

# The othelloboard package\*

Steven Hall  
stevenhall.uk@gmail.com

28 June 2011

## 1 Drawing a simple board diagram

	a	b	c	d	e	f	g	h
1								
2			●					
3		○	●	●	●		●	
4		○	●	○	●	○		
5			●	●	○	○		
6			●	○	○	○		
7								
8								

The board above was produced by the following list of commands:


```
\begin{othelloboard}{1}  
\dotmarkings  
\othelloarrayfirstrow {0}{0}{0}{0}{0}{0}{0}{0}  
\othelloarraysecondrow {0}{0}{2}{0}{0}{0}{0}{0}  
\othelloarraythirdrow {0}{1}{2}{2}{2}{0}{0}{0}  
\othelloarrayfourthrow {0}{1}{2}{1}{2}{1}{0}{0}  
\othelloarrayfifthrow {0}{0}{2}{2}{1}{1}{0}{0}  
\othelloarraysixthrow {0}{0}{2}{1}{1}{1}{0}{0}  
\othelloarrayseventhrow {0}{0}{0}{0}{0}{0}{0}{0}  
\othelloarrayeighthrow {0}{0}{0}{0}{0}{0}{0}{0}  
\end{othelloboard}
```

---

\*This document corresponds to othelloboard v1.01, dated 28/06/2011.

Try copying this block of code into your own document and typesetting it. Make sure you include `\usepackage{othelloboard}` somewhere in your preamble.

Try replacing any of the 0s, 1s and 2s in the array with any number 0–4. You should see that each number has the following effect (0 has no effect):

1	draws a white disc:		3	draws a white diamond:	
2	draws a black disc:		4	draws a black diamond:	

You should also see that where you choose to put a number in the grid determines where on the Othello board it appears. The `othelloboard` package has been designed to make creating a board diagram as intuitive and simple as possible, so the position of numbers in the array corresponds to exactly the squares you would expect in the diagram. Changing the ‘2’ to a ‘1’ e.g. in the second row of numbers in the code replaces the black disc on c2 with a white one.

## 1.1 The othelloboard environment

### 1.1.1 The basic commands

Look again at the opening and closing lines of code for the diagram above.

```
\begin{othelloboard}{1} % Mandatory argument for board size.
```

```
...
```

```
\end{othelloboard}
```

The board diagram was created by entering a series of commands inside of an `othelloboard` environment. The mandatory argument at the end of the `\begin` environment declaration specifies the size of board. Any size at all is possible without loss of detail in the diagram. The board diagram above was set with default value of 1. A value between 0 and 1 produces a smaller board, while a value greater than 1 produces a larger board.

`\dotmarkings` produces the tiny marker dots on the inside corners of the X-squares.

The command can be deleted or commented out if you don’t want the dots.

Next, the array of discs on the board is drawn by eight separate commands, one for each row. E.g. row three is drawn by this command:

```
\othelloarraythirdrow{0}{1}{2}{2}{2}{0}{0}{0} The command takes eight arguments, one for each square on the third row, a3–h3. The possible values of each argument are 0, 1, 2, 3, 4 (with effects described at the top of this page).
```

If you don’t wish to draw any discs at all for a particular row, it is not necessary to include the command for that row. We could easily, e.g., have omitted the first and eighth row commands in our diagram in the previous section (rather than give the command with a just series of 0s as arguments).

Here is a board drawn at size 0.75 without `\dotmarkings`, and with the `\othelloarray...row` commands for the first and eighth rows omitted:

	a	b	c	d	e	f	g	h	
1									<code>\begin{othelloboard}{0.75}</code>
2			●						<code>\othelloarraysecondrow {0}{0}{2}{0}{0}{0}{0}{0}</code>
3		○	●	●	●				<code>\othelloarraythirdrow {0}{1}{2}{2}{2}{0}{0}{0}</code>
4		○	●	○	●	○			<code>\othelloarrayfourthrow {0}{1}{2}{1}{2}{1}{0}{0}</code>
5			●	●	◆	○			<code>\othelloarrayfifthrow {0}{0}{2}{2}{4}{1}{0}{0}</code>
6			●	◆	◆	○			<code>\othelloarraysixthrow {0}{0}{2}{4}{4}{1}{0}{0}</code>
7					●				<code>\othelloarrayseventhrow {0}{0}{0}{0}{2}{0}{0}{0}</code>
8									<code>\end{othelloboard}</code>

Note the use of black diamonds to show the discs just flipped by the black disc placed at e7. This is the convention followed, e.g., in Brian Rose’s book (Rose 2005).

### 1.1.2 Useful tip

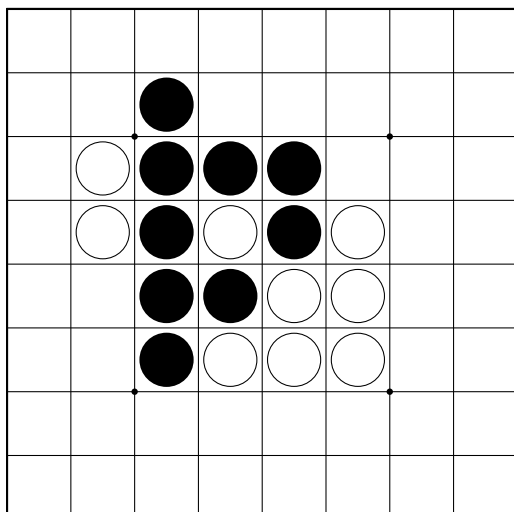
You may find that it helps see where you are putting discs if you align all of the rows of numbers in the array. You can do this (as with the examples given in this document) by putting tabs between the command name and its first argument. Don’t use spaces though, as unlike tabs (which aren’t read as gaps when the file compiles), spaces will break the commands and cause errors.

*tabs not spaces*

```
\othelloarrayfirstrow ↙ {0}{1}{1}{2}{1}{0}{0}{0}
```

## 1.2 The othelloboardnorefs environment

If you don’t want grid reference labels along the top and left edges, you can use the `othelloboardnorefs` environment instead of the `othelloboard` environment. The commands that work inside of the environment are all the same as before, but the outside boundaries of the diagram will be the edges of the 8×8 grid. Our opening example, changing only the environment, gives this output:

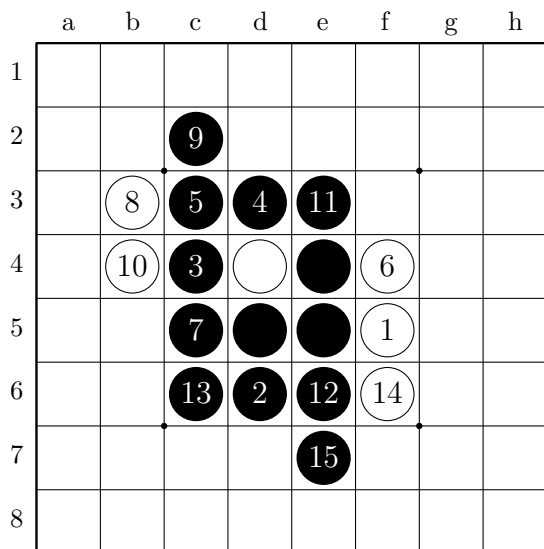


```
\begin{othelloboardnorefs}{1}
\dotmarkings
\othelloarrayfirstrow    {0}{0}{0}{0}{0}{0}{0}{0}
\othelloarraysecondrow   {0}{0}{2}{0}{0}{0}{0}{0}
\othelloarraythirdrow    {0}{1}{2}{2}{2}{0}{0}{0}
\othelloarrayfourthrow   {0}{1}{2}{1}{2}{1}{0}{0}
\othelloarrayfifthrow    {0}{0}{2}{2}{1}{1}{0}{0}
\othelloarraysixthrow    {0}{0}{2}{1}{1}{1}{0}{0}
\othelloarrayseventhrow  {0}{0}{0}{0}{0}{0}{0}{0}
\othelloarrayeighthrow   {0}{0}{0}{0}{0}{0}{0}{0}
\end{othelloboardnorefs}
```

A standard-size `othelloboard` diagram produces a 208pt×208pt box containing a 192pt×192pt board with grid labels in 16pt margins at the top and left side. A standard-size `othelloboardnorefs` diagram produces a 192pt×192pt box containing just the board (see Appendix A).

## 2 Annotations

You may want to add some annotations to some of the squares, perhaps for move numbers or to indicate possible moves at a particular point in a game.



```
\begin{othelloboard}{1}
\dotmarkings
\othelloarraysecondrow   {0}{0}{2}{0}{0}{0}{0}{0}
\othelloarraythirdrow    {0}{1}{2}{2}{2}{0}{0}{0}
\othelloarrayfourthrow   {0}{1}{2}{1}{2}{1}{0}{0}
\othelloarrayfifthrow    {0}{0}{2}{2}{2}{1}{0}{0}
\othelloarraysixthrow    {0}{0}{2}{2}{2}{1}{0}{0}
\othelloarrayseventhrow  {0}{0}{0}{0}{2}{0}{0}{0}
%annotations
\annotationsssecondrow   {} {} {9}{} {} {} {} {}
\annotationsthirdrow     {}{8}{5}{4}{11}{} {} {} {}
\annotationsfourthrow    {}{10}{3}{} {} {6}{} {} {}
\annotationsfifthrow     {} {} {7}{} {} {1}{} {} {}
\annotationsssixthrow     {}{}{13}{2}{12}{14}{}{}
\annotationssseventhrow  {} {} {} {}{15}{} {} {}
\end{othelloboard}
```

	a	b	c	d	e	f	g	h	
1									<code>\begin{othelloboard}{1}</code>
2			●						<code>\dotmarkings</code>
3		○	●	●	●		●		<code>\othelloarraysecondrow</code> {0}{0}{2}{0}{0}{0}{0}{0}
4		○	●	○	●	○			<code>\othelloarraythirdrow</code> {0}{1}{2}{2}{2}{0}{0}{0}
5	c		●	●	○	○			<code>\othelloarrayfourthrow</code> {0}{1}{2}{1}{2}{1}{0}{0}
6			●	○	○	○			<code>\othelloarrayfifthrow</code> {0}{0}{2}{2}{1}{1}{0}{0}
7				b	a?				<code>\othelloarraysixthrow</code> {0}{0}{2}{1}{1}{1}{0}{0}
8									<code>%annotations</code>
									<code>\annotationsssecondrow</code> {} {} {} {} {} {} {} {}
									<code>\annotationsthirdrow</code> {} {} {} {} {} {} {} {}
									<code>\annotationsfourthrow</code> {} {} {} {} {} {} {} {}
									<code>\annotationsfifthrow</code> {c}{} {} {} {} {} {} {}
									<code>\annotationsssixthrow</code> {} {} {} {} {} {} {} {}
									<code>\annotationssseventhrow</code> {} {} {}{b}{a?}{} {} {}
									<code>\end{othelloboard}</code>

A row of annotations is added with an `\annotations...` command. Again, there is one for each row, each taking eight arguments. Here is a possible instance of the command for adding annotations to the fifth row:

`\annotationssfifthrow{}{43}{17}{}{}{a}{b?}{??}` The command has an argument place for each of the squares a5–h5. The command simply prints the value of each argument, centred at the corresponding square. Any string is possible as an argument (including use of symbols in `mathmode`), though you will probably only want to use one- or two-digit numerals or single letters (as in both of the diagrams just given). To leave a square unannotated, put nothing – `{}` – between the braces; putting zero – `{0}` – prints a 0.

To annotate a diagram just add the corresponding annotation command for each row you want to annotate. Make sure that the `\annotations...` commands appear *below* the `\othelloarray...` commands.

**Text colour** You do not have to specify the colour of annotation text, since the `othelloboard` package is smart enough to work out the colour of the disc (or diamond) underneath the text on the same square. Annotations on a black shape are automatically set in white; other annotations are set in the default text colour (black), including those on empty squares.

## 2.1 Placing an individual annotation at a specified square

It is also possible to place an individual annotation at a specific square by giving the square name. The command for this is:

`\posannotation{<squarename>}{<annotationstring>}`

So `\posannotation{f2}{47}` puts the numeral ‘47’ at square f2.

## 2.2 Transcripts

Using `\annotations...` together with `\othelloarray...` commands, it is thus possible to set any transcript in the two standard styles:

	a	b	c	d	e	f	g	h		<code>\begin{othelloboard}{1}</code>	
1	54	51	34	30	31	32	41	42		<code>\othelloarrayfirstrow</code>	<code>{1}{2}{1}{1}{2}{1}{2}{1}</code>
2	55	50	43	33	29	28	39	58		<code>\othelloarraysecondrow</code>	<code>{2}{1}{2}{2}{2}{1}{2}{2}</code>
3	23	27	3	4	25	8	40	59		<code>\othelloarraythirdrow</code>	<code>{2}{2}{2}{1}{2}{1}{1}{1}</code>
4	24	22	5			6	37	60		<code>\othelloarrayfourthrow</code>	<code>{1}{1}{2}{1}{2}{1}{2}{2}</code>
5	47	20	14			1	35	38		<code>\othelloarrayfifthrow</code>	<code>{2}{1}{1}{2}{1}{2}{2}{1}</code>
6	26	21	15	2	9	7	12	36		<code>\othelloarraysixthrow</code>	<code>{1}{2}{2}{1}{2}{2}{1}{1}</code>
7	49	56	16	11	10	18	45	53		<code>\othelloarrayseventhrow</code>	<code>{2}{1}{1}{2}{1}{1}{2}{2}</code>
8	57	46	17	13	44	52	19	48		<code>\othelloarrayeighthrow</code>	<code>{2}{1}{2}{2}{1}{1}{2}{1}</code>
										<code>%annotations</code>	
										<code>\annotationsfirstrow</code>	<code>{54}{51}{34}{30}{31}{32}{41}{42}</code>
										<code>\annotationssecondrow</code>	<code>{55}{50}{43}{33}{29}{28}{39}{58}</code>
										<code>\annotationsthirdrow</code>	<code>{23}{27}{3}{4}{25}{8}{40}{59}</code>
										<code>\annotationsfourthrow</code>	<code>{24}{22}{5}{ } { } {6}{37}{60}</code>
										<code>\annotationsfifthrow</code>	<code>{47}{20}{14}{ } { } {1}{35}{38}</code>
										<code>\annotationssixthrow</code>	<code>{26}{21}{15}{2}{9}{7}{12}{36}</code>
										<code>\annotationseventhrow</code>	<code>{49}{56}{16}{11}{10}{18}{45}{53}</code>
										<code>\annotationseighthrow</code>	<code>{57}{46}{17}{13}{44}{52}{19}{48}</code>
										<code>\end{othelloboard}</code>	

	a	b	c	d	e	f	g	h		<code>\begin{othelloboard}{1}</code>	
1	54	51	34	30	31	32	41	42		<code>\othelloarrayfourthrow</code>	<code>{0}{0}{0}{1}{2}{0}{0}{0}</code>
2	55	50	43	33	29	28	39	58		<code>\othelloarrayfifthrow</code>	<code>{0}{0}{0}{2}{1}{0}{0}{0}</code>
3	23	27	3	4	25	8	40	59		<code>%annotations</code>	
4	24	22	5			6	37	60		<code>\annotationsfirstrow</code>	<code>{54}{51}{34}{30}{31}{32}{41}{42}</code>
5	47	20	14			1	35	38		<code>\annotationssecondrow</code>	<code>{55}{50}{43}{33}{29}{28}{39}{58}</code>
6	26	21	15	2	9	7	12	36		<code>\annotationsthirdrow</code>	<code>{23}{27}{3}{4}{25}{8}{40}{59}</code>
7	49	56	16	11	10	18	45	53		<code>\annotationsfourthrow</code>	<code>{24}{22}{5}{ } { } {6}{37}{60}</code>
8	57	46	17	13	44	52	19	48		<code>\annotationsfifthrow</code>	<code>{47}{20}{14}{ } { } {1}{35}{38}</code>
										<code>\annotationssixthrow</code>	<code>{26}{21}{15}{2}{9}{7}{12}{36}</code>
										<code>\annotationseventhrow</code>	<code>{49}{56}{16}{11}{10}{18}{45}{53}</code>
										<code>\annotationseighthrow</code>	<code>{57}{46}{17}{13}{44}{52}{19}{48}</code>
										<code>\end{othelloboard}</code>	

## 3 Automated methods for board creation





In addition to the manual row-by-row methods described in previous sections for creating board diagrams, `othelloboard` also supports the automated creation of board diagrams and transcripts from long strings of text in a standard format used by Othello software such as WZebra and Cassio.

### 3.1 Drawing a transcript from a list of moves

It is common practice to give a game transcript as a single long alphanumeric string. Here's an example exported from Cassio:

C4C3D3C5B4D2D6C6E6F4B3B5F3F5G3G4A5A6C2A4A3B6E1C1A7F2C7C8D7F6G6F7E2D1E8F8D8E7G5H3F1H6E3G1B2B1H4A1A2A8H5B7B8G2H7H8G8G7H2H1

To draw the board for this transcript you can use `\drawtranscript{<longstring>}`.

	a	b	c	d	e	f	g	h
1	48	46	24	34	23	41	44	60
2	49	45	19	6	33	26	54	59
3	21	11	2	3	43	13	15	40
4	20	5	1			10	16	47
5	17	12	4			14	39	51
6	18	22	8	7	9	30	31	42
7	25	52	27	29	38	32	58	55
8	50	53	28	37	35	36	57	56

```
\begin{othelloboard}{.8}
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\drawtranscript{C4C3D3C5B4D2D6C6E6...B8G2H7H8G8G7H2H1}
\end{othelloboard}
```















































### 3.2 Drawing a board diagram from a single long string

Several programs use a standard long string of dashes, Xs and Os to import and export board diagrams. `othelloboard` also supports such strings. Here is a typical string taken from Cassio:

X0---XXX-000-000-000000---00X0---00X000-00X00000XXXXX---XXXXXX--

An 'X' represents a black disc, an 'O' a white disc, and a dash ('-') an empty square. The string gives the state of every square reading from the top row down, left to right.

To create a board diagram from this string, we simply feed it as an argument into the `\drawboardfromstring{<longstring>}` command:

	a	b	c	d	e	f	g	h
1								
2								
3								
4								
5								
6								
7								
8								

```
\begin{othelloboard}{.8}
\drawboardfromstring{X0---XXX-000-...--XXXXXX--}
\end{othelloboard}
```

## 4 Tweaks

Feel free to tweak the code in the `othelloboard.sty` file if the diagrams aren't exactly how you like them. I've commented on it quite thoroughly so that it should be easy to see which bit to tweak in order, e.g., to change the font size of the grid references, or the size of the discs relative to the grid.

If you know that you will want marker dots on every diagram you make, you can simply add the following to your preamble (though after you call for the package):

```
let\Oldothellogrid\othellogrid
\renewcommand{\othellogrid}{\dotmarkings\Oldothellogrid}
```

Now the `\dotmarkings` command will be automatically executed each time you enter into an `othelloboard` (or `othelloboardnorefs`) environment.

If you have any suggestions for improving or adding to this package, I'd be very keen to hear them. Or if you need some help tweaking the package for a particular application I'd be happy to hear from you about that too. Email me on `stevenhall.uk@gmail.com`.

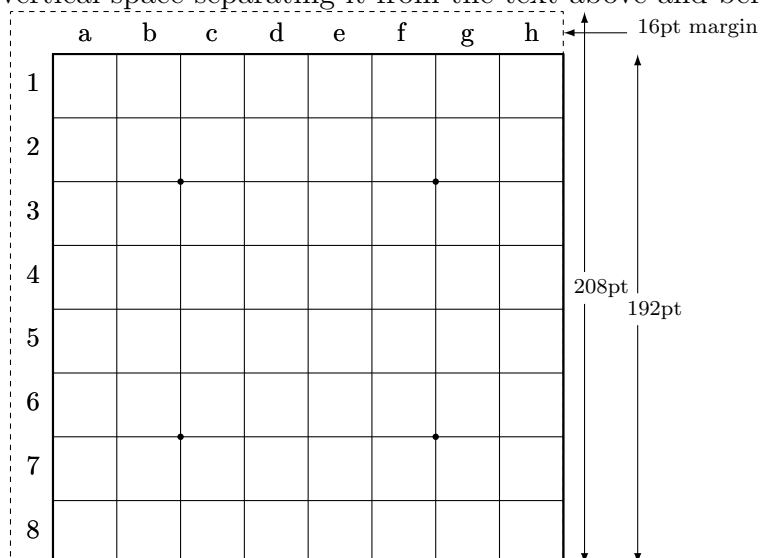


# Appendices

## A Layout of diagrams

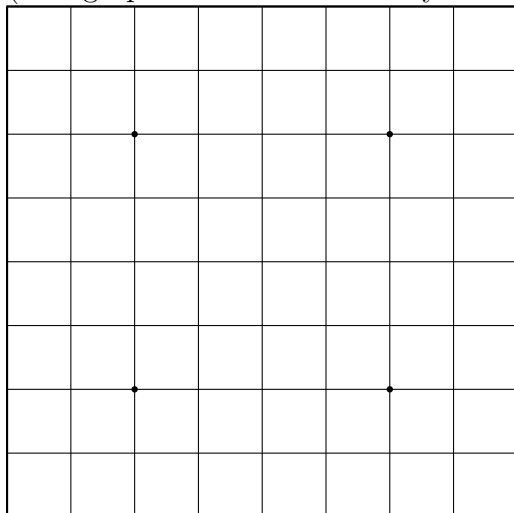
The board diagrams drawn with the `othelloboard` package are contained inside a `picture` environment box. They behave like any diagrams drawn inside of the `picture` environment and can be embedded within further boxes for accurate positioning around the page. They can also be enclosed within a `figure` environment or any other float and treated like any diagram you might insert into a document, e.g. with labels and captions. The function of this package is just to provide diagrams neatly embedded in a box; the positioning and use of this box is up to you. However, here is some more detailed information about how the box is created and a few tips you may find useful.

The grid of a standard-sized diagram is contained in a 192pt×192pt box that itself is positioned in the bottom right corner of a slightly larger 208pt×208pt box (see dotted line), creating 16pt margins on the left and at the top to allow for alphanumeric labels with a little room spare for padding (with scaled diagrams the proportions are all kept the same, including the margins). It is the larger containing box that you control when you create a standard Othello diagram. The diagram immediately below is set with no vertical space separating it from the text above and below.



It is conceivable that you might want to create an Othello board diagram with no labels and no margins at the top and left side. In this case, you can use the `othelloboardnorefs` environment (see 1.2), which works just like the `othelloboard` environment except that it produces board diagrams without labels. This environment leaves out the 208pt×208pt containing box that allows margins for the labels, and just draws the 192pt×192pt grid inside of a 192pt×192pt box. Here is an example, again with the diagram placed on the very next line of text with no vertical space, showing the absence of padding around the board.

(Paragraph of text immediately above the diagram code.)



The last two diagrams are the result of putting an `othelloboard` or `othelloboardnorefs` environment between two paragraphs with no extra thought about spacing and layout. It may be sufficient for your needs simply to add line breaks or vertical space before and after a diagram. Alternatively, here are a couple of examples using other methods with nicer results.

Here's a captioned diagram enclosed and centred within a `figure` environment. The code used for this and the following numbered figures is given at the end of this section.

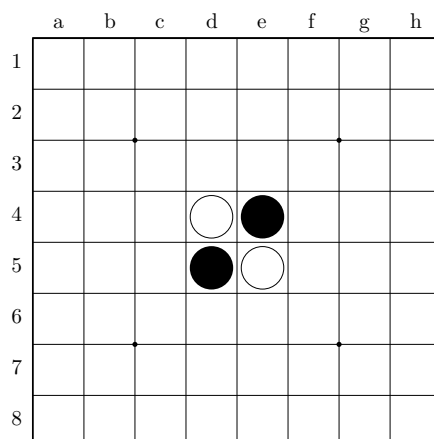


Figure 1: The opening position

Here's an example of a diagram set in a block of text using the `wrapfig` package.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint

blah blah blah blah blah blah Blah blah blah blah blah  
 blah blah Blah blah blah blah blah blah Blah blah  
 blah blah blah blah blah Blah blah blah blah blah blah  
 blah Blah blah blah blah blah blah blah blah Blah blah blah  
 blah blah blah blah Blah blah blah blah blah blah blah  
 Blah blah blah blah blah blah blah Blah blah blah blah  
 blah blah blah Blah blah blah blah blah blah blah Blah  
 blah blah blah blah blah blah Blah blah blah blah blah  
 blah blah Blah blah blah blah blah blah blah Blah blah  
 blah blah blah blah blah Blah blah blah blah blah blah  
 blah Blah blah blah blah blah blah blah Blah blah blah  
 blah blah Blah blah blah blah blah blah blah Blah blah  
 blah blah blah blah blah Blah blah blah blah blah blah  
 blah blah Blah blah blah blah blah blah blah Blah blah  
 blah blah blah blah blah blah

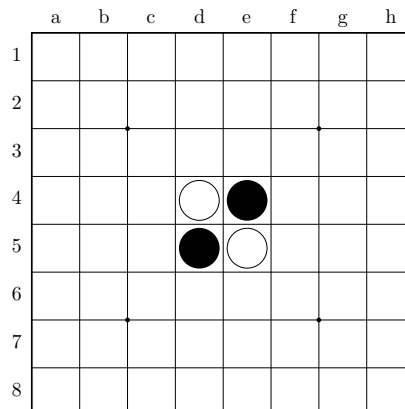


Figure 2: An example using the `wrapfig` package

Here are a couple of diagrams side-by-side in a `figure` environment.

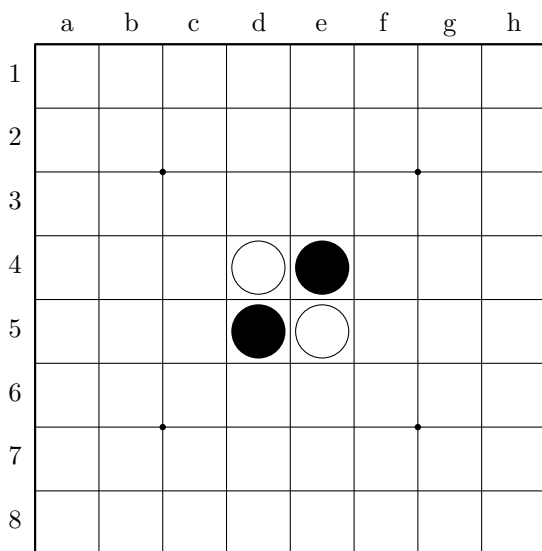


Figure 3: left diagram

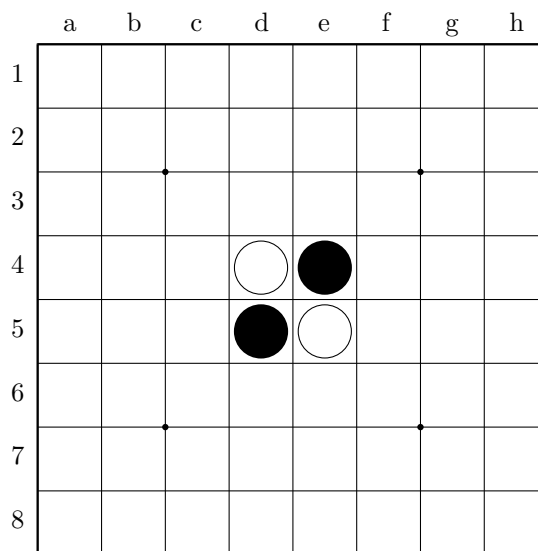


Figure 4: right diagram

You might also consider using `minipages` to align diagrams, or text and diagrams, horizontally:

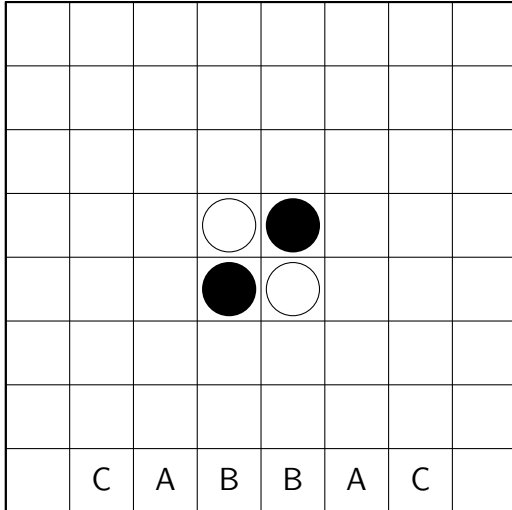


Figure 5

The text over here is in the `minipage` on the right. Both `minipage`s are contained within a `center` environment in this example.

The annotations are in a san-serif font this time.

## A.1 Code for the examples in this section

Figure 1:

```
\begin{figure}[h]
\begin{center}
\begin{othelloboard}{.8}
\dotmarkings
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\end{othelloboard}
\caption{The opening position}
\end{center}
\end{figure}
```

Figure 2:

```
id est laborum ...
\begin{wrapfigure}{r}{152pt}
\begin{othelloboard}{0.75}
\dotmarkings
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\end{othelloboard}
\caption{An example using the \texttt{wrapfig}
package}
\end{wrapfigure}
blah blah blah blah ...
```

Figures 3 & 4:

```
\begin{figure}[ht]
\begin{minipage}[b]{0.5\linewidth}
\centering
\begin{othelloboard}{1}
\dotmarkings
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\end{othelloboard}
```

```
\caption{left diagram}
\end{minipage}
\hspace{0.5cm}
\begin{minipage}[b]{0.5\linewidth}
\centering
\begin{othelloboard}{1}
\dotmarkings
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\end{othelloboard}
\caption{right diagram}
\end{minipage}
\end{figure}
```

Figure 5:

```
\begin{center}
\begin{minipage}[c]{192pt}
\textsf{
\begin{othelloboardnorefs}{1}
\othelloarrayfourthrow {0}{0}{0}{1}{2}{0}{0}{0}
\othelloarrayfifththrow {0}{0}{0}{2}{1}{0}{0}{0}
\annotationseighththrow}{C}{A}{B}{B}{A}{C}{-}
\end{othelloboardnorefs}}
\end{minipage}
\hfill
\begin{minipage}[c]{192pt}
Figure 5\\
```





The text over here is in the `\verb=minipage=` on the right. Both `\verb=minipage=s` are contained within a `\verb=center=` environment in this example.\\

The annotations are in a san-serif font this time.  
`\end{minipage}`  
`\end{center}`

## B Other commands

You may want to exploit some of the lower-level commands that were defined as part of main commands for this package. Suppose you want to draw just a couple of discs and

diamonds, but without putting them in a board diagram. You might want to make sure that these are the same size and style as those used in actual board diagrams (perhaps you're writing a key like the one on page 2). In that case, you can use these commands:

<code>\whitedisc</code>		<code>\whitediamond</code>	
<code>\blackdisc</code>		<code>\blackdiamond</code>	

They work best within a `picture` environment using the `\put` command for accurate placing. Note that the shapes are positioned from their centre-points. These are the same sub-commands that the `\othelloarray...row` commands call upon when they execute, so using these ensures the size and style of shapes will be the same as those appearing within board diagrams within the same document.

## B.1 Counters

The `othelloboard` package uses counters to store the value of discs / diamonds placed on each square (this is the same value you input in order to draw the discs / diamonds). The package currently only uses these counters to determine the text colour of annotations, but these values are available for the current board diagram until another is drawn (the counters are all reset initially as part of the code for a new diagram).

The counter names follow this format:

`disccolourxy`

where `xy` is the name for a square. Since  $\text{\LaTeX}$  doesn't allow numbers in counter names, the `x` and the `y` are both letters `a`–`h`. This needn't be confusing however. The convention is still standard: column name first, row name second. Thus the square `g2` is called `gb` for the purpose of counters.

To write the value (0–4) of the disc at `g2`, simply use the code: `\arabic{disccolourgb}`. To use the value of the disc at `g2` as part of another command, simply use `\value{disccolourgb}`.

To illustrate the availability of these counters, here is a lazy way of counting discs that you might use. First, define a couple of further counters to store the number of white and black discs:

```
\newcounter{numberwdiscs}
\newcounter{numberbdiscs}
```

Next, define a couple of commands for counting up white (black) discs and storing the result in the counters just defined:<sup>1</sup>

```
\newcommand{\countwhitediscs}{% increments the counter for every white disc found.
\ifthenelse{\equal{\value{disccolouraa}}{1}}{\addtocounter{numberwdiscs}{1}}{}
\ifthenelse{\equal{\value{disccolourba}}{1}}{\addtocounter{numberwdiscs}{1}}{}
... }
```

Now you can just run the count commands and write `\arabic{numberwdiscs}` or `\arabic{numberbdiscs}` to print the number of white (black) discs.

	a	b	c	d	e	f	g	h
1	●	●	●	●	●	●	●	●
2	●	●	●	●	●	○	○	●
3	●	○	○	●	○	○	○	●
4	●	○	●	○	○	○	○	●
5	●	●	●	○	○	○	○	●
6	●	●	○	○	○	○	○	●
7	○	○	●	○	●	●	●	●
8	○	○	●	○	○	○	○	○

Number of white discs: 29

Number of black discs: 33

Admittedly this doesn't seem like a very useful feature, but if you like playing with this sort of thing you might find it useful to know that the `disccolourxy` counters are available for use after each diagram is drawn.

## C Most likely errors

- Forgetting to specify a size argument along with the `\begin{othelloboard}` declaration.
- Putting spaces between the 'w' in a command name and the '{' of the first argument. Use tabs if you want to align the arguments for an array (see section 1.1.2 above). Also make sure there are no spaces between arguments.
- Forgetting to include an `\end{othelloboard}` declaration at the end of each diagram.

## References

- [1] Brian Rose (2005). *Othello: A Minute to Learn, A Lifetime to Master*.

---

<sup>1</sup>I've now included the full code for the example at the end of the package file in case you'd like to play with it (so you needn't define these commands yourself, they'll just work). The counting commands don't run automatically however with each diagram, to save unnecessary processing. So if you want to use the value of one of the counters, you have to run the count command first.

	a	b	c	d	e	f	g	h
1	54	51	34	30	31	32	41	42
2	55	50	43	33	29	28	39	58
3	23	27	3	4	25	8	40	59
4	24	22	5			6	37	60
5	47	20	14			1	35	38
6	26	21	15	2	9	7	12	36
7	49	56	16	11	10	18	45	53
8	57	46	17	13	44	52	19	48