store in = \ODw@NameDefault,
1704 other/.store in = \ODw@OtherDefault,
1705 % compass
1706 compline/.store in = \ODw@CompLine,
1707 compmid/.store in = \ODw@CompMid,
1708 compsize/.store in = \ODw@CompSize,

```

```

1709 }
1710 % compass
1711 \ODw@set{compshow/.is choice}
1712 \ODw@set{compshow/off/.code={\ODw@CompShowfalse}}
1713 \ODw@set{compshow/on/.code={\ODw@CompShowtrue}}
1714 \ODw@set{compturn/.is choice}
1715 \ODw@set{compturn/off/.code={\ODw@CompTurnfalse}}
1716 \ODw@set{compturn/on/.code={\ODw@CompTurntrue}}
1717 % bidding
1718 \ODw@set{bidders/.is choice}
1719 \ODw@set{bidders/off/.code={\ODw@Biddersfalse}}
1720 \ODw@set{bidders/on/.code={\ODw@Bidderstrue}}
1721 \ODw@set{bidfirst/.is choice}
1722 \ODw@set{bidfirst/N/.code={\ODw@FirstBidCol{N}}}
1723 \ODw@set{bidfirst/E/.code={\ODw@FirstBidCol{E}}}
1724 \ODw@set{bidfirst/S/.code={\ODw@FirstBidCol{S}}}
1725 \ODw@set{bidfirst/W/.code={\ODw@FirstBidCol{W}}}
1726 \ODw@set{bidline/.is choice}
1727 \ODw@set{bidline/off/.code={\ODw@BidLinefalse}}
1728 \ODw@set{bidline/on/.code={\ODw@BidLinetrue}}
1729 \ODw@set{bidlong/.is choice}
1730 \ODw@set{bidlong/off/.code={\ODw@LongCallsfalse}}
1731 \ODw@set{bidlong/on/.code={\ODw@LongCallstrue}}
1732 % synonyms
1733 \ODw@set{compshow/1/.code={\pgfkeys{/ODw/compshow=on}}}
1734 \ODw@set{compshow/true/.code={\pgfkeys{/ODw/compshow=on}}}
1735 \ODw@set{compturn/1/.code={\pgfkeys{/ODw/compturn=on}}}
1736 \ODw@set{compturn/true/.code={\pgfkeys{/ODw/compturn=on}}}
1737 \ODw@set{bidline/1/.code={\pgfkeys{/ODw/bidline=on}}}
1738 \ODw@set{bidders/true/.code={\pgfkeys{/ODw/bidders=on}}}
1739 \ODw@set{bidders/1/.code={\pgfkeys{/ODw/bidders=on}}}
1740 \ODw@set{bidline/true/.code={\pgfkeys{/ODw/bidline=on}}}
1741 \ODw@set{bidlong/1/.code={\pgfkeys{/ODw/bidlong=on}}}
1742 \ODw@set{bidlong/true/.code={\pgfkeys{/ODw/bidlong=on}}}
1743 %
1744 \ODw@set{compshow/0/.code={\pgfkeys{/ODw/compshow=off}}}
1745 \ODw@set{compshow/false/.code={\pgfkeys{/ODw/compshow=off}}}
1746 \ODw@set{compturn/0/.code={\pgfkeys{/ODw/compturn=off}}}
1747 \ODw@set{compturn/false/.code={\pgfkeys{/ODw/compturn=off}}}
1748 \ODw@set{bidders/0/.code={\pgfkeys{/ODw/bidders=off}}}
1749 \ODw@set{bidders/false/.code={\pgfkeys{/ODw/bidders=off}}}
1750 \ODw@set{bidline/0/.code={\pgfkeys{/ODw/bidline=off}}}
1751 \ODw@set{bidline/false/.code={\pgfkeys{/ODw/bidline=off}}}

```



```

1752 \ODw@set{bidlong/0/.code={\pgfkeys{/ODw/bidlong=off}}}
1753 \ODw@set{bidlong/false/.code={\pgfkeys{/ODw/bidlong=off}}}

```

\resetfonts

\resetfonts

```

1754 \newcommand\resetfonts{%
1755   \bidderfont{\ODw@BidderDefault}%
1756   \compassfont{\ODw@CompassDefault}%
1757   \gamefont{\ODw@GameDefault}%
1758   \legendfont{\ODw@LegendDefault}%
1759   \namefont{\ODw@NameDefault}%
1760   \otherfont{\ODw@OtherDefault}%
1761 }% resetfonts

```

\setdefaults

\setdefaults *{\langle key1=val1\rangle,\langle key2=val2\rangle,...}

The available keys are those defined in `\pgfkeys` some lines up from here. For the fonts they are: `bidder`, `compass`, `game`, `legend`, `name` and `other`. They store the new default value in the corresponding variable. In order to make the new default active, we must use `\setdefaults*` which will also call `\resetfonts`.

The keys for the compass are: `compline`, `compmid`, `compshow`, `compsize` and `compturn`. They control the thickness of the frame, the mid-text, the visibility of the compass, its size and the angle of the compass E–W letters.

For the bidding diagram we have: `bidders`, `bidfirst`, `bidline` and `bidlong`. They control if bidders are to be displayed at all, which bidder appears in the first column, draw a `\hline` below the header and showing long calls.

Furthermore defined elsewhere are the keys to control errors `err` and warnings `warn`, as well as the key to control the color e.g. of the symbols `colors`. Refer to section [3.1.3](#)

```

1762 \NewDocumentCommand\setdefaults{s m}{%
1763   \pgfkeys{/ODw,#2}%
1764   \IfBooleanTF{#1}{\resetfonts}{}%
1765 }% setdefaults
1766 %

```

\newgame

\newgame

`\newgame` resets and clears the stored game information to be ready for a new game. We do not reset the option for warn- and err-messages,

nor any selected font. Setting `\boardnr{0}` executes also:
`\ODw@BoardText{}\vulner[-1]\dealer[-1]}`.

```
1767 \newcommand\newgame{%
1768   \boardnr{0}%
1769   \headlinetext{}%
1770   \footlinetext{}%
1771 % clear the left/right upper/lower stuff
1772   \gdef\ODw@LeftUpperText{}%
1773   \gdef\ODw@LeftLowerText{}%
1774   \gdef\ODw@RightUpperText{}%
1775   \gdef\ODw@RightLowerText{}%
1776 % clear the hands
1777   \gdef\ODw@Nhand{\ODw@hand{t}{-}{-}{-}}%
1778   \gdef\ODw@Ehand{\ODw@hand{c}{-}{-}{-}}%
1779   \gdef\ODw@Shand{\ODw@hand{b}{-}{-}{-}}%
1780   \gdef\ODw@Whand{\ODw@hand{c}{-}{-}{-}}%
1781 %
1782 % set default for real bidders names: no names
1783 % we print only the symbolic names North, East, etc.
1784 %
1785   \namesNS{}{}\namesEW{}{}%
1786 %
1787 % reset consistency check stuff
1788 %
1789   \gdef\ODw@Spades{}%
1790   \gdef\ODw@Hearts{}%
1791   \gdef\ODw@Diamonds{}%
1792   \gdef\ODw@Clubs{}%
1793 %
1794   \gdef\ODw@NSpades{}\gdef\ODw@ESpades{}%
1795   \gdef\ODw@SSpades{}\gdef\ODw@WSpades{}%
1796   \gdef\ODw@NHearts{}\gdef\ODw@EHearts{}%
1797   \gdef\ODw@SHearts{}\gdef\ODw@WHearts{}%
1798   \gdef\ODw@NDiamonds{}\gdef\ODw@EDiamonds{}%
1799   \gdef\ODw@SDiamonds{}\gdef\ODw@WDiamonds{}%
1800   \gdef\ODw@NClubs{}\gdef\ODw@EClubs{}%
1801   \gdef\ODw@SClubs{}\gdef\ODw@WClubs{}%
1802 %
1803 }% newgame
```

6.8 Error Handling

6.8.1 Consistency Checks

We perform different checks on consistency of the cards entered:

1. Check that a hand not has more than 13 cards (E)
2. Check that a hand doesn't contain multiple cards (E)
3. Check that a deal doesn't contain multiple cards (E)
4. Check that a hand has less than 13 cards (W)
5. check that a suit of a deal has more than 13 cards (E)
6. check that a suit of a deal has less than 13 cards (W)
7. In play diagrams: check that a card is played only once (E)

The checks marked with (E) raise an error, those marked with (W) raise a warning. They can be controlled with the package options `err` and `warn`.

`\ODw@ChkNrOfCards`

```
\ODw@ChkNrOfCards{<cards>}{<hand>}
```

```
1804 \newcommand\ODw@ChkNrOfCards[2]{%
```

```
\ODw@ChkNrOfCards{cards}{hand}
```

```
1 2
```

```
#1 = a string with all cards of all suits of the hand denoted by #2
```

```
step 1: remove all "-" (that denotes an empty suit)
```

```
step 2: warn if StrLen < 13 ; Err if StrLen > 13
```

```
1805 \StrDel{#1}{-}[\ODw@CardStr]% remove voids
```

```
1806 \StrLen{\ODw@CardStr}[\ODw@CardLen]%
```

```
1807 \ifthenelse{\ODw@CardLen > 13}{%
```

```
1808 \ODw@Error{#2 has \ODw@CardLen{} cards}%
```

```
1809 }{%
```

```
1810 \ifthenelse{\ODw@CardLen < 13}{%
```

```
1811 \ODw@Warning{#2 has \ODw@CardLen{} cards}%
```

```
1812 }{}%
```

```
1813 }%
```

```
1814 }% \ODw@ChkNrOfCards
```

`\ODw@PrErr`

```
\ODw@PrErr{<rank>}{<count>}{<suit>}
```

```
1815 \newcommand\ODw@PrErr[3]{%
```

```
\ODw@PrErr{rank}{count}{suit}
```

```
1 2 3
```

This macro only outputs the warning/error if the card specified by

rank (#1) and suit (#3) does not occur (denoted by #2) exactly $1 \times$. An exception for spotcards must not be made, because in `\ODw@ChkSameCards` they are not taken into account. (In fact they are already filtered out by `ODw@translate`).

```

1816 \bgroup%
1817   \if#1T\def\ODw@T{10}\else\def\ODw@T{#1}\fi%
1818   \ifthenelse{#2 > 1}{%
1819     \ODw@Error{Card #3\,\ODw@T{ } occurs #2 times}%
1820   }{%
1821     \ifthenelse{#2 = 0}{%
1822       \ODw@Warning{Card #3\,\ODw@T{ } fails}}{%
1823     }%
1824 \egroup%
1825 }% ODw@PrErr

```

`\ODw@ChkSameCards`

```
\ODw@ChkSameCards{<cards>}{<suit>}
```

```

1826 \newcommand\ODw@ChkSameCards [2] {%
#1 = a string with all cards of 1 suit (denoted by #2) of all hands
step 1: remove all "-"
step 2: we count the frequency of all ranks 2--9,T,J,Q,K,A (ODw@CCnt)
step 3: Warn if Freq(card) = 0 ; Err if Freq(card) > 1 (ODw@PrErr)

```

```

1827 \StrDel{#1}{-}[\ODw@CardStr]%
1828 \StrCount{\ODw@CardStr}{2}[\ODw@CCnt] \ODw@PrErr{2}{\ODw@CCnt}{#2}%
1829 \StrCount{\ODw@CardStr}{3}[\ODw@CCnt] \ODw@PrErr{3}{\ODw@CCnt}{#2}%
1830 \StrCount{\ODw@CardStr}{4}[\ODw@CCnt] \ODw@PrErr{4}{\ODw@CCnt}{#2}%
1831 \StrCount{\ODw@CardStr}{5}[\ODw@CCnt] \ODw@PrErr{5}{\ODw@CCnt}{#2}%
1832 \StrCount{\ODw@CardStr}{6}[\ODw@CCnt] \ODw@PrErr{6}{\ODw@CCnt}{#2}%
1833 \StrCount{\ODw@CardStr}{7}[\ODw@CCnt] \ODw@PrErr{7}{\ODw@CCnt}{#2}%
1834 \StrCount{\ODw@CardStr}{8}[\ODw@CCnt] \ODw@PrErr{8}{\ODw@CCnt}{#2}%
1835 \StrCount{\ODw@CardStr}{9}[\ODw@CCnt] \ODw@PrErr{9}{\ODw@CCnt}{#2}%
1836 \StrCount{\ODw@CardStr}{T}[\ODw@CCnt] \ODw@PrErr{T}{\ODw@CCnt}{#2}%
1837 \StrCount{\ODw@CardStr}{J}[\ODw@CCnt] \ODw@PrErr{J}{\ODw@CCnt}{#2}%
1838 \StrCount{\ODw@CardStr}{Q}[\ODw@CCnt] \ODw@PrErr{Q}{\ODw@CCnt}{#2}%
1839 \StrCount{\ODw@CardStr}{K}[\ODw@CCnt] \ODw@PrErr{K}{\ODw@CCnt}{#2}%
1840 \StrCount{\ODw@CardStr}{A}[\ODw@CCnt] \ODw@PrErr{A}{\ODw@CCnt}{#2}%
1841 }% ODw@ChkSameCards

```

6.8.2 Controlling Messages

```
1842 \newbool{ODw@Warnings}
```

```

1843 \newbool{ODw@Errors}
1844 %
1845 \ODw@set{warn/off/.code={%
1846   \global\setbool{ODw@Warnings}{false}}}
1847 \ODw@set{warn/on/.code={%
1848   \global\setbool{ODw@Warnings}{true}}}
1849 \ODw@set{err/off/.code={%
1850   \global\setbool{ODw@Errors}{false}}}
1851 \ODw@set{err/on/.code={%
1852   \global\setbool{ODw@Errors}{true}}}
1853 \ODw@set{warn=off}
1854 \ODw@set{err=on}
1855
1856 \ProcessPgfOptions{/ODw}

```

\ODw@Error

```

1857 \newcommand\ODw@Error[1]{%
1858   \ifbool{ODw@Errors}{%
1859     \par\textcolor{red}{Error: #1}\par}{}%
1860 }% \ODw@Error

```

\ODw@Warning

```

1861 \newcommand\ODw@Warning[1]{%
1862   \ifbool{ODw@Warnings}{%
1863     \par\textcolor{blue}{Warning: #1}\par}{}%
1864 }% \ODw@Warning

```

6.9 Misc Bridge Terms

6.9.1 Honour Cards

These macros retrieve the translations of the 4 alternative forms of the honour cards from the ODw-dictionary of the active language.

\Ace

```

\ace 1865 \NewDocumentCommand{\Ace}{s t!}{%
1866   \bgroup%
1867   %JW   \ODw@GameFont%
1868   \IfBooleanTF{#1}{%
1869     \IfBooleanTF{#2}{%
1870       {\translate{A-(ODw)}}}%
1871       {\translate{Ace-(ODw)}}}%
1872   }{%
1873     \IfBooleanTF{#2}%

```

```

1874         {\translate{a-(ODw)}}}%
1875         {\translate{ace-(ODw)}}}%
1876     }%
1877     \egroup%
1878     \xspace%
1879 }% Ace
1880 %
1881 \def\ace{\Ace*!}

\King
\king 1882 \NewDocumentCommand{\King}{s t!}{%
1883     \bgroup%
1884     %JW     \ODw@GameFont%
1885         \IfBooleanTF{#1}{%
1886             \IfBooleanTF{#2}{%
1887                 {\translate{K-(ODw)}}}%
1888                 {\translate{King-(ODw)}}}%
1889             }{%
1890                 \IfBooleanTF{#2}{%
1891                     {\translate{k-(ODw)}}}%
1892                     {\translate{king-(ODw)}}}%
1893             }%
1894     \egroup%
1895     \xspace%
1896 }% King
1897 %
1898 \def\king{\King*!}

\Queen
\queen 1899 \NewDocumentCommand{\Queen}{s t!}{%
1900     \bgroup%
1901     %JW     \ODw@GameFont%
1902         \IfBooleanTF{#1}{%
1903             \IfBooleanTF{#2}{%
1904                 {\translate{Q-(ODw)}}}%
1905                 {\translate{Queen-(ODw)}}}%
1906             }{%
1907                 \IfBooleanTF{#2}{%
1908                     {\translate{q-(ODw)}}}%
1909                     {\translate{queen-(ODw)}}}%
1910             }%
1911     \egroup%
1912     \xspace%

```

```

1913 }% Queen
1914 %
1915 \def\queen{\Queen*!}

\Jack
\jack 1916 \NewDocumentCommand{\Jack}{s t!}{%
1917   \bgroup%
1918   %JW   \ODw@GameFont%
1919     \IfBooleanTF{#1}{%
1920       \IfBooleanTF{#2}{%
1921         {\translate{J-(ODw)}}}%
1922         {\translate{Jack-(ODw)}}}%
1923       }{%
1924         \IfBooleanTF{#2}{%
1925           {\translate{j-(ODw)}}}%
1926           {\translate{jack-(ODw)}}}%
1927       }%
1928     \egroup%
1929     \xspace%
1930 }% Jack
1931 %
1932 \def\jack{\Jack*!}

\Ten
1933 \newcommand{\Ten}{%
1934   \bgroup%
1935   %JW   \ODw@GameFont%
1936     10%
1937   \egroup%
1938   \xspace%
1939 }% Ten

```

6.9.2 Vulnerability

These macros retrieve the translations of the 4 alternative forms of the commands `\All` and `\None` from the `ODw`-dictionary of the active language. As there is no short form for them in the English language, we just code these entries in the `ODw`-dictionaries with an exclamation mark '!'.

```

\All
\all 1940 \NewDocumentCommand{\All}{s t!}{%
1941   \bgroup%

```

```

1942     \ODw@OtherFont%
1943     \IfBooleanTF{#1}{%
1944         \IfBooleanTF{#2}%
1945             {\translate{All!-(ODw)}}}%
1946             {\translate{All-(ODw)}}}%
1947     }{%
1948         \IfBooleanTF{#2}%
1949             {\translate{all!-(ODw)}}}%
1950             {\translate{all-(ODw)}}}%
1951     }%
1952 \egroup%
1953 \xspace%
1954 }% All
1955 %
1956 \def\all{\All*}

\None
\none 1957 \NewDocumentCommand{\None}{s t!}{%
1958     \bgroup%
1959         \ODw@OtherFont%
1960         \IfBooleanTF{#1}{%
1961             \IfBooleanTF{#2}%
1962                 {\translate{None!-(ODw)}}}%
1963                 {\translate{None-(ODw)}}}%
1964         }{%
1965             \IfBooleanTF{#2}%
1966                 {\translate{none!-(ODw)}}}%
1967                 {\translate{none-(ODw)}}}%
1968         }%
1969     \egroup%
1970     \xspace%
1971 }% None
1972 %
1973 \def\none{\None*}
1974 %

```

6.9.3 Diagram Annotations

These macros retrieve the translations of the 4 alternative forms of the commands `\Contract`, `\Lead`, `\Declarer`, `\Board` and `\Deal` from the `ODw`-dictionary of the active language.

```

\Contract
\contract

```



```

1975 \NewDocumentCommand{\Contract}{s t!}{%
1976   \bgroup%
1977     \ODw@OtherFont%
1978     \IfBooleanTF{#1}{%
1979       \IfBooleanTF{#2}{%
1980         {\translate{Contr-(ODw)}}}%
1981         {\translate{Contract-(ODw)}}}%
1982     }{%
1983       \IfBooleanTF{#2}{%
1984         {\translate{contr-(ODw)}}}%
1985         {\translate{contract-(ODw)}}}%
1986     }%
1987   \egroup%
1988   \xspace%
1989 }% Contract
1990 %
1991 \def\contract{\Contract*}
1992 %

\Lead
\lead 1993 \NewDocumentCommand{\Lead}{s t!}{%
1994   \bgroup%
1995     \ODw@OtherFont%
1996     \IfBooleanTF{#1}{%
1997       \IfBooleanTF{#2}{%
1998         {\translate{Lead!-(ODw)}}}%
1999         {\translate{Lead-(ODw)}}}%
2000     }{% else #1
2001       \IfBooleanTF{#2}{%
2002         {\translate{lead!-(ODw)}}}%
2003         {\translate{lead-(ODw)}}}%
2004     }% #1
2005   \egroup%
2006   \xspace%
2007 }% Lead
2008 %
2009 \def\lead{\Lead*}
2010 %

\Declarer
\declarer 2011 \NewDocumentCommand{\Declarer}{s t!}{%
2012   \bgroup%
2013     \ODw@OtherFont%
2014     \IfBooleanTF{#1}{%

```

```

2015     \IfBooleanTF{#2}%
2016         {\translate{Decl-(ODw)}}}%
2017         {\translate{Declarer-(ODw)}}}%
2018     }{%
2019     \IfBooleanTF{#2}%
2020         {\translate{decl-(ODw)}}}%
2021         {\translate{declarer-(ODw)}}}%
2022     }%
2023     \egroup%
2024     \xspace%
2025 }% Declarer
2026 %
2027 \def\declarer{\Declarer*}
2028 %

\by
2029 \newcommand\by{%
2030     \bgroup%
2031     \ODw@OtherFont%
2032     \translate{by-(ODw)}%
2033     \egroup%
2034     \xspace%
2035 }% by

\Board
\board 2036 \NewDocumentCommand{\Board}{s t!}{%
2037     \bgroup%
2038     \ODw@OtherFont%
2039     \IfBooleanTF{#1}{%
2040         \IfBooleanTF{#2}%
2041             {\translate{Brd-(ODw)}}}%
2042             {\translate{Board-(ODw)}}}%
2043     }{%
2044         \IfBooleanTF{#2}%
2045             {\translate{brd-(ODw)}}}%
2046             {\translate{board-(ODw)}}}%
2047     }%
2048     \egroup%
2049     \xspace%
2050 }% Board
2051 %
2052 \def\board{\Board*}
2053 %

```

```

\Deal
\deal 2054 \NewDocumentCommand{\Deal}{s t!}{%
2055   \bgroup%
2056     \ODw@OtherFont%
2057     \IfBooleanTF{#1}{%
2058       \IfBooleanTF{#2}{%
2059         {\translate{Deal!-(ODw)}}}%
2060         {\translate{Deal-(ODw)}}}%
2061     }{%
2062     \IfBooleanTF{#2}{%
2063       {\translate{deal!-(ODw)}}}%
2064       {\translate{deal-(ODw)}}}%
2065     }%
2066   \egroup%
2067   \xspace%
2068 }% Deal
2069 %
2070 \def\deal{\Deal*}
2071 %

```

\doubled The commands \doubled and \redoubled do not have a short form.

```

2072 \NewDocumentCommand{\doubled}{s}{%
2073   \bgroup%
2074     \ODw@OtherFont%
2075     \IfBooleanTF{#1}{%
2076       \translate{Doubled-(ODw)}}{%
2077       \translate{doubled-(ODw)}}%
2078     }%
2079   \egroup%
2080   \xspace%
2081 }% doubled

```

\redoubled

```

2082 \NewDocumentCommand{\redoubled}{s}{%
2083   \bgroup%
2084     \ODw@OtherFont%
2085     \IfBooleanTF{#1}{%
2086       \translate{Redoubled-(ODw)}}{%
2087       \translate{redoubled-(ODw)}}%
2088     }%
2089   \egroup%
2090   \xspace%
2091 }% redoubled

```

6.9.4 Point Units

These macros retrieve the translations of the commands `\hpts`, `\lpts`, `\dpts` and `\tpts` from the `ODw`-dictionary of the active language.

```
\hpts
\HCP 2092 \NewDocumentCommand{\hpts}{s t!}{%
2093   \bgroup%
2094     \ODw@OtherFont%
2095     \IfBooleanTF{#1}{%
2096       \IfBooleanTF{#2}{%
2097         {\translate{HCP-(ODw)}}%
2098         {\translate{High Card Points-(ODw)}}%
2099       }{%
2100         \IfBooleanTF{#2}{%
2101           {\translate{hcp-(ODw)}}%
2102           {\translate{high card points-(ODw)}}%
2103         }%
2104       \egroup%
2105     \xspace%
2106 }% High Card Points
2107 %
2108 \def\HCP{\,\hpts*!}

\lpts
\LP 2109 \NewDocumentCommand{\lpts}{s t!}{%
\HLP 2110   \bgroup%
2111     \ODw@OtherFont%
2112     \IfBooleanTF{#1}{%
2113       \IfBooleanTF{#2}{%
2114         {\translate{LP-(ODw)}}%
2115         {\translate{Length Points-(ODw)}}%
2116       }{%
2117         \IfBooleanTF{#2}{%
2118           {\translate{lp-(ODw)}}%
2119           {\translate{length points-(ODw)}}%
2120         }%
2121       \egroup%
2122     \xspace%
2123 }% Length Points
2124 %
2125 \def\LP{\,\lpts*!}
2126 \def\HLP{\,\bgroup\ODw@OtherFont\translate{HLP-(ODw)}\egroup\xspace}
```

```

\dpts
\DP 2127 \NewDocumentCommand{\dpts}{s t!}{%
2128   \bgroup%
2129   \ODw@OtherFont%
2130   \IfBooleanTF{#1}{%
2131     \IfBooleanTF{#2}{%
2132       {\translate{DP-(ODw)}}}%
2133       {\translate{Distribution Points-(ODw)}}}%
2134     }{%
2135     \IfBooleanTF{#2}{%
2136       {\translate{dp-(ODw)}}}%
2137       {\translate{distribution points-(ODw)}}}%
2138     }%
2139   \egroup%
2140   \xspace%
2141 }% Distribution Points
2142 %
2143 \def\DP{\,\dpts*!}

```

```

\tpts
\TP 2144 \NewDocumentCommand{\tpts}{s t!}{%
2145   \bgroup%
2146   \ODw@OtherFont%
2147   \IfBooleanTF{#1}{%
2148     \IfBooleanTF{#2}{%
2149       {\translate{TP-(ODw)}}}%
2150       {\translate{Total Points-(ODw)}}}%
2151     }{%
2152     \IfBooleanTF{#2}{%
2153       {\translate{tp-(ODw)}}}%
2154       {\translate{total points-(ODw)}}}%
2155     }%
2156   \egroup%
2157   \xspace%
2158 }% Total Points
2159 %
2160 \def\TP{\,\tpts*!}

```

6.9.5 Forcings

These macros retrieve the translations of the commands `\gforce`, `\sforce`, `\nmforce`, `\tsforce` and `\fsforce` from the `ODw`-dictionary of the active language.

\gforce

```
\GF 2161 \NewDocumentCommand{\gforce}{s t!}{%
2162   \bgroup%
2163   \ODw@OtherFont%
2164   \IfBooleanTF{#1}{%
2165     \IfBooleanTF{#2}{%
2166       {\translate{GF-(ODw)}}%
2167       {\translate{Game Forcing-(ODw)}}%
2168     }{%
2169       \IfBooleanTF{#2}{%
2170         {\translate{gf-(ODw)}}%
2171         {\translate{game forcing-(ODw)}}%
2172       }%
2173   \egroup%
2174   \xspace%
2175 }% Game Forcing
2176 %
2177 \def\GF{\gforce*!}
```

\sforce

```
\SF 2178 \NewDocumentCommand{\sforce}{s t!}{%
2179   \bgroup%
2180   \ODw@OtherFont%
2181   \IfBooleanTF{#1}{%
2182     \IfBooleanTF{#2}{%
2183       {\translate{SF-(ODw)}}%
2184       {\translate{Semi Forcing-(ODw)}}%
2185     }{%
2186       \IfBooleanTF{#2}{%
2187         {\translate{sf-(ODw)}}%
2188         {\translate{semi forcing-(ODw)}}%
2189       }%
2190   \egroup%
2191   \xspace%
2192 }% Semi Forcing
2193 %
2194 \def\SF{\sforce*!}
```

\nmforce

```
\NMF 2195 \NewDocumentCommand{\nmforce}{s t!}{%
2196   \bgroup%
2197   \ODw@OtherFont%
2198   \IfBooleanTF{#1}{%
```

```

2199     \IfBooleanTF{#2}%
2200         {\translate{NMF-(ODw)}}}%
2201         {\translate{New Minor Forcing-(ODw)}}}%
2202     }{%
2203     \IfBooleanTF{#2}%
2204         {\translate{nmf-(ODw)}}}%
2205         {\translate{new minor forcing-(ODw)}}}%
2206     }%
2207 \egroup%
2208 \xspace%
2209 }% New Minor Forcing
2210 %
2211 \def\NMF{\nmforce*!}

\tsforce
\TSF 2212 \NewDocumentCommand{\tsforce}{s t!}{%
2213     \bgroup%
2214     \ODw@OtherFont%
2215     \IfBooleanTF{#1}{%
2216         \IfBooleanTF{#2}%
2217             {\translate{TSF-(ODw)}}}%
2218             {\translate{Third Suit Forcing-(ODw)}}}%
2219     }{%
2220     \IfBooleanTF{#2}%
2221         {\translate{tsf-(ODw)}}}%
2222         {\translate{third suit forcing-(ODw)}}}%
2223     }%
2224 \egroup%
2225 \xspace%
2226 }% Third Suit Forcing
2227 %
2228 \def\TSF{\tsforce*!}

\fsforce
\FSF 2229 \NewDocumentCommand{\fsforce}{s t!}{%
2230     \bgroup%
2231     \ODw@OtherFont%
2232     \IfBooleanTF{#1}{%
2233         \IfBooleanTF{#2}%
2234             {\translate{FSF-(ODw)}}}%
2235             {\translate{Fourth Suit Forcing-(ODw)}}}%
2236     }{%
2237     \IfBooleanTF{#2}%
2238         {\translate{fsf-(ODw)}}}%

```

```

2239     {\translate{fourth suit forcing-(ODw)}}}%
2240   }%
2241   \egroup%
2242   \xspace%
2243 }% Fourth Suit Forcing
2244 %
2245 \def\FSF{\fsforce*!}

```

6.10 Initialization

It's time to prepare everything. We clear the game and set the defaults.

```
2246 \newgame
```

Set the default fonts

```

2247 \setdefaults{bidder=\mdseries\sffamily}
2248 \setdefaults{compass=\mdseries\sffamily}
2249 \setdefaults{game=\bfseries\sffamily}
2250 \setdefaults{legend=\mdseries\rmfamily}
2251 \setdefaults{name=\mdseries\slshape}
2252 \setdefaults*{other=\bfseries\sffamily}

```

Set default coloring to black and red

```
2253 \setdefaults{colors=b+r}
```

Set the compass

```
2254 \setdefaults{compshow=on,compturn=off}
```

set the start column for bidding (West is recommended) and the long form.

```
2255 \setdefaults{bidfirst=W,bidders=on,bidlong=on}
```

Now we load the dictionaries for the languages that are to be used in the document. We use `tracklang` to iterate over all the document languages and load the corresponding ODw-dictionaries.

```

2256 %\AtBeginDocument{%
2257 \usedictionary{ODw}
2258 %%%
2259 \ForEachTrackedLanguage{\thislang}{%
2260   \PackageInfo{ODw}{Language \thislang\ detected}%
2261 %
2262 % Commented out due to translator
2263 %   \ifthenelse{\equal{\thislang}{norsk}}{%
2264 %     {\def\thislang{norwegian}}{}}%
2265 %

```



```

2266 \IfEqCase{\thislang}{%
2267   {danish}{\IfFileExists{ODw-Danish.dict}%
2268             {\uselanguage{Danish}%
2269               \PackageInfo{ODw}{Dictionary ODw-Danish.dict loaded}
2270               }{\PackageWarning{ODw}{Dictionary ODw-Danish.dict missing}}}%
2271 %
2272   {dutch}{\IfFileExists{ODw-Dutch.dict}%
2273            {\uselanguage{Dutch}%
2274              \PackageInfo{ODw}{Dictionary ODw-Dutch.dict loaded}
2275              }{\PackageWarning{ODw}{Dictionary ODw-Dutch.dict missing}}}%
2276 %
2277   {english}{\IfFileExists{ODw-English.dict}%
2278             {\uselanguage{English}%
2279               \PackageInfo{ODw}{Dictionary ODw-English.dict loaded}
2280               }{\PackageWarning{ODw}{Dictionary ODw-English.dict missing}}}%
2281 %
2282   {french}{\IfFileExists{ODw-French.dict}%
2283            {\uselanguage{French}%
2284              \PackageInfo{ODw}{Dictionary ODw-French.dict loaded}
2285              }{\PackageWarning{ODw}{Dictionary ODw-French.dict missing}}}%
2286 %
2287   {german}{\IfFileExists{ODw-German.dict}%
2288            {\uselanguage{German}%
2289              \PackageInfo{ODw}{Dictionary ODw-German.dict loaded}
2290              }{\PackageWarning{ODw}{Dictionary ODw-German.dict missing}}}%
2291 %
2292   {norsk}{\IfFileExists{ODw-Norsk.dict}%
2293           {\uselanguage{Norsk}%
2294             \PackageInfo{ODw}{Dictionary ODw-Norsk.dict loaded}
2295             }{\PackageWarning{ODw}{Dictionary ODw-Norsk.dict missing}}}%
2296 %
2297   {spanish}{\IfFileExists{ODw-Spanish.dict}%
2298            {\uselanguage{Spanish}%
2299              \PackageInfo{ODw}{Dictionary ODw-Spanish.dict loaded}
2300              }{\PackageWarning{ODw}{Dictionary ODw-Spanish.dict missing}}}%
2301 %
2302   {swedish}{\IfFileExists{ODw-Swedish.dict}%
2303            {\uselanguage{Swedish}%
2304              \PackageInfo{ODw}{Dictionary ODw-Swedish.dict loaded}
2305              }{\PackageWarning{ODw}{Dictionary ODw-Swedish.dict missing}}}%
2306 %
2307   {turkish}{\IfFileExists{ODw-Turkish.dict}%
2308            {\uselanguage{Turkish}%

```

```

2309             \PackageInfo{ODw}{Dictionary ODw-Turkish.dict loaded}
2310             }{\PackageWarning{ODw}{Dictionary ODw-Turkish.dict missing}}}%
2311 %
2312     }[%
2313         \PackageWarning{ODw}%
2314         {Language \thislang\space not supported by OneDown}
2315     ]% IfEqCase
2316 %
2317 }% ForEach
2318 %%%
2319 }% AtBeginDocument
2320 % Look if we can find a file called onedown.symbols somewhere in the \LaTeX-path
2321 % If so, we load it (the code for inputting card symbols directly from keyboard)
2322 % If not, we keep silent about it
2323 \@input{onedown.symbols}
2324 %
That's it folks, happy TEXing!
2325 \endinput% onedown.sty

```

7 References

- [1] Kees van der Laan: *Typsetting Bridge via T_EX*, TUGboat Vol. 11, No. 2 (1990), p265ff
- [2] Richard Pavlicek: *Bridge Writing Style Guide*, <http://www.rpbridge.net/7z69.htm>

8 Change History

<p>v0.1</p> <p>General:</p> <ul style="list-style-type: none"> - Reorganized the bzd versioning. bzd will contain only onedown, the former bidnplay stuff is archived. The ToDo and Known-Bugs lists are cleaned. - We are version 0.1 now, trying to keep the bzd 	<p>version number equal to the changes minor number. 1</p> <p>v0.2</p> <p>General:</p> <ul style="list-style-type: none"> - We finally have a List of User Commands. - In the compass we can print vulner in red and mark the dealer. We have a hook \CompassMid to
--	--

write something in the middle of the compass.

- Added several macros to auto-translate common stuff like 'lead' etc.
- Added the danish language.
- Corrected a bug in `\dealer` and `\vulner`.
- Removed pgf-key 'lang': we now load the needed languages on the fly.
- Finally we revised the documentation. 1

replaced by neat expl3 code.

- We load necessary dictionaries automatically on the fly and enhanced the colors options.
- Corrected some minor bugs and reorganized the documentation.
- One can now also generate the documentation without the list of user commands 1

v0.3

play:

- In order to avoid empty columns in environment `playtricks` we reorganized it. Rather than just 1 table we use 3 tables. The middle one typesets the relevant tricks, stored in an `lrbox`, while generating on the fly a string with the winning tricks. Finally we put the running trick-number in `TableI`, we 'use' `TableII` and construct `TableIII` from the string with the winning tricks. 91

v0.5

General:

- Associated the names and bidders in a fixed way.
- Added checks to `onesuitNS/EW`.
- Made `ODw@OtherFont` local where necessary.
- Redefined `columntypes`.
- Made 'T' a code for '10'.
- Adapted the 'translate' macro to enable both `1H` and `1\He` etc.
- Enhanced the documentation. 1

v0.4

General:

- Major change in `playtricks`: the winner is now determined by the cards played, and code is added to check consistency.
- Dirty coded macros like `\ODw@symbol` and `\ODw@(@)Card(s)` are

v0.6

General:

- Made all internal names hidden by adding 'ODw@' to it.
- Changed `\ODw@AccTricksN` in `\ODw@AccTricks`.
- Some minor adaptations of the documentation 1

v0.6a

General:

- In order to test which suit (`\Cl,...`) was encountered in

<p><code>\ODw@translate</code> (see page 45) we <i>must</i> define the suits as a <i>renewrobustcommand</i>. So we <code>\define</code> them first. The idea was given on LaTeX StackExchange by egreg, see https://tex.stackexchange.com/questions/420257/test-which-macro-is-called-in-tabular/420258#420258 39</p>	<p>token '-' 1</p>
<p>v0.7</p> <p>General:</p> <ul style="list-style-type: none"> - Final(?) edits of the documentation: Consequent use of 'card diagram', 'bidding' diagram and 'play diagram'. - Corrected some minor bugs. - Changed the name of the 'playtricks' environment into 'play'. - Can now disable all colors in the manual for monochrome printing. - Changed signature of <code>\expertquiz</code> and <code>\boardtext</code>. 1 	<p>v0.8</p> <p>General:</p> <ul style="list-style-type: none"> - Added package <code>verbatimbox</code> to adjust verbatim font easily. - Added <code>\setdefaults</code> with keys for fonts, compass and bidding diagrams. - Made <code>\FirstBidCol</code> internally hidden. - Removed <code>\longcalls</code>, <code>\CompassMid</code>, added <code>\compassfont</code>. - Changed the order of the hands in <code>\onesuitAll</code>. - Renewed the implementation of the compass and added extra features. 1
<p>v0.7a</p> <p>General:</p> <ul style="list-style-type: none"> - Major change: Removed all <code>\bidXX</code> and <code>\bidXXpair</code> commands. - The biddings can now be shown with <code>\showXX+</code> (with token '+'). - The macros <code>\hand-</code> and the bidding environments suppress their output with 	<p>v0.9</p> <p>General:</p> <ul style="list-style-type: none"> - Adapted <code>\ODwset</code> and <code>\setdefaults</code>. - Separated key <i>messages</i> into keys <i>warn</i> and <i>err</i>. - Adapted <code>\ODw@Compass</code>. - Removed legends from <code>\showNS</code>. - Added <code>\sbox1</code> to all <code>\showXX</code> macros with a N-hand. - Changed <code>ODw@[No]Warnings</code> and <code>ODw@[No]Errors</code>. - Corrected some minor bugs and adapted the documentation accordingly. 1
<p>v1.0</p> <p>General:</p> <ul style="list-style-type: none"> - Adapted the urls, the 	

- directories and some filenames to conform to the CTAN-standard and made the bundle ready for upload.
 - Corrected a small bug in `\ODw@Compass` that was introduced in v0.9.
 - Made all relevant text-writing macros in 4 versions with/without tokens `'*` and `'!`.
 - Corrected a sizing/font bug.
 - Added 4 variants of many other text-writing macros, `onesuit-NE/NW/`.
 - Adapted `\handskip`.
 - Added code to work-around a `babel-translations` inconsistency w.r.t. `norsk/norwegian`. 1
- v1.1
- General:
 - Corrected `onedown-ref.tex`
 - . Added the missing rows in lines 419-422.
 - . and the explanatory text on line 457.
 - Changed in all `ODw*.trsl` files: The translation of `'pass'` into `'p'` and `'Pass'` into `'P'`.
 - Corrected some typos in `ShowAll.tex`:
 - . `\contract*` into `\contract`,
 - . `\declarer*` into `\declarer` and
 - . `\lead*` into `\lead`.
 - Removed some illegal
- v1.1a
- chars from changes entries. 1
- v1.2a
- General:
 - Enhanced the documentation (both `onedown.dtx` and `onedown-ref.tex`: The command tokens are no longer shown as `'[*!]`, but as `'*!` with a different background color.
 - Added a thinspace (`'\,`) before all points counts (HCP, LP, DP and TP)
 - Improved `\alert` by using `\textasteriskcentered` rather than just a `'*`
 - The changes history has a better layout now
 - `\showNS` and `\showEW` now can selectively display only the N- or S-hand, resp the E- or W-hand. . . 1
- v1.2b
- General:
 - Corrected a bug: Added `\ODw@monochromefalse` in all initializations of multi colors in `\ODw@set`
 - Made all relevant `setdefaults` settings (like `bidfirst`) non-global
 - Made all `\ODw@BidderX` and `\ODw@NameX` non-global
 - Removed `'\global'` from `\ODw@CompShow` (false and true), `\ODw@CompTurn`, `\ODw@Bidders`, `\ODw@BidLine` and `\ODw@LongCalls`
 - In command `\suit`

	changed 'JW \ODw@GameSize' into '\ODw@GameSize'	1		by 'scalefont' in \gamefont - Added support for the spanish language - Made onedown-ref.tex and onedown-examples.tex multilingual. We can output the onedown commands in all supported languages. - \DeclareSymbolFont {symbols}{OMS} {txsy}{m}{n} interferes with package newtxmath. I renamed <i>symbols</i> into ODw@symbols and <i>symbolsC</i> into ODw@symbolsC to solve this.	1
v1.3a	General: - Added \HLP - Changed/Added in all relevant ODw-lang.trsl: - 'V0.4' into 'V0.4-2018/12/01' - 'FP' into 'F' , 'fp' into 'f' (entry HCP/hcp) - 'Figuren-Punkte' into 'Figurenpunkte' , 'figuren-punkte' into 'figurenpunkte' - 'FL' into 'L' , 'fl' into 'l' - \NewDictTranslation {HLP-(ODw)}{FL} (german)	1	v1.5	General: - Added a check if the etoolbox package is recent enough. - Added code to load onedown.symbols (card symbols) and explained in onedown-examples (CardSymbols) how to use it.	1
v1.3b	General: - Added setdefaults to the list of User Commands - Corrected \setdefaults in onedown-examples: - changed [] into {} - Adapted/corrected onedown-ref and onedown-examples - Added color 'gray' for special effects	1	v1.6	General: - Switched to translator due to incompatibilities with translations - We now use \translate rather than \GetTranslation - Translation files are named like: ODw-German.dict - Adapted the code to read them on the fly	1
v1.4	General: - Made <i>onedown-ref</i> independent of <i>onedown-examples</i> and <i>subfiles</i> - Corrected some mistakes ODw-*.trsl files, all V0.51-2019/10/15 - Added macro \Ten to get rid of the extra enlargement				

9 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	D
<code>\,</code> 33	<code>\Deal</code> 2054
<code>\@input</code> 2323	<code>\deal</code> 2054
<code>_</code> 2260	<code>\dealer</code> 1203 , 1204 , 1205 , 1206 , 1207 , 1208 , 1209 , 1210 , 1211 , 1212 , 1213 , 1214 , 1215 , 1216 , 1217 , 1218 , 1219 , 1302
A	<code>\dealertext</code> 1314
<code>\Ace</code> 282 , 1865	<code>\Declarer</code> 2011
<code>\ace</code> 1865	<code>\declarer</code> 2011
<code>\alert</code> 1323	<code>\Di</code> 117 , 123 , 139 , 152 , 167 , 183 , 199 , 269 , 320 , 662 , 687 , 703 , 807 , 808 , 851 , 897 , 942 , 982 , 1610
<code>\All</code> 1940	<code>\Double</code> 308 , 516
<code>\all</code> 1024 , 1025 , 1026 , 1027 , 1207 , 1210 , 1213 , 1216 , 1940	<code>\double</code> 516
<code>\Allpass</code> 307 , 499	<code>\doubled</code> 2072
<code>\allpass</code> 499	<code>\DP</code> 2127
<code>\announce</code> 1324	<code>\dpts</code> 2127
B	E
<code>\bidderfont</code> 86 , 1755	<code>\East</code> 383 , 439 , 441 , 1043 , 1044 , 1045 , 1046 , 1205 , 1209 , 1213 , 1217 , 1645
<code> bidding (environment)</code> 28 , 1505	<code>\east</code> 383 , 620
<code> biddingpair (environment)</code> 29 , 1521	<code>\easthand</code> 618
<code>\Board</code> 2036	<code>\EastWest</code> 435 , 1032 , 1033 , 1034 , 1035 , 1206 , 1209 , 1212 , 1219
<code>\board</code> 2036	<code>\eastwest</code> 435
<code>\boardnr</code> 1188 , 1768	<code>\endODw@Bidding</code> 1514
<code>\boardtext</code> 1184	<code>\endODw@Biddingpair</code> 1531
<code>\by</code> 2029	environments:
C	<code> bidding</code> 28 , 1505
<code>\Cl</code> 117 , 121 , 137 , 150 , 165 , 181 , 197 , 268 , 319 , 663 , 688 , 704 , 809 , 810 , 852 , 898 , 943 , 983 , 1611	<code> biddingpair</code> 29 , 1521
<code>\compassfont</code> 87 , 1756	<code> ODw@Bidding</code> 1339
<code>\Contract</code> 1975	
<code>\contract</code> 1975	

ODw@Biddingpair	1436	\ifODw@short	58, 1347, 1353, 1359,
play	29, 1569		1365, 1389, 1395, 1401,
\expertquiz	1665		1407, 1444, 1450, 1472, 1478
\explainit	1329		
F		J	
\footlinetext	1225, 1770	\Jack	288, 1916
\FSF	2229	\jack	1916
\fsforce	2229		
G		K	
\gamefont	91, 1757	\King	284, 1882
\GF	2161	\king	1882
\gforce	2161		
H		L	
\hand	658	\LaTeX	2320
\handskip	1279	\Lead	1993
\HCP	2092	\lead	1993
\He	117, 125, 140, 154, 169, 185, 201, 270, 305, 321, 323, 661, 686, 702, 805, 806, 850, 896, 941, 981, 1609	\leftlower	1234
\headlinetext	1223, 1769	\leftupper	1227
\height	1080, 1085, 1089, 1095	\legendfont	89, 1758
\HLP	2109	\LP	2109
\hpts	2092	\lpts	2109
I		M	
\IfNoValueTF	864, 867, 874, 877, 908, 911, 919, 922	\markit	25, 1325
\ifODw@Bidders	61, 1344, 1386, 1441, 1469		
\ifODw@BidLine	62, 1376, 1418, 1459, 1487	N	
\ifOdW@CardSkip	60, 312	\namefont	88, 1759
\ifODw@CompShow	64, 1070	\namesEW	599, 1785
\ifODw@CompTurn	65, 1084, 1096	\namesNS	594, 1785
\ifODw@description	57, 1422, 1491	\newcolumnntype	1333
\ifODw@EmptyFooter	1271	\newgame	1767, 2246
\ifODw@EmptyHeader	1256	\NMF	2195
\ifODw@LongCalls	63, 306, 307, 308, 309, 1316	\nmforce	2195
\ifODw@monochrome	59, 1172	\None	1957
		\none	1020, 1021, 1022, 1023, 1204, 1211, 1214, 1217, 1957
		\North	370, 426, 428, 1039, 1040, 1041, 1042, 1204, 1208, 1212, 1216, 1307, 1643
		\north	370, 606
		\northhand	604
		\NorthSouth	422, 1028, 1029, 1030, 1031, 1205, 1208, 1215, 1218, 1313

\northsouth	422	\ODw@BidLinefalse	1727
\NT	209, 272	\ODw@BidLinetrue	1728
\nt	209	\ODw@BoardText	1183, 1185, 1186, 1190, 1203, 1221
O			
\ODw@AccTricks	359, 1538	\ODw@Box	717, 731, 744, 757, 770, 1176
\ODw@All@Names	597, 602, 1369, 1411, 1454, 1482	\ODw@BTfer	367, 1334
\ODw@append	48, 1540	\ODw@Card@Skip	73, 74, 312, 682, 697
\ODw@AppendCard	282, 284, 286, 288, 291, 293, 294, 295, 296, 297, 298, 299, 300, 331	\ODw@CardLen	1806, 1807, 1808, 1810, 1811
\ODw@appendcard	332, 336, 337, 338, 339	\ODw@Cards	685, 686, 687, 688, 701, 702, 703, 704, 716, 717, 718, 719, 730, 732, 743, 745, 756, 758, 769, 770, 781, 785
\ODw@AwardText	1301, 1688	\ODw@CardSkiptrue	786
\ODw@Bid@Width	75, 836, 882, 927, 967, 1007, 1383, 1426, 1466, 1495, 1655	\ODw@CardStr	1805, 1806, 1827, 1828, 1829, 1830, 1831, 1832, 1833, 1834, 1835, 1836, 1837, 1838, 1839, 1840
\ODw@BidBox	70, 836, 838, 882, 884, 927, 929, 967, 969, 1007, 1009, 1384, 1467, 1518, 1535, 1677, 1681	\ODw@CCnt	1828, 1829, 1830, 1831, 1832, 1833, 1834, 1835, 1836, 1837, 1838, 1839, 1840
\ODw@BidderDefault	1699, 1755	\ODw@ChkNrOfCards	606, 620, 632, 648, 659, 776, 803, 805, 807, 809, 1804
\ODw@BidderFont	86, 1346, 1352, 1358, 1364, 1388, 1394, 1400, 1406, 1443, 1449, 1471, 1477	\ODw@ChkSameCards	660, 661, 662, 663, 709, 725, 738, 751, 764, 777, 804, 806, 808, 810, 849, 850, 851, 852, 895, 896, 897, 898, 940, 941, 942, 943, 980, 981, 982, 983, 1608, 1609, 1610, 1611, 1826
\ODw@BidderI	553, 563, 573, 583, 1347, 1348, 1389, 1390, 1444, 1445, 1472, 1473	\ODw@Clubs	336, 799, 809, 810, 848, 852, 894, 898, 939, 943, 979, 983, 1580, 1611, 1792
\ODw@BidderII	554, 564, 574, 584, 1353, 1354, 1395, 1396	\ODw@clubsuit	115, 122, 138, 151, 166, 182, 198
\ODw@BidderIII	555, 565, 575, 585, 1359, 1360, 1401, 1402, 1450, 1451, 1478, 1479	\ODw@Compass	717, 731, 744, 757, 770, 818,
\ODw@BidderIV	556, 566, 576, 586, 1365, 1366, 1407, 1408		
\ODw@Biddersfalse	1719		
\ODw@Bidderstrue	1720		
\ODw@Bidding	1512		
ODw@Bidding (environment)	1339		
\ODw@Biddingpair	1529		
ODw@Biddingpair (environment)	1436		

860, 909, 920, 951, 991, 1014
 \ODw@CompassDefault
 1056, 1700, 1756
 \ODw@CompassFont 87
 \ODw@Compasssize
 71, 612, 638, 1061, 1067
 \ODw@CompLine 55, 1064, 1073, 1706
 \ODw@CompMid 1174, 1707
 \ODw@CompShowfalse 1712
 \ODw@CompShowtrue 1713
 \ODw@CompSize 54, 1060, 1708
 \ODw@CompTurnfalse 1715
 \ODw@CompTurntrue 1716
 \ODw@CondNewLine 835,
 837, 881, 883, 926, 928,
 966, 968, 1006, 1008, 1652
 \ODw@D 1016, 1039, 1040, 1041,
 1042, 1043, 1044, 1045,
 1046, 1047, 1048, 1049,
 1050, 1051, 1052, 1053,
 1054, 1115, 1128, 1141, 1156
 \ODw@Dealer . 1039, 1040, 1041,
 1042, 1043, 1044, 1045,
 1046, 1047, 1048, 1049,
 1050, 1051, 1052, 1053,
 1054, 1304, 1305, 1307, 1314
 \ODw@DealerText 1283, 1314
 \ODw@descriptionfalse 1511, 1528
 \ODw@descriptiontrue . 1511, 1528
 \ODw@Diagram@Box 68
 \ODw@Diagram@Width
 72, 826, 870, 914,
 958, 998, 1259, 1274, 1656
 \ODw@Diamonds
 337, 796, 807, 808, 847,
 851, 893, 897, 938, 942,
 978, 982, 1581, 1610, 1791
 \ODw@E 387, 389,
 448, 465, 554, 563, 576,
 585, 1132, 1133, 1137, 1138
 \ODw@East 387,
 389, 466, 554, 563, 576, 585
 \ODw@East@Name
 558, 567, 580, 589, 597, 600
 \ODw@EClubs 624, 800, 894, 939, 1800
 \ODw@EDiamonds
 623, 797, 893, 938, 1798
 \ODw@Ehand 625, 822,
 830, 911, 922, 955, 962, 1778
 \ODw@EHearts
 622, 794, 892, 937, 1796
 \ODw@EmptyFooter 66
 \ODw@EmptyHeader 66
 \ODw@Error 1808, 1819, 1857
 \ODw@ESpades
 621, 791, 891, 936, 1794
 \ODw@EW 464
 \ODw@EXtra 1589, 1597, 1619, 1646
 \ODw@FirstBidCol
 550, 1722, 1723, 1724, 1725
 \ODw@FooterText 816,
 858, 904, 949, 989, 1225, 1276
 \ODw@FTfer 363, 1336
 \ODw@GameDefault 1701, 1757
 \ODw@GameFont 91, 1653,
 1867, 1884, 1901, 1918, 1935
 \ODw@GameSize 91, 666,
 698, 712, 728, 741, 754,
 767, 780, 814, 856, 902,
 947, 987, 1506, 1523, 1578
 \ODw@gsetlength 42,
 826, 870, 914, 958, 998, 1061
 \ODw@hand
 613, 627, 641, 654, 693,
 695, 1777, 1778, 1779, 1780
 \ODw@Hand@Box
 69, 664, 674, 675, 1675, 1680
 \ODw@HeaderText 815,
 857, 903, 948, 988, 1223, 1261
 \ODw@Hearts
 338, 793, 805, 806, 846,
 850, 892, 896, 937, 941,
 977, 981, 1582, 1609, 1790
 \ODw@hhand 669, 678
 \ODw@Last 1539, 1543

\ODw@LastTrick	1539, <u>1551</u>	\ODw@North@Name	
\ODw@LeftLowerText 557, 570, 579, 588, 595, 602	
.	823, 831, <u>1234</u> , 1773	\ODw@NS	<u>464</u>
\ODw@LeftUpperText	821,	\ODw@NSpades	
829, 994, 1001, <u>1227</u> , 1772		607, 791, 845, 936, 976, 1794	
\ODw@LegendDefault	1702, 1758	\ODw@OtherDefault	1704, 1760
\ODw@LegendFont		\ODw@OtherFont	
.	<u>89</u> , 1261, 1276, 1671, 1684	<u>90</u> , 211, 372, 385,
\ODw@LongCallsfalse	1730	398, 411, 424, 437, 484,	
\ODw@LongCallstrue	1731	501, 518, 535, 1192, 1285,	
\ODw@mid	1105, <u>1174</u>	1290, 1942, 1959, 1977,	
\ODw@monochromefalse		1995, 2013, 2031, 2038,	
.	149, 164, 180, 196	2056, 2074, 2084, 2094,	
\ODw@monochrometrue	120, 136	2111, 2126, 2129, 2146,	
\ODw@N	374, 376,	2163, 2180, 2197, 2214, 2231	
<u>448</u> , 464, 553, 566, 575,		\ODw@PrErr	<u>1815</u> ,
584, 1119, 1120, 1124, 1125		1828, 1829, 1830, 1831,	
\ODw@NameDefault	1703, 1759	1832, 1833, 1834, 1835,	
\ODw@NameFont		1836, 1837, 1838, 1839, 1840	
.	<u>88</u> , 1371, 1372, 1373,	\ODw@Print	1080, 1081,
1374, 1413, 1414, 1415,		1086, 1089, 1097, 1099, <u>1111</u>	
1416, 1456, 1457, 1484, 1485		\ODw@PrintColor	
\ODw@NameI	557, 567, 577,	1119, 1124, 1132, 1137,
587, 1371, 1413, 1456, 1484		1145, 1152, 1160, 1165, <u>1171</u>	
\ODw@NameII		\ODw@ProcessFooter	
558, 568, 578, 588, 1372, 1414		832, 878, 923, 963, 1003, <u>1270</u>	
\ODw@NameIII	559, 569, 579,	\ODw@ProcessHeader	
589, 1373, 1415, 1457, 1485		828, 872, 916, 960, 1000, <u>1255</u>	
\ODw@NameIV		\ODw@PTfer	<u>342</u> , 365, 1338
560, 570, 580, 590, 1374, 1416		\ODw@RightLowerText	
\ODw@NClubs	823, 831, <u>1248</u> , 1775
610, 800, 848, 939, 979, 1800		\ODw@RightUpperText	
\ODw@NDiamonds		821, 829, 954, 961, <u>1241</u> , 1774	
609, 797, 847, 938, 978, 1798		\ODw@S	400, 402,
\ODw@NextLead	1545,	<u>448</u> , 464, 555, 564, 573,	
1547, 1560, 1561, 1562,		586, 1146, 1148, 1152, 1153	
1563, 1564, 1565, 1566, 1574		\ODw@SClubs	636, 800, 848, 1801
\ODw@Nhand	<u>611</u> , 821, 829, 864,	\ODw@Scratch	50, 51, 85, 1579, 1647
874, 954, 961, 994, 1001, 1777		\ODw@SDiamonds	635, 797, 847, 1799
\ODw@NHearts		\ODw@set	29, 30, 31, 32, 118, 131,
608, 794, 846, 937, 977, 1796		132, 134, 145, 147, 160,	
\ODw@North	374,	162, 175, 176, 178, 191,	
376, <u>466</u> , 553, 566, 575, 584		192, 194, 207, 208, 1711,	

1712, 1713, 1714, 1715,
1716, 1718, 1719, 1720,
1721, 1722, 1723, 1724,
1725, 1726, 1727, 1728,
1729, 1730, 1731, 1733,
1734, 1735, 1736, 1737,
1738, 1739, 1740, 1741,
1742, 1744, 1745, 1746,
1747, 1748, 1749, 1750,
1751, 1752, 1753, 1845,
1847, 1849, 1851, 1853, 1854
\ODw@SetRank
. 225, 279, 283, 285, 287,
289, 291, 293, 294, 295,
296, 297, 298, 299, 300, 302
\ODw@Shand
637, 823, 831, 867, 877, 1779
\ODw@SHearts 634, 794, 846, 1797
\ODw@shorttrue 1510, 1527
\ODw@Skip@Width 76,
77, 95, 1281, 1654, 1658, 1661
\ODw@Skipwidth
. 56, 77, 95, 1280, 1281
\ODw@South 400,
402, 466, 555, 564, 573, 586
\ODw@South@Name
. 559, 568, 577, 590, 596, 602
\ODw@Spades
. 339, 790, 803, 804, 845,
849, 891, 895, 936, 940,
976, 980, 1583, 1608, 1789
\ODw@spadesuit 112,
128, 142, 157, 172, 188, 204
\ODw@SSpades 633, 791, 845, 1795
\ODw@SuitLead
. 226, 230, 250, 279, 315, 326
\ODw@SuitPlayed 225, 230, 237,
250, 268, 269, 270, 271,
272, 279, 282, 284, 286,
288, 291, 293, 294, 295,
296, 297, 298, 299, 300,
315, 319, 320, 321, 322, 326
\ODw@T 1817, 1819, 1822
\ODw@TestIfEmpty
. 815, 816, 857, 858, 903,
904, 948, 949, 988, 989, 1264
\ODw@Tmp@Len 78,
1655, 1656, 1657, 1658, 1659
\ODw@Tmp@Width 79,
1064, 1065, 1677, 1680, 1687
\ODw@translate 258, 261
\ODw@Tricks 364, 1543
\ODw@TrumpSuit . . . 227, 237, 1573
\ODw@V 1016, 1020, 1021, 1022,
1023, 1024, 1025, 1026,
1027, 1028, 1029, 1030,
1031, 1032, 1033, 1034,
1035, 1117, 1118, 1122,
1123, 1130, 1131, 1135,
1136, 1143, 1144, 1150,
1151, 1158, 1159, 1163, 1164
\ODw@vardiamond
. 114, 124, 153, 168, 184, 200
\ODw@varheart
. 113, 126, 155, 170, 186, 202
\ODw@vhand 668, 692
\ODw@Vulner . 1020, 1021, 1022,
1023, 1024, 1025, 1026,
1027, 1028, 1029, 1030,
1031, 1032, 1033, 1034,
1035, 1310, 1311, 1313, 1315
\ODw@VulnerText 1288, 1317, 1319
\ODw@W 413, 415,
448, 465, 556, 565, 574,
583, 1160, 1161, 1165, 1166
\ODw@Warning . . 1811, 1822, 1861
\ODw@WClubs 652, 800, 894, 979, 1801
\ODw@WDiamonds
. . . 651, 797, 893, 978, 1799
\ODw@West 413,
415, 466, 556, 565, 574, 583
\ODw@West@Name
. 560, 569, 578, 587, 597, 601
\ODw@Whand 653, 822,
830, 908, 919, 995, 1002, 1780

10 List of User Commands

<code>\ace</code> , 22	<code>\FSF</code> , 14	<code>\onesuitAll</code> , 15
<code>\Ace</code> , 22	<code>\sforce</code> , 24	<code>\onesuitEW</code> , 16
<code>\alert</code> , 25	<code>\gamefont</code> , 20	<code>\onesuitNE</code> , 16
<code>\all</code> , 24	<code>\GF</code> , 13	<code>\onesuitNS</code> , 16
<code>\All</code> , 24	<code>\gforce</code> , 24	<code>\onesuitNW</code> , 16
<code>\allpass</code> , 12	<code>\hand</code> , 15	<code>\otherfont</code> , 20
<code>\Allpass</code> , 23	<code>\handskip</code> , 19	<code>\pass</code> , 12
<code>\announce</code> , 25	<code>\HCP</code> , 13	<code>\Pass</code> , 22
<code>\bidderfont</code> , 20	<code>\headlinetext</code> , 18	<code>play (env)</code> , 29
<code>bidding (env)</code> , 28	<code>\He</code> , 12	<code>\queen</code> , 22
<code>biddingpair (env)</code> , 29	<code>\HLP</code> , 13	<code>\Queen</code> , 22
<code>\board</code> , 24	<code>\hpts</code> , 23	<code>\redouble</code> , 12
<code>\Board</code> , 24	<code>\jack</code> , 22	<code>\Redouble</code> , 23
<code>\boardnr</code> , 19	<code>\Jack</code> , 22	<code>\resetfonts</code> , 28
<code>\boardtext</code> , 19	<code>\king</code> , 22	<code>\rightlower</code> , 18
<code>\by</code> , 24	<code>\King</code> , 22	<code>\rightupper</code> , 18
<code>\Cl</code> , 12	<code>\lead</code> , 25	<code>\setdefaults</code> , 27
<code>\compassfont</code> , 20	<code>\Lead</code> , 25	<code>\SF</code> , 13
<code>\contract</code> , 24	<code>\leftlower</code> , 18	<code>\sforce</code> , 24
<code>\Contract</code> , 24	<code>\leftupper</code> , 18	<code>\showAll</code> , 17
<code>\deal</code> , 25	<code>\legendfont</code> , 20	<code>\showEW</code> , 17
<code>\Deal</code> , 25	<code>\LP</code> , 13	<code>\showNE</code> , 17
<code>\dealer</code> , 18	<code>\lpts</code> , 23	<code>\showNS</code> , 17
<code>\dealertext</code> , 19	<code>\markit</code> , 25	<code>\showNW</code> , 17
<code>\declarer</code> , 25	<code>\namefont</code> , 20	<code>\south</code> , 13
<code>\Declarer</code> , 25	<code>\namesEW</code> , 14	<code>\South</code> , 23
<code>\Di</code> , 12	<code>\namesNS</code> , 14	<code>\southhand</code> , 14
<code>\double</code> , 12	<code>\newgame</code> , 26	<code>\Sp</code> , 12
<code>\Double</code> , 23	<code>\NMF</code> , 13	<code>\suit</code> , 16
<code>\DP</code> , 13	<code>\nmforce</code> , 24	<code>\TP</code> , 13
<code>\dpts</code> , 23	<code>\none</code> , 24	<code>\tpts</code> , 23
<code>\east</code> , 13	<code>\None</code> , 24	<code>\TSF</code> , 14
<code>\East</code> , 23	<code>\north</code> , 13	<code>\tsforce</code> , 24
<code>\easthand</code> , 14	<code>\North</code> , 23	<code>\vulner</code> , 18
<code>\eastwest</code> , 13	<code>\northhand</code> , 14	<code>\vulnerntext</code> , 19
<code>\EastWest</code> , 23	<code>\northsouth</code> , 13	<code>\west</code> , 13
<code>\expertquiz</code> , 26	<code>\NorthSouth</code> , 23	<code>\West</code> , 23
<code>\explainit</code> , 25	<code>\NT</code> , 12	<code>\westhand</code> , 14
<code>\footlinetext</code> , 18	<code>\nt</code> , 22	