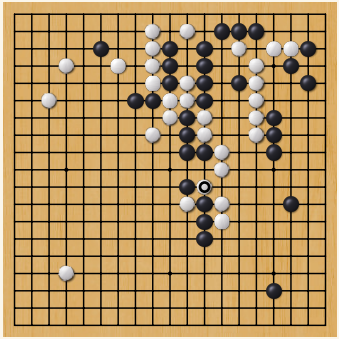
pgf-go

Go boards, games and problems in LaTeX

Jonatán Perren Hug

Version 0.2026.11.27 · November 27, 2026



The tour

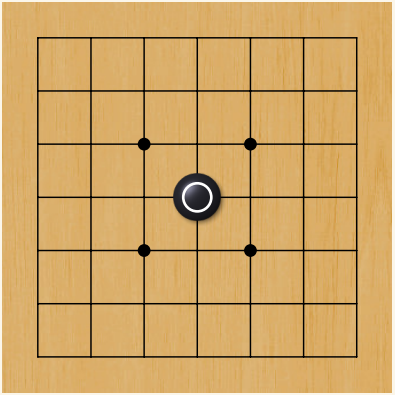


1. First steps
2. The board
3. Coordinates
4. Stones and moves
5. Marks, arrows and lines
6. The game
7. Memory and views
8. Games in SGF
9. Layout
10. Profiles and images
11. Languages
12. Closing

Each example shows the code and what it draws. The slides that advance are seen in any viewer; the animations with buttons, in Acrobat Reader, Foxit or PDF-XChange.

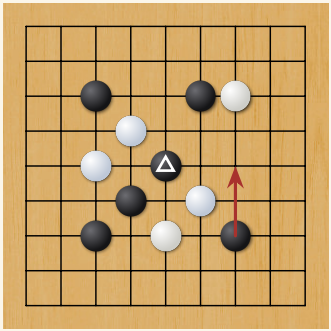
Part 1

First steps



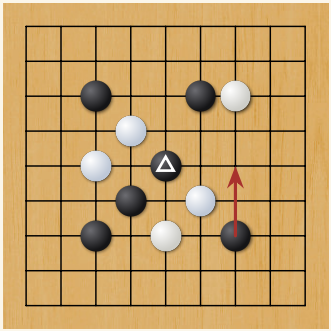
What pgf-go is

- Go boards and positions, in color, for documents and slides.



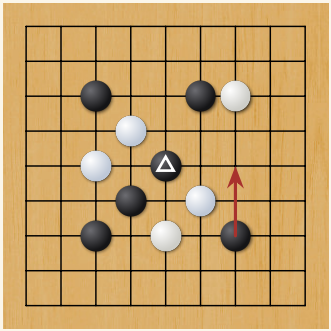
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.



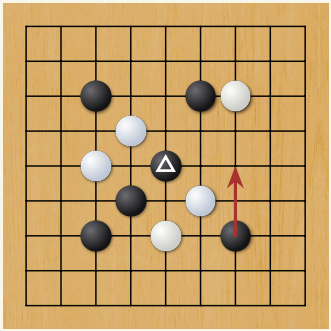
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.



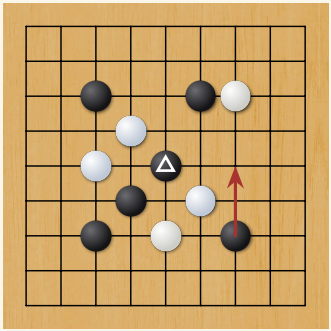
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.



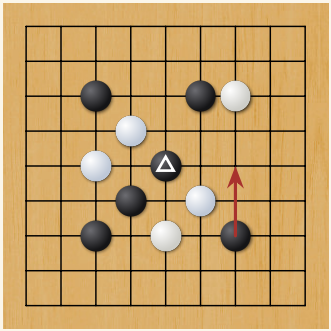
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.
- Boards with a name: variations and views in the middle of the text.



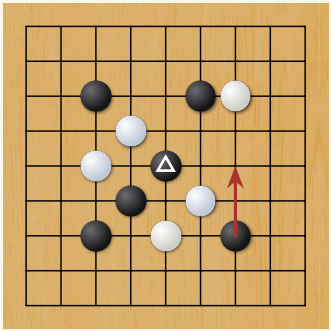
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.
- Boards with a name: variations and views in the middle of the text.
- Marks, arrows, lines and custom marks.



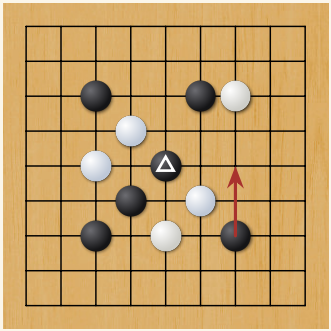
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.
- Boards with a name: variations and views in the middle of the text.
- Marks, arrows, lines and custom marks.
- A grid to lay out diagrams.



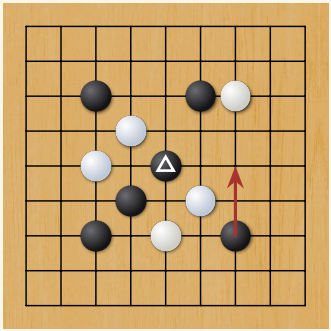
What pgf-go is

- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.
- Boards with a name: variations and views in the middle of the text.
- Marks, arrows, lines and custom marks.
- A grid to lay out diagrams.
- Texts in ten languages and Western, Japanese or Chinese coordinates.



What pgf-go is

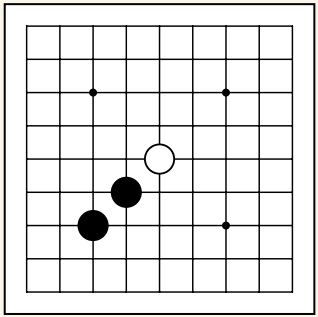
- Go boards and positions, in color, for documents and slides.
- Games in SGF: one board, diagrams as in books, animations and problems.
- The rules, with the Lua kernel: captures, “N at M” notes, territory and score.
- Visual profiles, with images: wood, slate and shell.
- Boards with a name: variations and views in the middle of the text.
- Marks, arrows, lines and custom marks.
- A grid to lay out diagrams.
- Texts in ten languages and Western, Japanese or Chinese coordinates.
- A syntax to write many coordinates with few characters.



A minimal document

```
\documentclass{article}
\usepackage{pgf-go}

\begin{document}
  \begin{goban}
    \stones[b]{C3, D4}
    \stones[w]{E5}
  \end{goban}
\end{document}
```



By default: 9×9, white background, black border, no coordinates.

Two kernels: expl3 and Lua



The computation (coordinates, turns, numbering, memory) is done by a *kernel*; the drawing is always the same.

	`expl3`	`lua`
Engine	any	LuaLaTeX
Stones, moves, marks	✓	✓
Memory and views	✓	✓
Captures (`captures`)		✓
`\territory{auto}`		✓
Games in SGF		✓

With LuaLaTeX, the Lua kernel is used on its own. To choose the other one:

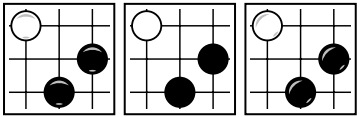
```
\usepackage[kernel = expl3]{pgf-go}
```

Both give the same diagrams.

\pgfgoset and the keys that turn on and off

```
\pgfgoset{goban = {5, partial = B2:D4}}

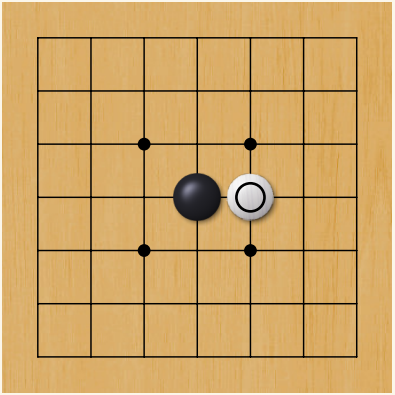
% The key alone (or true) turns on.
\begin{goban}[stone = {shine}]
  \stones[b]{C2, D3} \stone[w]{B4}
\end{goban}
% Giving options does not turn on...
\begin{goban}[stone = {shine = {angle = 45}}]
  \stones[b]{C2, D3} \stone[w]{B4}
\end{goban}
% ...except with true.
\begin{goban}[stone = {shine = {true, angle = 45}}]
  \stones[b]{C2, D3} \stone[w]{B4}
\end{goban}
```



What `\pgfgoset` sets applies to everything that follows; each `goban` can change it.

Part 2

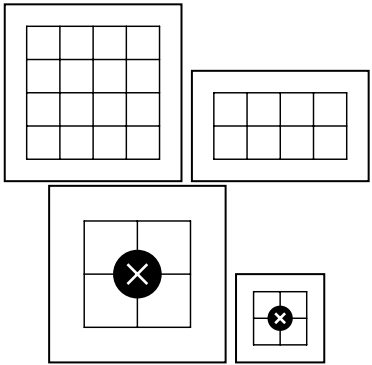
The board



Size and scale

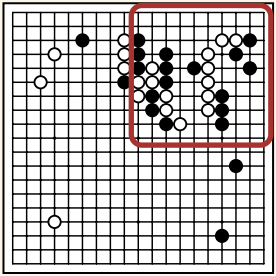
```
\begin{goban}[5]
\end{goban}
\begin{goban}[5x3]
\end{goban}

\begin{goban}[3, scale = 1.6]
  \stone[b, mark]{B2}
\end{goban}
\begin{goban}[3, scale = 0.8]
  \stone[b, mark]{B2}
\end{goban}
```

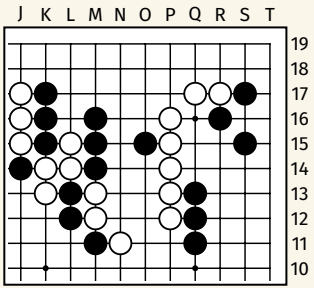


Without the word `size`: a number for a square board, `5x3` for a rectangular one. `scale` does not change the width of the lines; with `scale line width`, it scales it too. With more than 25 columns, `AA`, `AB`… follow Z.

Part of the board: `partial`

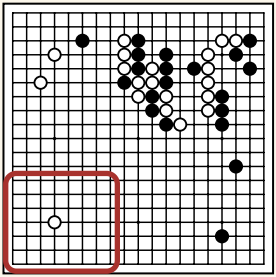


`partial = J10:T19`

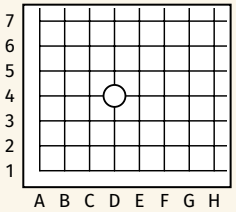


Two opposite corners, in any order: `H7:A1` is the same as `A1:H7`.

Part of the board: `partial`

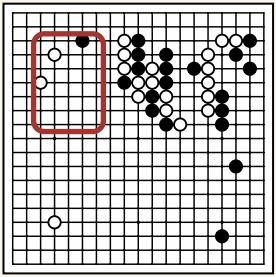


`partial = A1:H7`

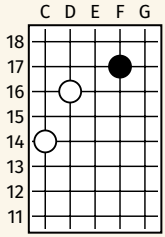


Two opposite corners, in any order: `H7:A1` is the same as `A1:H7`.

Part of the board: `partial`

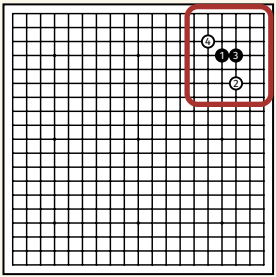


```
partial = C11:G18
```

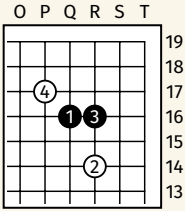


Two opposite corners, in any order: `H7:A1` is the same as `A1:H7`.

Part of the board: `partial`



`partial = auto`: the part used by what is drawn

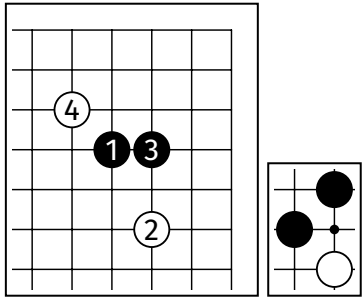


Two opposite corners, in any order: `H7:A1` is the same as `A1:H7`.

partial = auto

```
\pgfgoset{goban = {19, scale = 1.2}}

\begin{goban}[partial = auto]
  \moves{Q16, R14, R16, P17}
\end{goban}
% No extra lines around.
\begin{goban}[partial = auto,
    partial margin = 0]
  \stones[b]{J10, K11}
  \stones[w]{K9}
\end{goban}
```

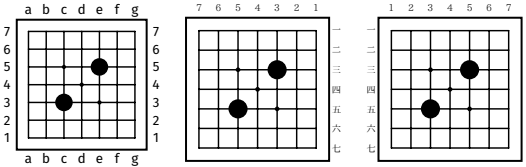


The board takes what the stones, moves, marks, arrows, lines and territories use, plus `partial margin` lines. A side two lines from the edge reaches the edge.

The coordinates on the border

```
\pgfgoset{goban = {7}}

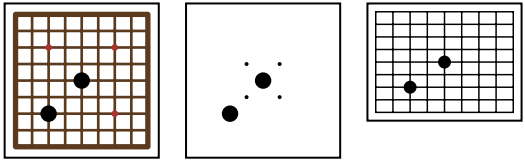
\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[label = {at = all,
      letter case = lower}]
    \stones[b]{C3, E5}
  \end{goban}
  \begin{goban}[label = {at = {north,
   east},
      style = japanese, font = \kanji}]
    \stones[b]{C3, E5}
  \end{goban}
  \begin{goban}[label = {at = {north,
   west},
      style = chinese, font = \kanji}]
    \stones[b]{C3, E5}
  \end{goban}
\end{pgfgolayout}
```



`western`: letters from the left, numbers from the bottom, no I (`letter case`: in lower case). `japanese`: numbers from the right, kanji from the top. The commands do not change.

Grid and star points

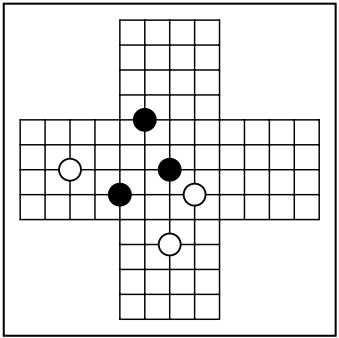
```
\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[grid = {line = {color =
  walnut,
      width = 1pt}, edge = {width = 2pt}},
      hoshi = {radius = 2.5pt, color =
   crimson}]
    \stones[b]{C3, E5}
  \end{goban}
  \begin{goban}[grid = {line = false},
      hoshi = {at = {D4, F6, D6, F4}}]
    \stones[b]{C3, E5}
  \end{goban}
  \begin{goban}[grid = {x sep = 1.8em},
      hoshi = false]
    \stones[b]{C3, E5}
  \end{goban}
\end{pgfgolayout}
```



edge: the four outer lines, with a width or color of their own.

Boards with holes: `blank`

```
\begin{goban}[13, scale = 0.75,
    blank = {A:D 1:4, K:N 1:4,
      A:D 10:13, K:N 10:13}]
  \stones[b]{G7, F9, E6}
  \stones[w]{H6, G4, C7}
\end{goban}
```

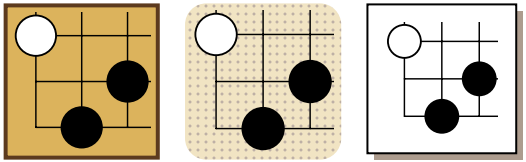


The intersections of `blank` are not part of the board: they have no lines and no star points. Next to a hole, the line ends as at the edge.

The background and the border

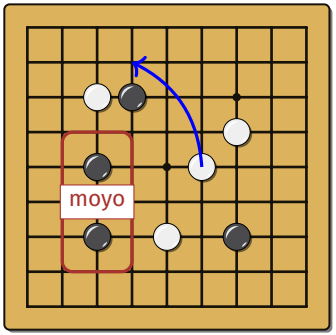
```
\pgfgoset{goban = {5, partial = A1:C3}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[background = {fill =
  {color = wood},
      line = {color = walnut, width =
  1.5pt}}]
    \stones[b]{B1, C2} \stone[w]{A3}
  \end{goban}
  \begin{goban}[background = {fill =
  {color = sand,
      pattern = {name = dots, color =
  walnut!40}},
      line = false, rounded corners = 8pt}]
    \stones[b]{B1, C2} \stone[w]{A3}
  \end{goban}
  \begin{goban}[background = {shadow =
  {true,
      color = walnut, offset = 3pt}, sep =
  1.3em}]
    \stones[b]{B1, C2} \stone[w]{A3}
  \end{goban}
\end{pgfgolayout}
```



TikZ inside the board

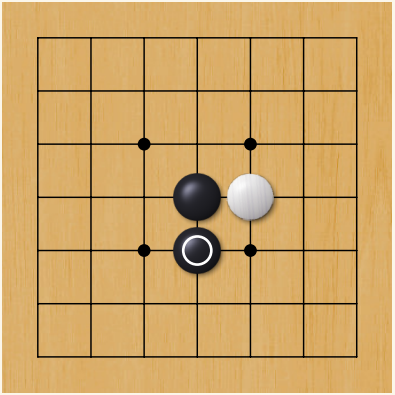
```
\begin{goban}[9, profile = pgfgo, scale = 1.05]
  \stones[b]{C3, C5, D7, G3}
  \stones[w]{E3, F5, G6, C7}
  \draw[crimson, very thick, rounded corners]
    (go cs:B2) rectangle (go cs:D6);
  \node[crimson, fill = white, draw, font = \small]
    at (go cs:C4) {moyo};
  \draw[blue, very thick, ->]
    (go cs:F5) to[bend right] (go cs:D8);
\end{goban}
```



`goban` is a `tikzpicture`: each intersection is a point, `(go cs:C3)`, or `(3,3)`.

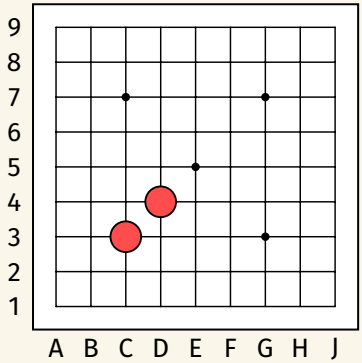
Part 3

Coordinates



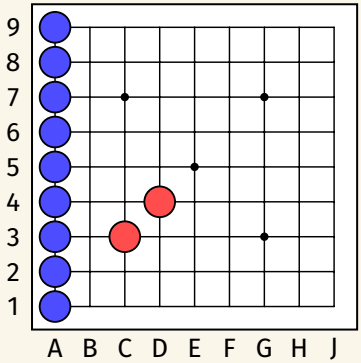
Rows, columns, segments and rectangles

- C3, D4



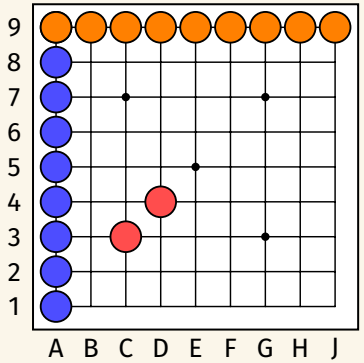
Rows, columns, segments and rectangles

- C3, D4
- A



Rows, columns, segments and rectangles

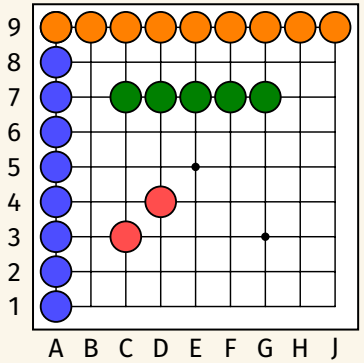
- C3, D4
- A
- 9



Rows, columns, segments and rectangles

- C3, D4
- A
- 9
- C:G7

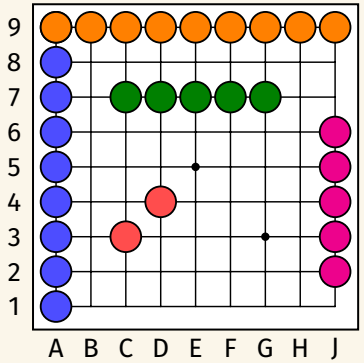
Segments go from the first end to the second:
`J6:2` goes down.



Rows, columns, segments and rectangles

- C3, D4
- A
- 9
- C:G7
- J6:2

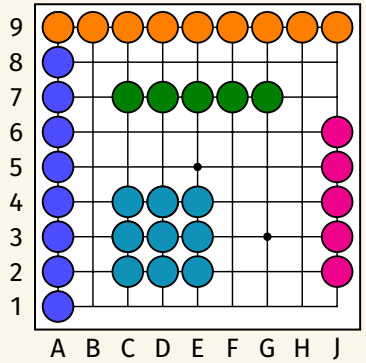
Segments go from the first end to the second:
J6:2 goes down.



Rows, columns, segments and rectangles

- C3, D4
- A
- 9
- C:G7
- J6:2
- C2:E4

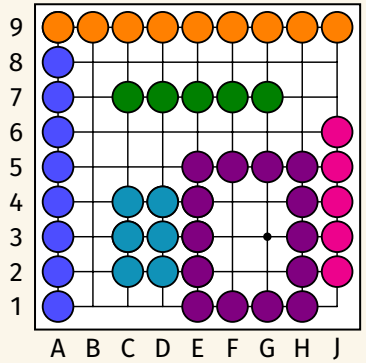
Segments go from the first end to the second: `J6:2` goes down. `LN:LN` is a filled rectangle, as in `partial`; `r`, only its outline.



Rows, columns, segments and rectangles

- `C3, D4`
- `A`
- `9`
- `C:G7`
- `J6:2`
- `C2:E4`
- `E1 r H5`

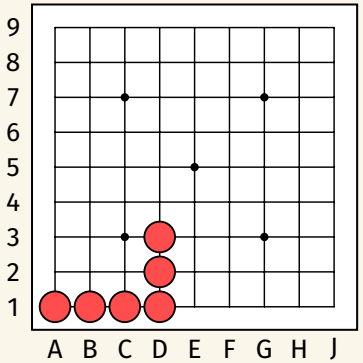
Segments go from the first end to the second: `J6:2` goes down. `LN:LN` is a filled rectangle, as in `partial`; `r`, only its outline.



Paths: -|, |- and --

- A1 -| D3

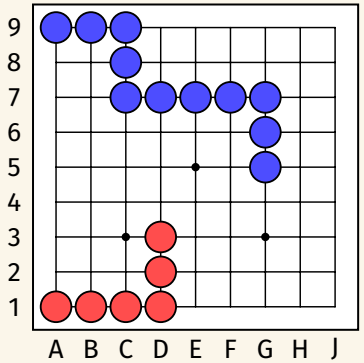
-| goes horizontally first and then vertically, as in TikZ.



Paths: -|, |- and --

- A1 -| D3
- A9 -| C7 -| G5

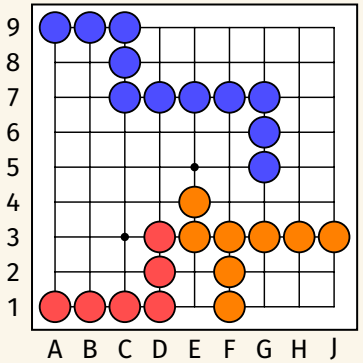
-| goes horizontally first and then vertically, as in Ti*k*Z.



Paths: -|, |- and --

- A1 -| D3
- A9 -| C7 -| G5
- F1 |- J3 -| E4

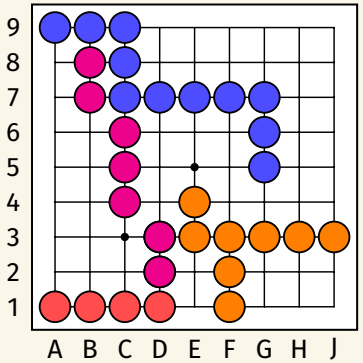
-| goes horizontally first and then vertically, as in Tikz.



Paths: -|, |- and --

- A1 -| D3
- A9 -| C7 -| G5
- F1 |- J3 -| E4
- B8 -- D2

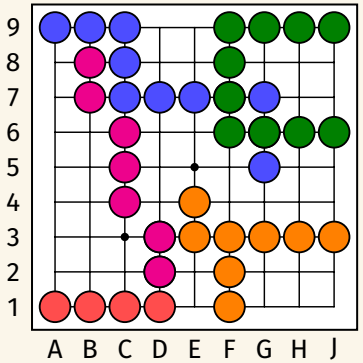
-| goes horizontally first and then vertically, as in Ti*k*Z. -- is the line of stones closest to the straight line.



Paths: -|, |- and --

- A1 -| D3
- A9 -| C7 -| G5
- F1 |- J3 -| E4
- B8 -- D2
- J9 -| F6 -- J6

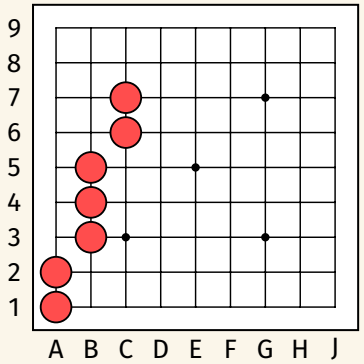
-| goes horizontally first and then vertically, as in Ti*k*Z. -- is the line of stones closest to the straight line. Paths can be chained.



Stairs: -|- and |-|

- A1 -- C7

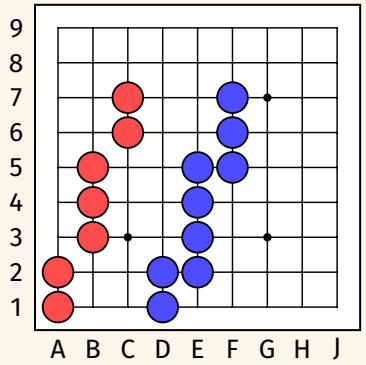
The stones of -- touch at the corners: they do not form a group. -|- fills each step sideways first; |-|, upwards first.



Stairs: -|- and |-|

- A1 -- C7
- D1 -|- F7

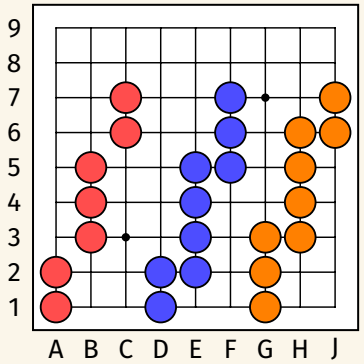
The stones of -- touch at the corners: they do not form a group. -|- fills each step sideways first; |-|, upwards first.



Stairs: -|- and |-|

- A1 -- C7
- D1 -|- F7
- G1 |-| J7

The stones of -- touch at the corners: they do not form a group. -|- fills each step sideways first; |-|, upwards first.



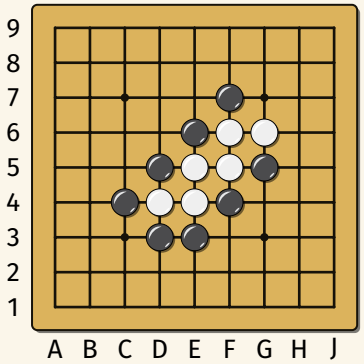
Stairs: -|- and |-|

- A1 -- C7
- D1 -|- F7
- G1 |-| J7

The stones of `--` touch at the corners: they do not form a group. `-|-` fills each step sideways first; `|-|`, upwards first.

A ladder (*shicho*) is written with its ends:

```
\stones [w]{D4 -|- F6 -- G6}
\stones [b]{C4, D3, D5 -- F7, E3 --
G5}
```



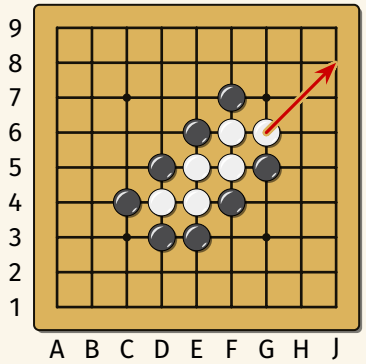
Stairs: -|- and |-|

- A1 -- C7
- D1 -|- F7
- G1 |-| J7

The stones of `--` touch at the corners: they do not form a group. `-|-` fills each step sideways first; `|-|`, upwards first.

A ladder (*shicho*) is written with its ends:

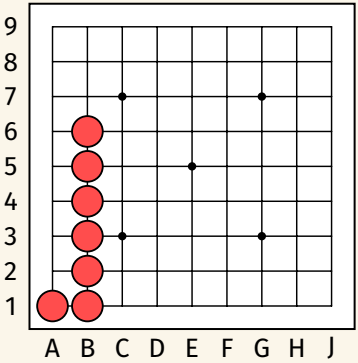
```
\stones [w]{D4 -|- F6 -- G6}
\stones [b]{C4, D3, D5 -- F7, E3 --
G5}
```



Chaining segments and rectangles

- `A1 -| B4:6`

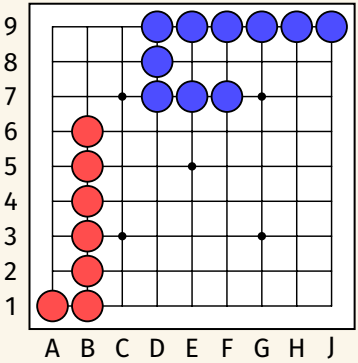
Between two operators there can be a segment or a rectangle: each operator goes from the last point of what comes before to the first one of what follows.



Chaining segments and rectangles

- A1 -| B4:6
- J9 -| D:F7

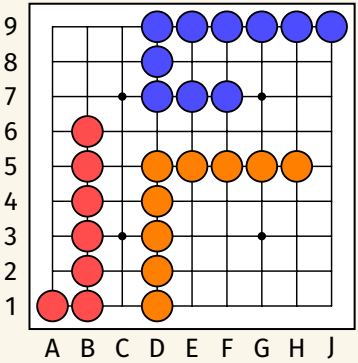
Between two operators there can be a segment or a rectangle: each operator goes from the last point of what comes before to the first one of what follows.



Chaining segments and rectangles

- A1 -| B4:6
- J9 -| D:F7
- D1:3 |- H5

Between two operators there can be a segment or a rectangle: each operator goes from the last point of what comes before to the first one of what follows.

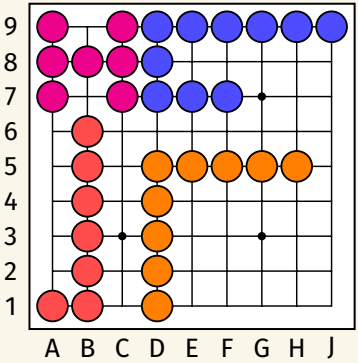


Chaining segments and rectangles

- `A1 -| B4:6`
- `J9 -| D:F7`
- `D1:3 |- H5`
- `A9:7 -- C9:7`

Between two operators there can be a segment or a rectangle: each operator goes from the last point of what comes before to the first one of what follows.

`A9:7 -- C9:7` goes down A, crosses diagonally from A7 to C9 and goes down C: an N.

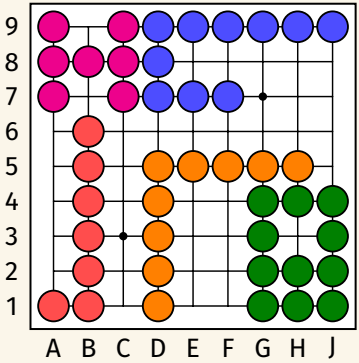


Chaining segments and rectangles

- `A1 -| B4:6`
- `J9 -| D:F7`
- `D1:3 |- H5`
- `A9:7 -- C9:7`
- `J4 r G2 r J1`

Between two operators there can be a segment or a rectangle: each operator goes from the last point of what comes before to the first one of what follows.

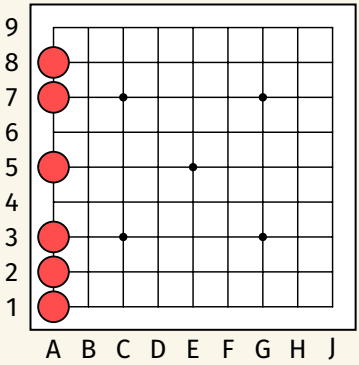
`A9:7 -- C9:7` goes down A, crosses diagonally from A7 to C9 and goes down C: an N.



Cartesian products

- `A[1:3, 5, 7:8]`

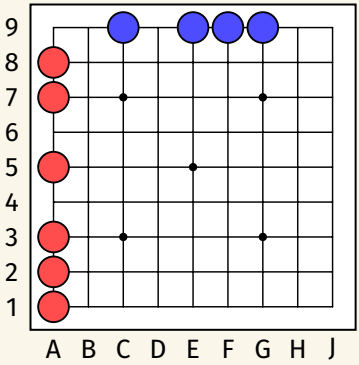
What is outside is distributed over each element inside: `A[1:3, 5, 7:8]` is `A1:3, A5, A7:8`.



Cartesian products

- `A[1:3, 5, 7:8]`
- `[C, E:G]9`

What is outside is distributed over each element inside: `A[1:3, 5, 7:8]` is `A1:3, A5, A7:8`.

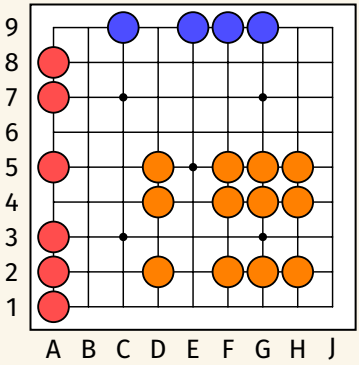


Cartesian products

- `A[1:3, 5, 7:8]`
- `[C, E:G]9`
- `[D, F:H][2, 4:5]`

What is outside is distributed over each element inside: `A[1:3, 5, 7:8]` is `A1:3, A5, A7:8`.

With two brackets, all with all: `D2, D4:5, F:H2, F:H 4:5`.

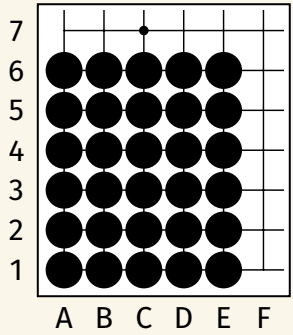


Removing what is left over: the semicolon

After a `;`, the coordinates are removed from the list; after another `;`, they are added again.

```
\stones [b]{A1:E6}
```

A filled rectangle.

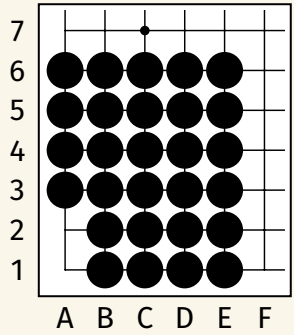


Removing what is left over: the semicolon

After a `;`, the coordinates are removed from the list; after another `;`, they are added again.

```
\stones [b]{A1:E6; A1:2}
```

Without the segment `A1:2`.

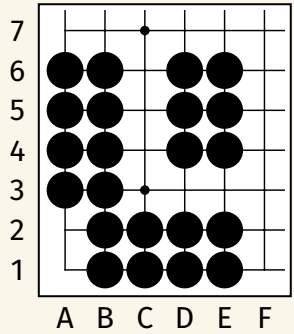


Removing what is left over: the semicolon

After a `;`, the coordinates are removed from the list; after another `;`, they are added again.

```
\stones [b]{A1:E6; A1:2, C6 |- E3}
```

Without the path `C6 |- E3`.

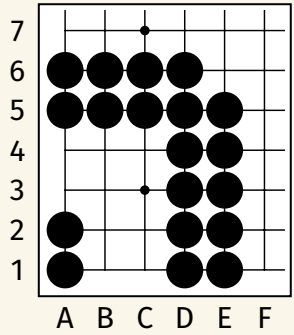


Removing what is left over: the semicolon

After a `;`, the coordinates are removed from the list; after another `;`, they are added again.

```
\stones [b]{A1:E6; A1:C4, E6; A1:2}
```

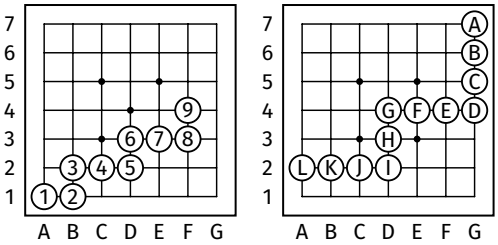
Add, remove, add: `A1:2` comes back.



The order of the intersections

```
\pgfgoset{goban = {7, label = {at = {south,
   west}}}}

\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}
    \stones[w]{A1 -|- F4}
    \marks[sequence]{A1 -|- F4}
  \end{goban}
  \begin{goban}
    \stones[w]{G7:4 -| D2 -- A2}
    \marks[sequence = {format = Alph}]
      {G7:4 -| D2 -- A2}
  \end{goban}
\end{pgfgolayout}
```

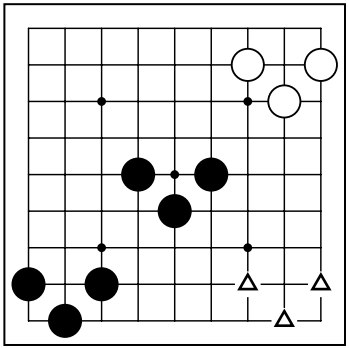


The intersections come out in the order in which they are written: it is the order of `\moves` and of `sequence`. Rectangles have no fixed order: there, you get a warning.

Moving a shape: `shift`

```
\def\tiger{A2, B1, C2}

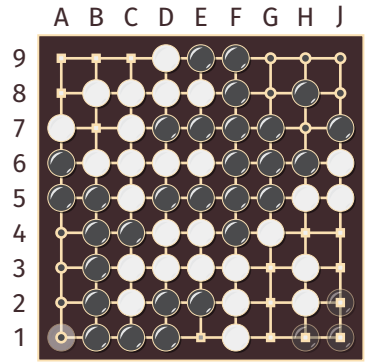
\begin{goban}[9, scale = 1.1]
  \stones[b]{\tiger}
  \stones[b, shift = {3, 3}]{\tiger}
  \stones[w, shift = {6, 6}]{\tiger}
  \marks[triangle, shift = {6, 0}]{\tiger}
\end{goban}
```



`shift = {columns, rows}` moves the coordinates of the command, to the right and up: a shape written once goes anywhere. What falls off the board is dropped, with a warning.

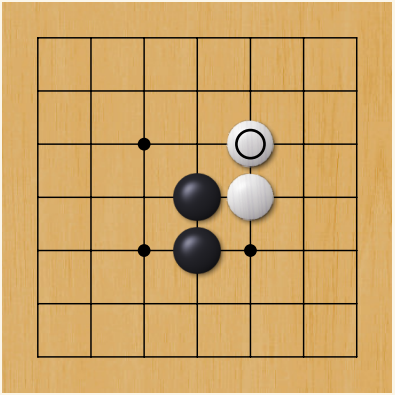
All together: a finished game in two lines

```
\begin{goban}[profile = wain, scale = 1.05,
    label = {at = {above, left}}]
  \stones[b]{A6|-B5|-D1|-E2, C4, E9-|F4,
    D5-|G7--D7, H6, H8, J7}
  \stones[w]{E8-|C5, B:E6, A7, B8, D9,
    J6|-H5, H2:3, F1|-C3, C2, D:E4, G4}
  \prisoners[b]{H1-|J2}
  \prisoner[w]{A1}
  \territory[dame]{auto}
\end{goban}
```



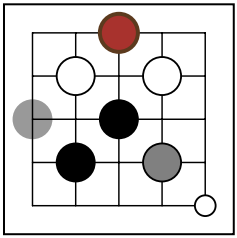
Part 4

Stones and moves



Stones

```
\begin{goban}[5, scale = 1.3]
  \stones[b]{B2, C3}
  \stones[w]{D4, B4}
  \stone[n]{D2} % Neutral: nobody's.
  \stone[fill = {color = crimson},
    line = {color = walnut, width = 1.5pt}]{C5}
  \stone[w, radius = 3pt]{E1}
  \stone[b, fill = {opacity = 0.4}]{A3}
\end{goban}
```

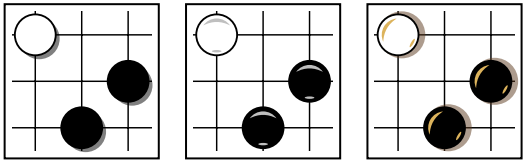


`b`, `w` and `n` choose the player (also `black`, `white`, `neutral`). Each stone accepts the options of `stone`.

Shadows and shine

```
\pgfgoset{goban = {5, partial = B2:D4}}

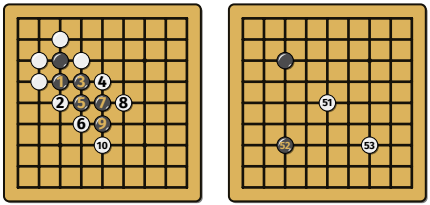
\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[stone = {shadow = {true,
      angle = -45, offset = 1.5pt}}]
    \stones[b]{C2, D3} \stone[w]{B4}
  \end{goban}
  \begin{goban}[stone = {shine}]
    \stones[b]{C2, D3} \stone[w]{B4}
  \end{goban}
  \begin{goban}[stone = {shine = {true,
      angle = 60, color = wood},
      shadow = {true, color = walnut,
      scale = 1.15}}]
    \stones[b]{C2, D3} \stone[w]{B4}
  \end{goban}
\end{pgfgolayout}
```



Numbered moves: \moves

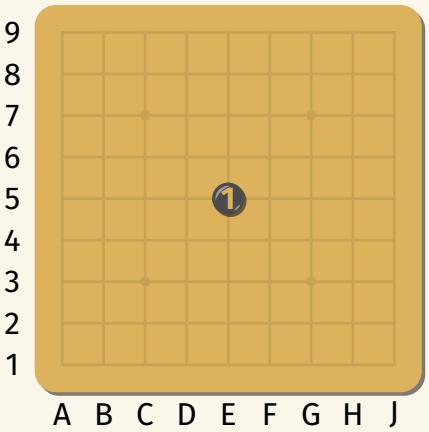
```
\pgfgoset{profile = pgfgo}

\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}
    \stones[b]{C7}
    \stones[w]{B6, B7, C8, D7}
    \moves[b]{C6, C5, D6, E6, D5,
      D4, E5, F5, E4, E3}
  \end{goban}
  \begin{goban}
    \moves[w, from = 51]{E5, C3, G3}
    \moves[numbers = false]{C7}
  \end{goban}
\end{pgfgolayout}
```



Black and White alternate, starting from the given color. `from` chooses the first number; inside a `goban`, the numbering continues from one `\moves` to the next.

Move by move



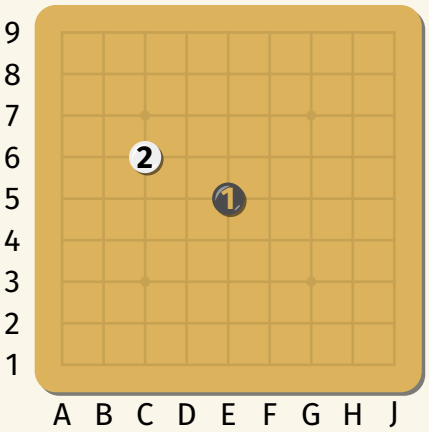
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 1 of 9.

Move by move



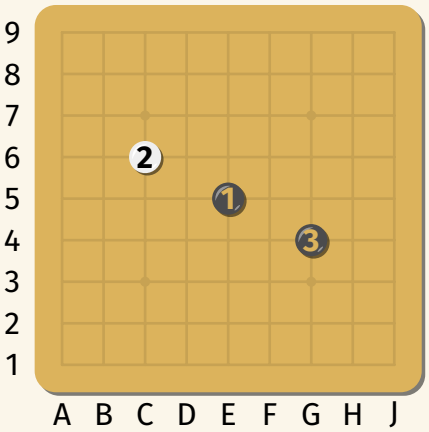
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 2 of 9.

Move by move



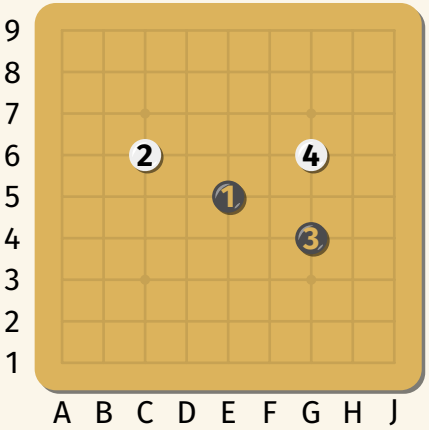
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 3 of 9.

Move by move



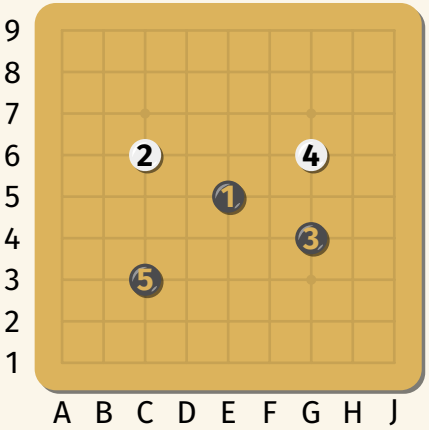
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 4 of 9.

Move by move



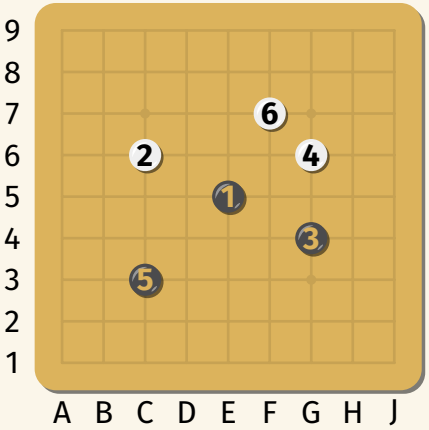
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 5 of 9.

Move by move



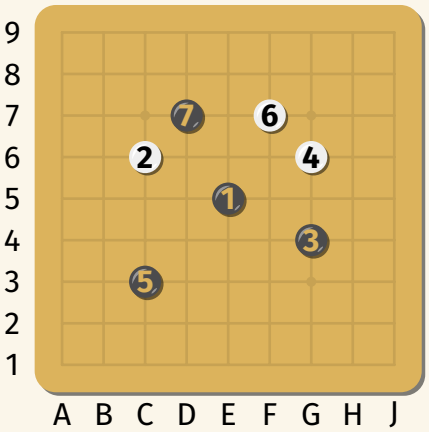
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 6 of 9.

Move by move



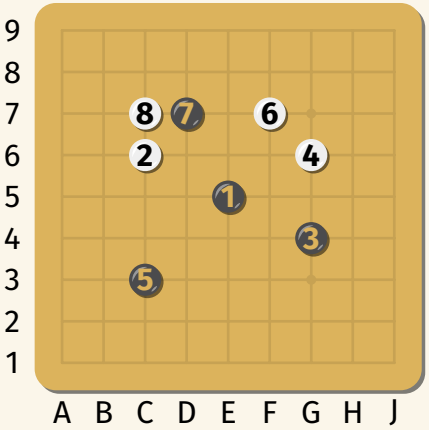
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

...

Move 7 of 9.

Move by move



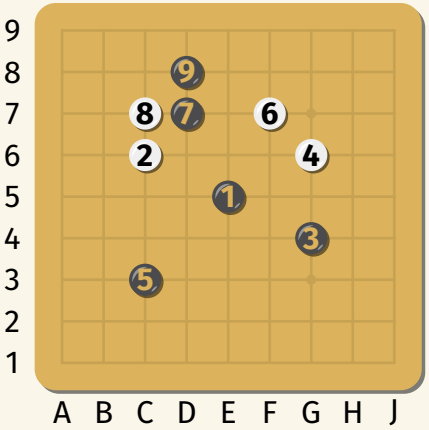
Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

…

Move 8 of 9.

Move by move



Each `\moves` continues the numbering and the turn of the previous one, so beamer's `\only` is enough to add moves one by one:

```
\only<1->{\moves{E5}}
\only<2->{\moves{C6}}
\only<3->{\moves{G4}}
```

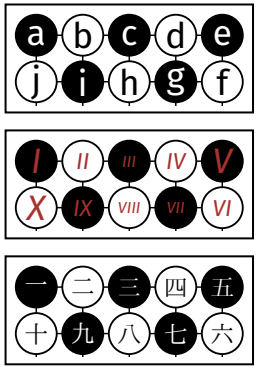
…

Move 9 of 9.

The format of the numbers

```
\pgfgoset{goban = {5, partial = A4:E5, scale = 1.4}}

\begin{pgfgolayout}[columns = 1]
  \begin{goban}[move = {label = {format = alph}}]
    \moves{A:E5, E:A4}
  \end{goban}
  \begin{goban}[move = {label = {format = Roman,
      color = crimson, font = \itshape}}]
    \moves{A:E5, E:A4}
  \end{goban}
  \begin{goban}[move = {label = {format = kanji,
      font = \japanese}}]
    \moves{A:E5, E:A4}
  \end{goban}
\end{pgfgolayout}
```

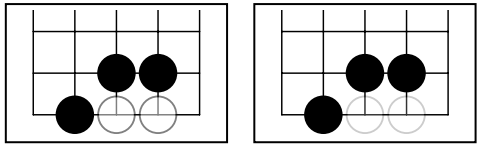


Formats: `arabic`, `alph`, `Alph`, `roman`, `Roman` and `kanji`.

Prisoners

```
\pgfgoset{goban = {5, partial = A1:E3}}

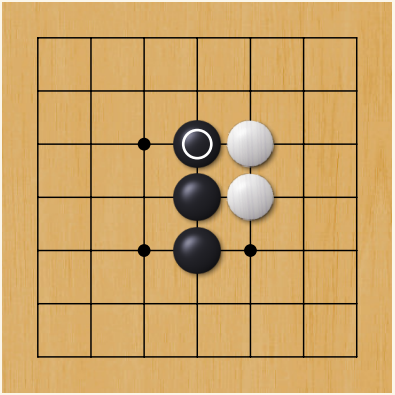
\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}
    \stones[b]{B1, C2, D2}
    \prisoners[w]{C1:D1}
  \end{goban}
  \begin{goban}[prisoner = {fill = {opacity =
   0.2}}]
    \stones[b]{B1, C2, D2}
    \prisoners[w]{C1:D1}
  \end{goban}
\end{pgfgolayout}
```



A prisoner is a dead stone that is still on the board: translucent, and it counts as the opponent's territory.

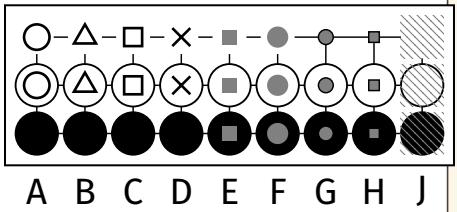
Part 5

Marks, arrows and lines



The marks

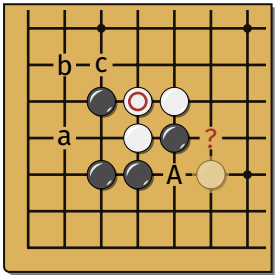
```
\begin{goban}[9x3, scale = 1.45,
    label = {at = {south}}]
  \stones[b]{A:J1}
  \stones[w]{A:J2}
  \marks[circle]{A}
  \marks[triangle]{B}
  \marks[square]{C}
  \marks[cross]{D}
  \marks[filled square]{E}
  \marks[filled circle]{F}
  \marks[t-circle]{G}
  \marks[t-square]{H}
  \marks[hatched lines]{J}
\end{goban}
```



On an empty point, the mark covers the grid; on a stone, it takes the color that stands out.

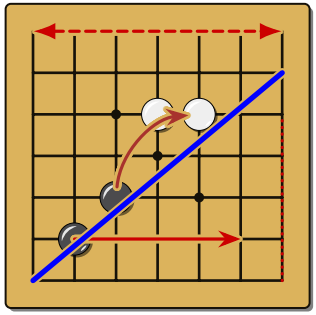
Letters, symbols and sequences

```
\begin{goban}[9, partial = A1:G7, scale = 1.1,
    profile = pgfgo]
  \stones[b]{C3, D3, E4, C5}
  \stones[w]{D4, D5, E5, F3}
  \marks[custom = A]{E3}
  \marks[custom = ?, label = {color = crimson}]{F4}
  \marks[sequence = {format = alph}]{B4, B6, C6}
  \marks[dim]{F3} % Dimmed.
  \marks[circle, line = {color = crimson}]{D5}
\end{goban}
```



Arrows and lines

```
\begin{goban}[7, scale = 1.25, profile = pgfgo]
  \stones[b]{B2, C3}
  \stones[w]{E5, D5}
  \arrow{B2}{F2}
  \arrow[color = crimson, bend = 40,
    shorten = 4pt]{C3}{E5}
  \arrow[dash = dashed, tips = both,
    tip = Latex]{A7}{G7}
  \line[color = blue, width = 2pt]{A1}{G6}
  \line[dash = dotted, halo = false]{G1}{G5}
\end{goban}
```

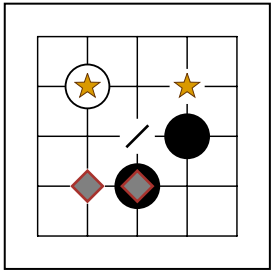


Above the stones, with a halo of the background color that separates them from what is below. The style of all of them, in `goban/arrow` and `goban/line`: each profile brings its own.

Custom marks

```
\pgfgodefinemark{diamond}{
  \pgfpathmoveto{\pgfpointxy{0}{0.7}}
  \pgfpathlineto{\pgfpointxy{0.7}{0}}
  \pgfpathlineto{\pgfpointxy{0}{-0.7}}
  \pgfpathlineto{\pgfpointxy{-0.7}{0}}
  \pgfpathclose \pgfusepath{fill, stroke}}
\pgfgodefinemark{bar}{\draw (-0.5,-0.5) -- (0.5,0.5);}
\pgfgodefinemark[image = {file = pgfgo-star.pdf}]
  {star}

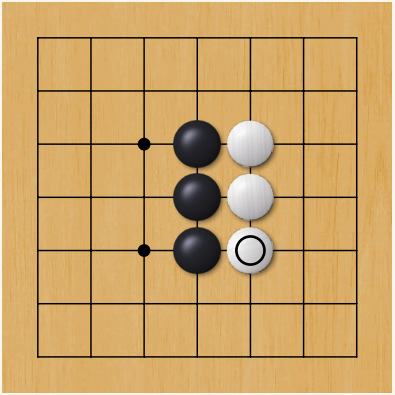
\begin{goban}[5, scale = 1.5]
  \stones[b]{C2, D3} \stone[w]{B4}
  \marks[diamond, line = {color = crimson}]{C2, B2}
  \marks[bar]{D3, C3}
  \marks[star]{D4, B4}
\end{goban}
```



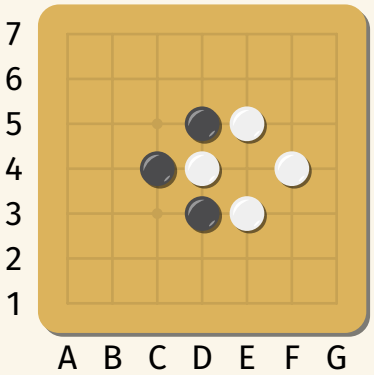
pgf or TikZ code, measured in stone radii, or an image.

Part 6

The game



Captures

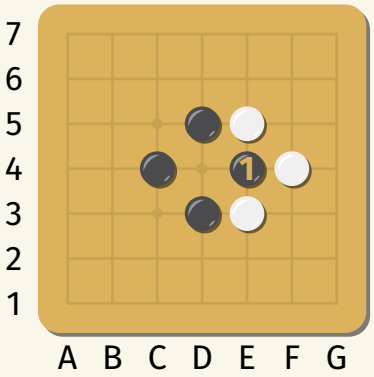


With `captures`, the `goban` follows the rules: a group without liberties is removed.

The position.

Prisoners: Black 0, White 0.

Captures

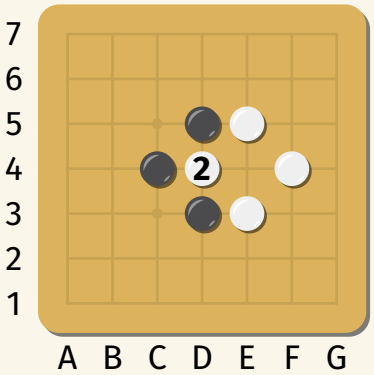


With `captures`, the `goban` follows the rules: a group without liberties is removed.

1: Black captures the white stone at D4...

Prisoners: Black 1, White 0.

Captures

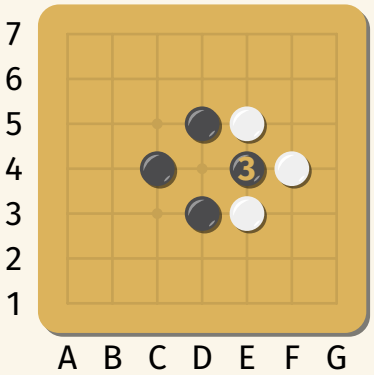


With `captures`, the `goban` follows the rules: a group without liberties is removed.

2: … and White takes back at D4, capturing 1.

Prisoners: Black 1, White 1.

Captures



With `captures`, the `goban` follows the rules: a group without liberties is removed.

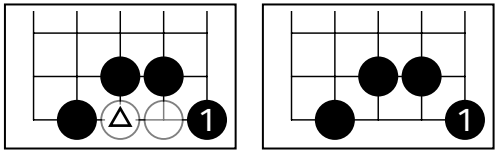
3: Black takes back again. It is a ko: in a game, 3 would have to wait one move. The `goban` does not prevent it.

Prisoners: Black 2, White 1.

Captures you can see: fade

```
\pgfgoset{goban = {5, partial = A1:E3}}

\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}[captures = fade]
    \stones[w]{C1, D1}
    \stones[b]{B1, C2, D2}
    \marks[triangle]{C1}
    \moves{E1}
  \end{goban}
  \begin{goban}[captures,
      captured marks = remove]
    \stones[w]{C1, D1}
    \stones[b]{B1, C2, D2}
    \marks[triangle]{C1}
    \moves{E1}
  \end{goban}
\end{pgfgolayout}
```

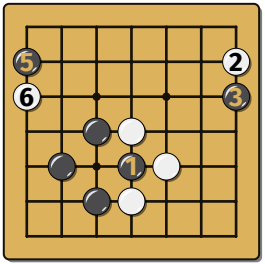


With `fade`, the captured stones stay as prisoners. `captured marks` decides whether the marks go with them.

The ko and the “N at M” notes

```
\begin{goban}[7, captures, notes, scale = 1.05,
    profile = pgfgo]
  \stones[b]{B3, C2, C4}
  \stones[w]{C3, D2, D4, E3}
  \moves{D3, G6, G5, C3, A6, A5}
  \moves{D3} % 7 retakes the ko.
\end{goban}

\pgfgomovenotes. Prisoners: Black
\pgfgoprisoners{b}, White \pgfgoprisoners{w}.
```



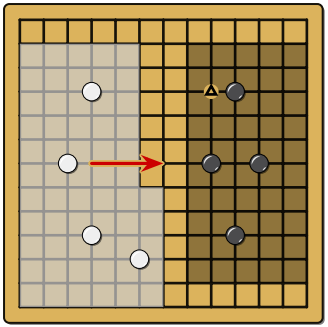
7 at 1. Prisoners: Black 2, White 1.

A move on a point already played is not drawn: it is noted below, as in books.

Territory by hand: polygons

```
\begin{goban}[13, profile = pgfgo, scale = 0.72]
  \stones[b]{K4, K10, J7, L7}
  \stones[w]{D4, D10, C7, F3}
  \territory[b, draw = region]{J3, M3, M11, J11}
  \territory[w, draw = region]
    {B2, F2, F5, E5, E11, B11}
  \marks[triangle]{J10}
  \arrow{D7}{G7}
\end{goban}

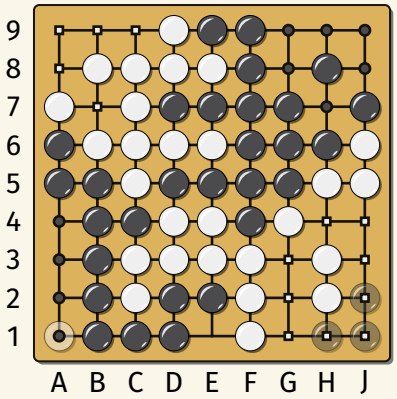
Black framework: \pgfgoregion{b}{area} points;
White: \pgfgoregion{w}{area}.
```



Black framework: 50 points; White: 60.

A polygon by its vertices. With `draw = region`, each color's territory is a single polygon that `\pgfgoregion` measures.

Automatic territory

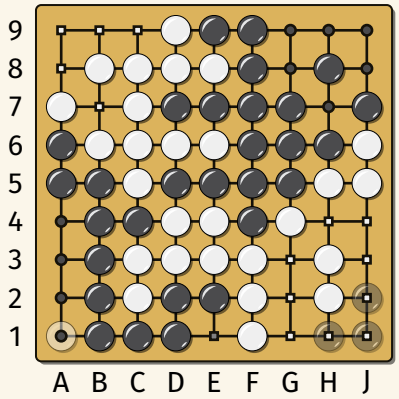


`\territory{auto}`: each empty region belongs to the color that surrounds it; prisoners count as dead.

`\territory {auto}`
The marks of each player.

Black 11, White 23.5.

Automatic territory

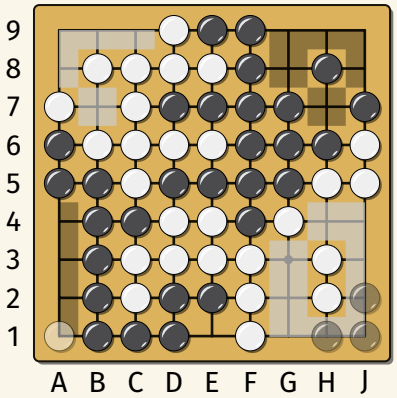


`\territory{auto}`: each empty region belongs to the color that surrounds it; prisoners count as dead.

`\territory [dame]{auto}`
The neutral points too.

Black 11, White 23.5.

Automatic territory

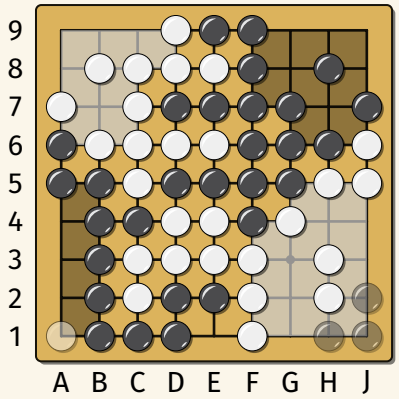


`\territory{auto}`: each empty region belongs to the color that surrounds it; prisoners count as dead.

`\territory [draw = fill]{auto}`
A fill under the stones.

Black 11, White 23.5.

Automatic territory

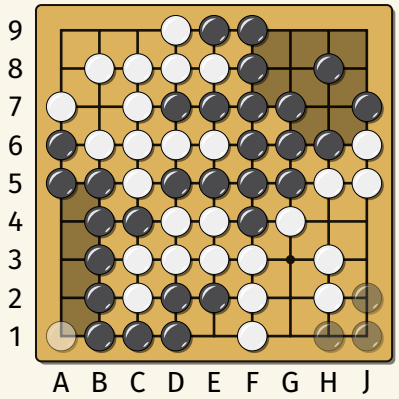


`\territory{auto}`: each empty region belongs to the color that surrounds it; prisoners count as dead.

`\territory [draw = region]{auto}`
One polygon per color.

Black 11, White 23.5.

Automatic territory



`\territory{auto}`: each empty region belongs to the color that surrounds it; prisoners count as dead.

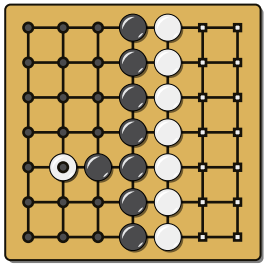
`\territory [draw = region, b]{auto}`
Only Black's.

Black 11, White 6.5.

The score of a finished game

```
\begin{goban}[sgf size = commented, captures,
    profile = pgfgo, scale = 1.05]
  \sgf[marks = false, numbers = false]{commented}
  \prisoner[w]{B3} % The dead white stone.
  \territory{auto}
\end{goban}

Black \pgfgoscore{b}, White
\pgfgoscore[komi = 6.5]{w}.
According to the file: \sgfinfo{commented}{RE}.
```



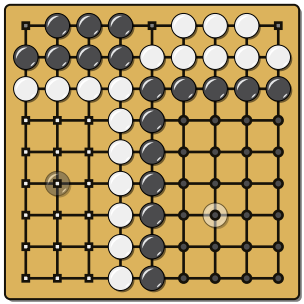
Black 21, White 20.5. According to the file: B+0.5.

Score with Japanese rules: territory, plus the opponent's dead stones and captures, plus komi.

Seki: eyes that are not territory

```
\begin{goban}[profile = pgfgo, scale = 0.95]
  \stones[b]{A8:D8, B9:D9, E7:J7, E1:E6}
  \stones[w]{E8:J8, F9:H9, A7:D7, D1:D6}
  \prisoner[w]{G3}
  \prisoner[b]{B4}
  \territory[dame]{auto}
  \territory[n]{A9}
  \territory[n]{J9}
\end{goban}

Black \pgfgoscore{b},
White \pgfgoscore[komi = 6.5]{w}.
```

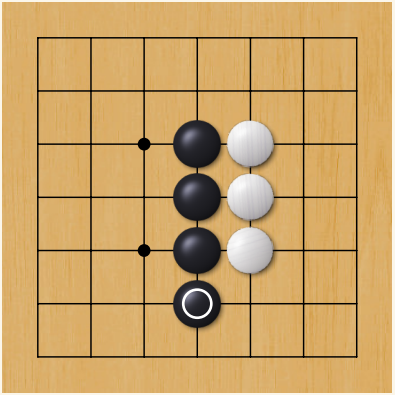


Black 25, White 25.5.

`auto` looks at which color touches each region, it does not read the position: the eyes of the seki become neutral with one `\territory[n]` for each.

Part 7

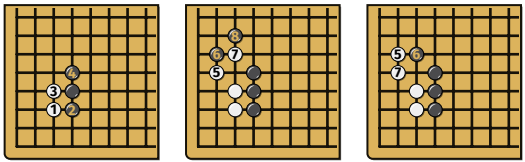
Memory and views



Variations with `name` and `resume`

```
\pgfgoset{goban = {19, partial = A1:H8,
  profile = pgfgo}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[name = base]
    \stones[b]{D4}
    \moves[w]{C3, D3, C4, D5}
  \end{goban}
  \begin{goban}[resume = base]
    \moves{B5, B6, C6, C7}
  \end{goban}
  \begin{goban}[resume = base]
    \moves{B6, C6, B5}
  \end{goban}
\end{pgfgolayout}
```

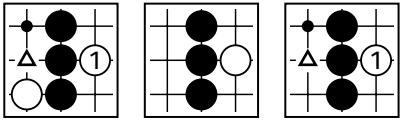


A `goban` with a name remembers what it draws. `resume` starts with that board: the numbering and the turn continue from there.

What resume brings

```
\pgfgoset{goban = {5, partial = B2:D4}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[name = saved]
    \stones[b]{C2:4}
    \moves[w]{D3}
    \marks[triangle]{B3}
    \stone[w, forget]{B2}
    \territory[b]{B4}
  \end{goban}
  \begin{goban}[resume = saved]
  \end{goban}
  \begin{goban}[resume = {saved,
      numbers, marks, territory}]
  \end{goban}
\end{pgfgolayout}
```



By default, only the stones. What is drawn with `forget` does not travel.

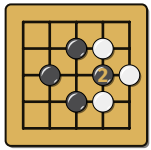
Views in the middle of the text

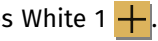
```
\begin{goban}[5, name = ko, captures,
    profile = pgfgo, scale = 0.8]
  \stones[b]{B3, C2, C4}
  \stones[w]{D2, D4, E3}
  \moves[w]{C3, D3}
\end{goban}

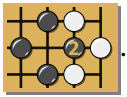
Black 2 \pgfgoview{ko}{D3} captures
White 1 \pgfgoview{ko}{C3}.

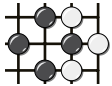
The ko: \pgfgoview{ko}{B2:E4}.

Without numbers or background:
\pgfgoview[numbers = false,
  background = false]{ko}{B2:E4}.
```



Black 2  captures White 1 .

The ko: .

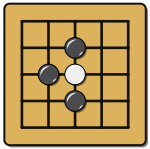
Without numbers or background: .

`\pgfgoview` shows an intersection or a rectangle of a `goban` with a name, as it was at the end, at the size of the text.

A stone alone and views of several parts

```
\begin{goban}[5, name = net,
    profile = pgfgo, scale = 0.8]
  \stones[b]{B3, C2, C4}
  \stones[w]{C3}
\end{goban}

The white stone \pgfgostone[w, mark = triangle]
has three black neighbours:
\pgfgoview{net}{B3, C2, C4}.
Black plays 1 \pgfgostone[b, number = 1]
at D3 and captures it.
```



The white stone Ⓐ has three black neighbours: , and . Black plays 1 ❶ at D3 and captures it.

`\pgfgostone`: a stone without a board, with a mark or a number. With commas, `\pgfgoview` gives one view per part, joined with `separator` and `last`.

Views before the board, and options

```
Further below, the white stone at C3
  \pgfgoview{atari}{C3}
is in atari. With another profile:
\pgfgoview[profile = simple]{atari}{B2:D4};
without arrows: \pgfgoview[arrows = false,
  raise = 0pt]{atari}{B2:D4}.

\begin{goban}[5, name = atari, scale = 1.0]
  \stones[b]{B3, C2, C4}
  \stones[w]{C3}
  \marks[sequence]{D3}
  \arrow{E5}{D4}
\end{goban}
```

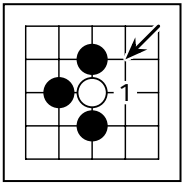
Further below, the white stone at C3 ○ is in atari. With another profile:



; without arrows:



.



A view can go before its board, like a reference: it is resolved with the `.aux`, in the next compilation.

A board in another document



In the article:

```
\begin{goban}[19, name = fight,
    export = fight]
  ...
\end{goban}
```

The `goban` is saved in `fight.pgfgo`, with its style.

In the slides:

```
\pgfgoimport{fight}
...
The ko of \pgfgoview{fight}{Q16:S18}.

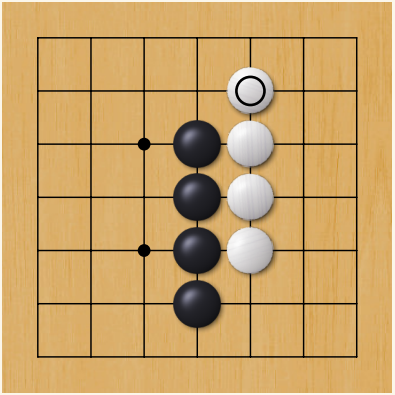
\begin{goban}[resume = {fight, marks},
    partial = auto]
  \moves{E3}
\end{goban}
```

As if it had been drawn there: it works for `\pgfgoview` and for `resume`.

For another program, the package option `pages`: one `goban` per page, and a script that turns each one into SVG or PNG, with its name.

Part 8

Games in SGF



A name for each file, and its information

```
% In the preamble:
% \pgfgodefinesgf{lee}{lee.sgf}

\begin{tabular}{rl}
  Black: & \sgfinfo{lee}{PB}
   (\sgfinfo{lee}{BR})\\
  White: & \sgfinfo{lee}{PW}\\
  Event: & \sgfinfo{lee}{RO}\\
  Place: & \sgfinfo{lee}{PC}\\
  Date: & \sgfinfo{lee}{DT}\\
  Rules: & \sgfinfo{lee}{RU}, komi
   \sgfinfo{lee}{KM}\\
  Result: & \sgfinfo{lee}{RE}
\end{tabular}
```

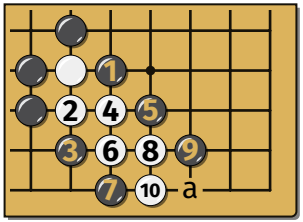
Black:	Lee Sedol (9d)
White:	AlphaGo
Event:	Game 1
Place:	Seoul, Korea
Date:	2016-03-09
Rules:	Chinese, komi 7.50
Result:	W+Resign

With a name, the path is written once and the file is read only once in the whole document.

Diagrams of Sensei's Library

```
\begin{pgfgosgf}{ladder}
$$B The ladder works
$$ . X . . . . . |
$$ X O 1 . . . . |
$$ X 2 4 5 . . . |
$$ . 3 6 8 9 . . |
$$ . . 7 0 a . . |
$$ --------------+
\end{pgfgosgf}

\sgfdiagrams[goban = {profile = pgfgo,
  scale = 1.2}]{ladder}
```



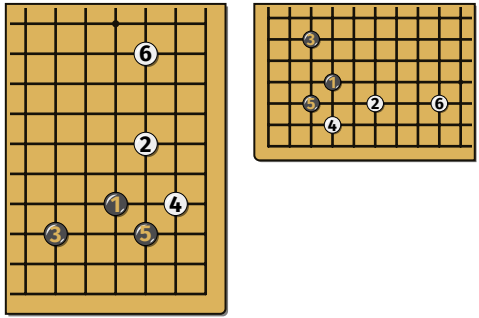
The ladder works (1–10)

A diagram copied from Sensei's Library, or from a forum between `[go]` and `[/go]`, is read as a game: the title is the caption, the numbers are moves and the edges place it on the board.

Turning a file: `origin` and `transpose`

```
% In the preamble:
% \pgfgodefinesgf[origin = bottom left]
% {below}{joseki.sgf}
% \pgfgodefinesgf[transpose]
% {transposed}{joseki.sgf}

\pgfgoset{profile = pgfgo,
  goban = {19, partial = auto}}
\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}\sgf{below}\end{goban}
  \begin{goban}\sgf{transposed}\end{goban}
\end{pgfgolayout}
```

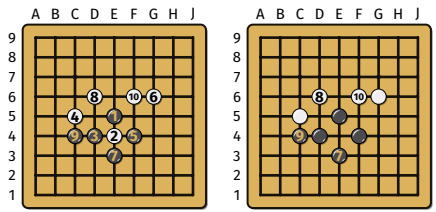


`origin`: the corner where the point `aa` of the file goes. `transpose`: swaps rows and columns. Together, the eight symmetries of the board; the joseki of the file is in the upper right.

\sgf: a game on a board

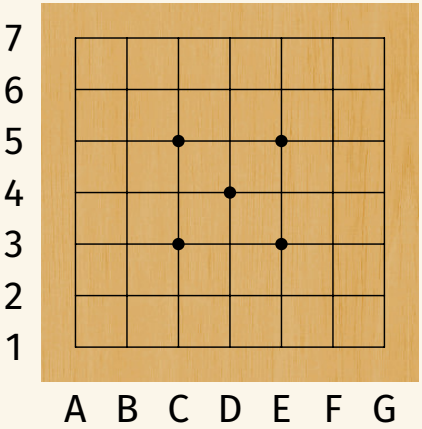
```
\pgfgoset{profile = pgfgo,
  goban = {label = {at = {north, west}}}}

\begin{pgfgolayout}[columns = 2, fit]
  % Without captures: 2 is still there.
  \begin{goban}[sgf size = game]
    \sgf{game}
  \end{goban}
  \begin{goban}[sgf size = game, captures]
    \sgf[from = 7]{game}
  \end{goban}
\end{pgfgolayout}
```



`from`: the first move with a number. `to`: the last one played. After that, `\moves` continues with the turn.

A commented game, move by move



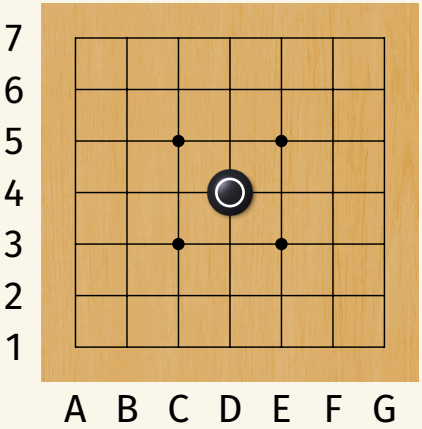
A short game

Black against White, komi 6.5.

Move 0

A game on a 7x7 board, with comments.

A commented game, move by move



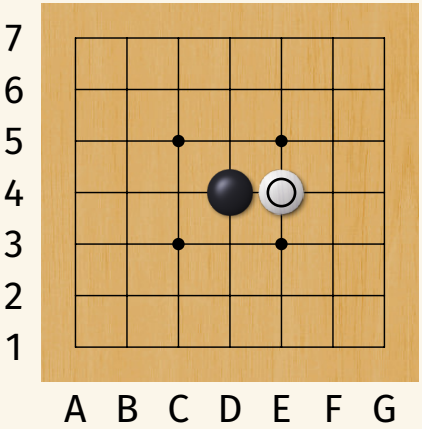
A short game

Black against White, komi 6.5.

Move 1

Black takes the middle line.

A commented game, move by move

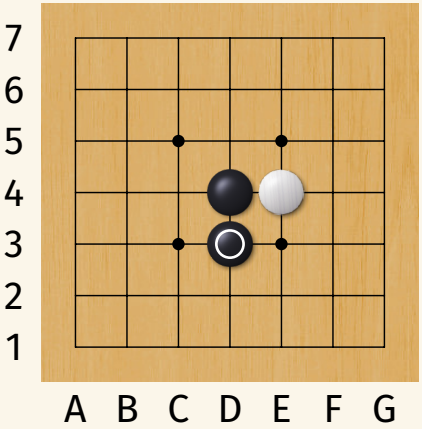


A short game

Black against White, komi 6.5.

Move 2

A commented game, move by move

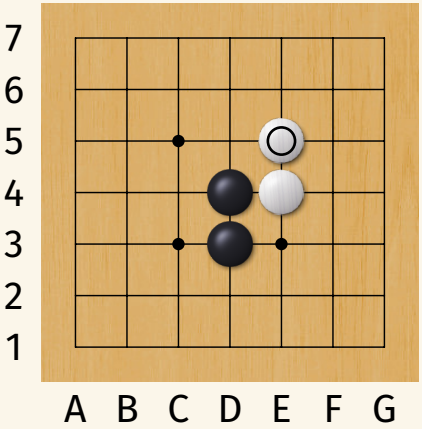


A short game

Black against White, komi 6.5.

Move 3

A commented game, move by move

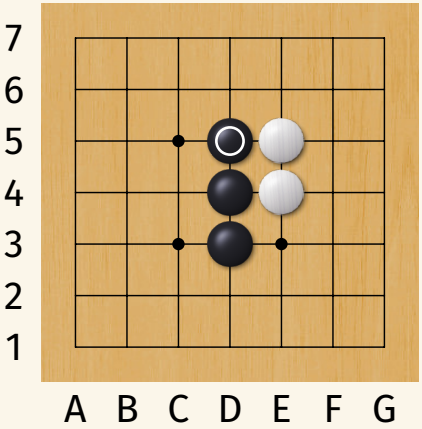


A short game

Black against White, komi 6.5.

Move 4

A commented game, move by move

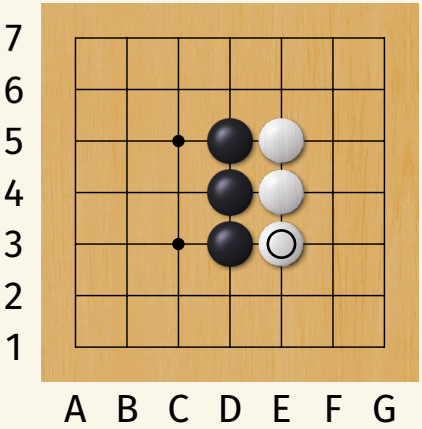


A short game

Black against White, komi 6.5.

Move 5

A commented game, move by move

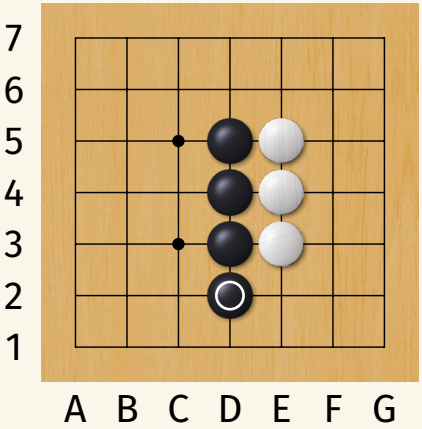


A short game

Black against White, komi 6.5.

Move 6

A commented game, move by move

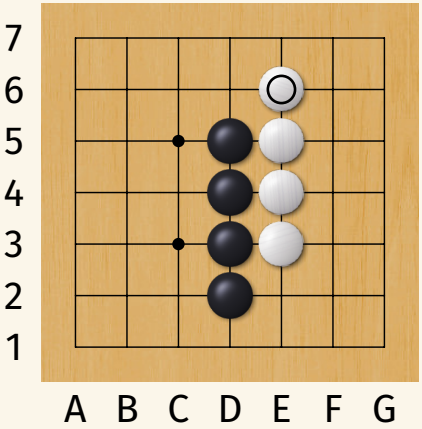


A short game

Black against White, komi 6.5.

Move 7

A commented game, move by move

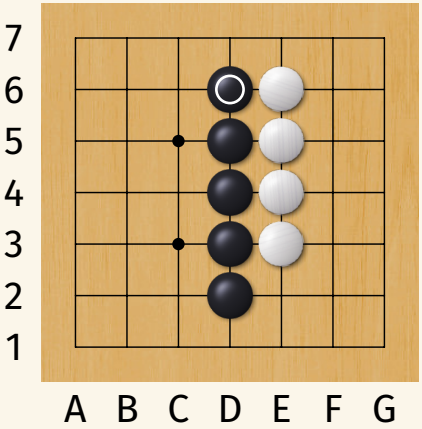


A short game

Black against White, komi 6.5.

Move 8

A commented game, move by move



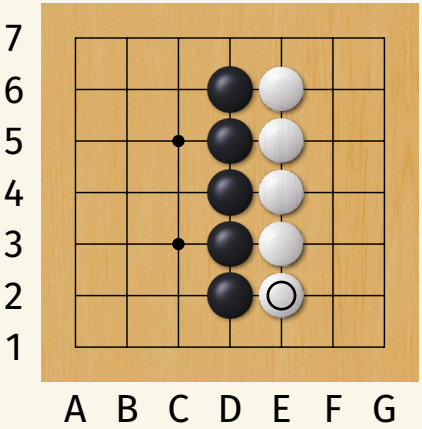
A short game

Black against White, komi 6.5.

Move 9

Black will have three columns, White only two.

A commented game, move by move

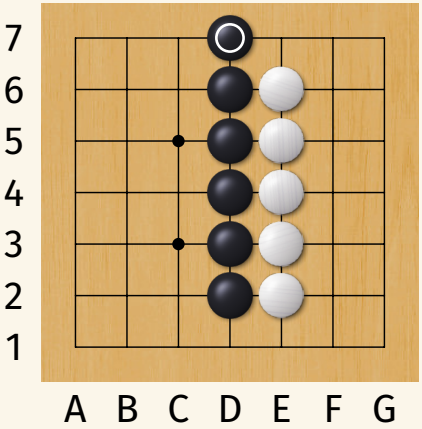


A short game

Black against White, komi 6.5.

Move 10

A commented game, move by move

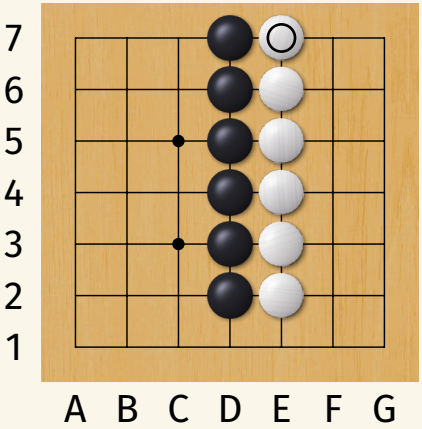


A short game

Black against White, komi 6.5.

Move 11

A commented game, move by move

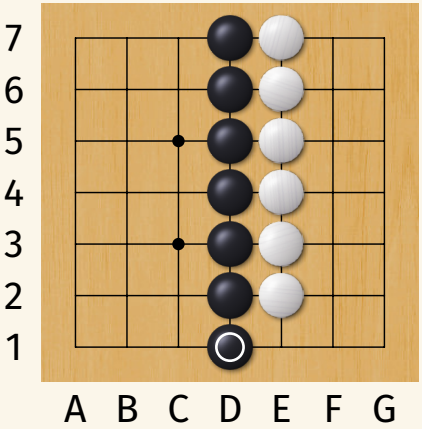


A short game

Black against White, komi 6.5.

Move 12

A commented game, move by move



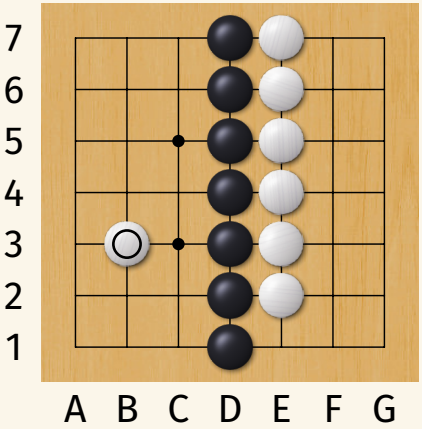
A short game

Black against White, komi 6.5.

Move 13

Tesuji. The last point of the wall.

A commented game, move by move



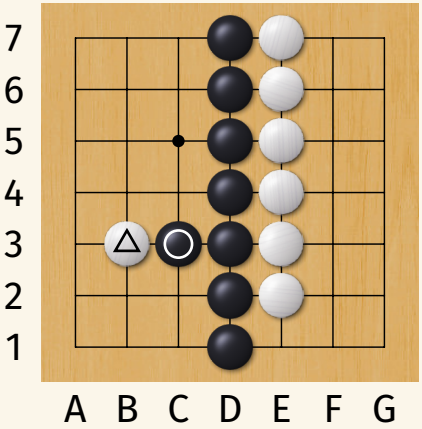
A short game

Black against White, komi 6.5.

Move 14

Bad move. White invades, but there is no room for two eyes.

A commented game, move by move



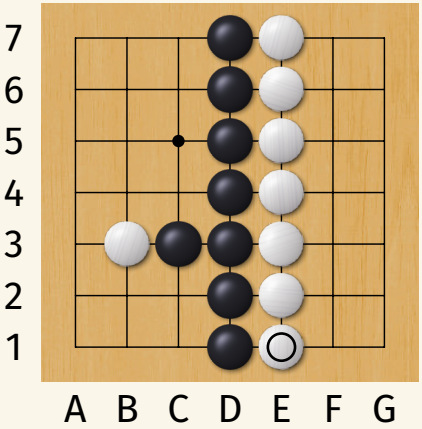
A short game

Black against White, komi 6.5.

Move 15

Black takes the vital point. The white stone is dead.

A commented game, move by move

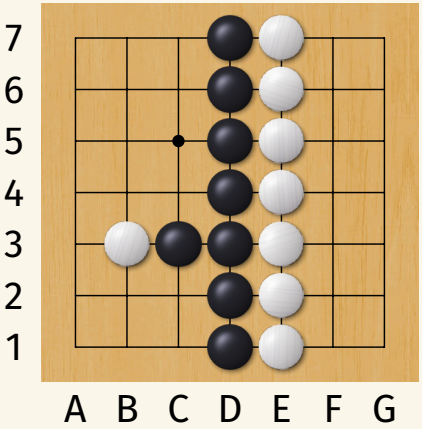


A short game

Black against White, komi 6.5.

Move 16

A commented game, move by move

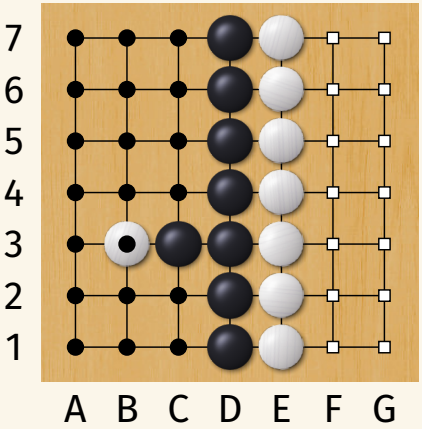


A short game

Black against White, komi 6.5.

Move 17

A commented game, move by move



A short game

Black against White, komi 6.5.

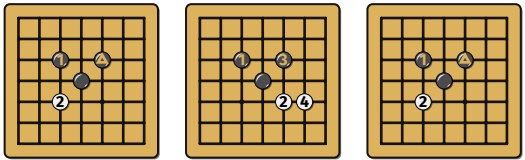
Move 18

Black: 20 points and a prisoner. White: 14 points and 6.5 of komi.

Variations

```
\pgfgoset{profile = pgfgo,
  goban = {sgf size = variations}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}
    \sgf[variation = 2]{variations}
  \end{goban}
  \begin{goban}
    \sgf[variation = 1.2]{variations}
  \end{goban}
  \begin{goban}
    \sgf[variation = {C5, C3}]{variations}
  \end{goban}
\end{pgfgolayout}
```

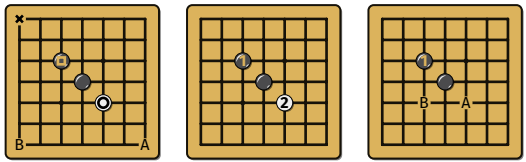


At each fork, the given variation (`1.2`: the first one, then the second), or the moves of the line, as in a dictionary.

The marks of the file

```
\pgfgoset{profile = pgfgo,
  goban = {sgf size = variations}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}
    \sgf[to = 2]{variations}
  \end{goban}
  \begin{goban}
    \sgf[to = 2, marks = false]{variations}
  \end{goban}
  \begin{goban}
    \sgf[to = 1, marks = false,
      variations = children]{variations}
  \end{goban}
\end{pgfgolayout}
```



Triangles, squares, circles, crosses, letters, arrows, lines and territory from the file. `variations` marks with letters the moves that can follow.

Diagrams as in books

```
\sgfdiagrams[step = 4,
  layout = {columns = 3, fit}]
  {game}
```

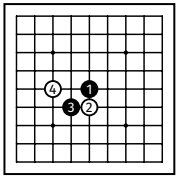


Diagram 1 (1–4)

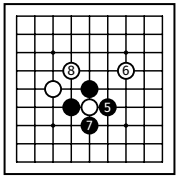
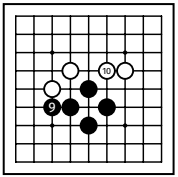


Diagram 2 (5–8)



Dia-
gram 3 (9–10)

One diagram every `step` moves, from the real position: the earlier ones have no number, and each one has its caption and its notes.

Diagrams with their comments

```
\sgfdiagrams[step = 5,
  to = 10, comments,
  goban = {scale = 0.8},
  layout = {columns = 2}]
  {commented}
```

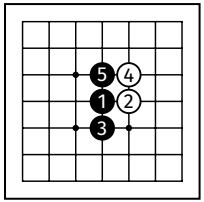


Diagram 1 (1–5)

1: Black takes the middle line.

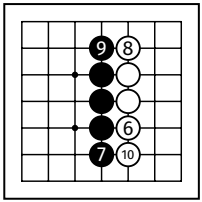


Diagram 2 (6–10)

9: Black will have three columns, White only two.

A whole game: one diagram per slide

```
\pgfgoset{profile = shell}

\sgfdiagrams[step = 50, restart,
  goban = {scale = 0.48},
  layout = {per frame}]{lee}
```

`per frame` puts each diagram on the next slide, with its caption. `restart` numbers each diagram from 1, as in books.

Lee Sedol against AlphaGo, 2016-03-09.

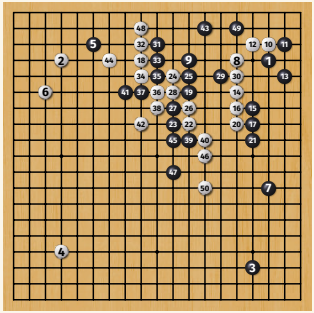


Diagram 1 (1–50)

A whole game: one diagram per slide

```
\pgfgoset{profile = shell}

\sgfdiagrams[step = 50, restart,
  goban = {scale = 0.48},
  layout = {per frame}]{lee}
```

`per frame` puts each diagram on the next slide, with its caption. `restart` numbers each diagram from 1, as in books.

Lee Sedol against AlphaGo, 2016-03-09.

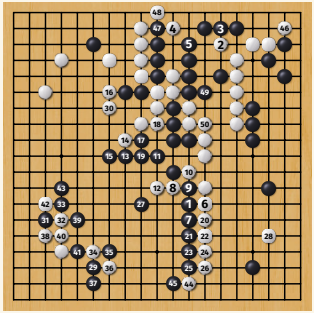


Diagram 2 (51–100)

A whole game: one diagram per slide

```
\pgfgoset{profile = shell}

\sgfdiagrams[step = 50, restart,
  goban = {scale = 0.48},
  layout = {per frame}]{lee}
```

`per frame` puts each diagram on the next slide, with its caption. `restart` numbers each diagram from 1, as in books.

Lee Sedol against AlphaGo, 2016-03-09.

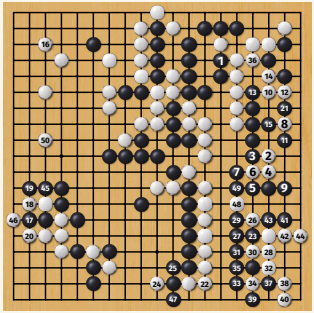


Diagram 3 (101–150)

A whole game: one diagram per slide

```
\pgfgoset{profile = shell}

\sgfdiagrams[step = 50, restart,
  goban = {scale = 0.48},
  layout = {per frame}]{lee}
```

`per frame` puts each diagram on the next slide, with its caption. `restart` numbers each diagram from 1, as in books.

Lee Sedol against AlphaGo, 2016-03-09.

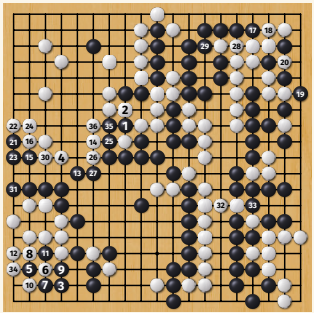
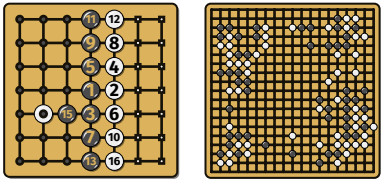


Diagram 4 (151–186)
Result: White wins by resignation

Collections: several games in one file

```
% In the preamble:
% \pgfgodefinesgf{first}{games.sgf}
% \pgfgodefinesgf[game = 2]{second}{games.sgf}
\pgfgodefinesgf{analyzed}{../doc/en/analyzed.sgf}

\pgfgoset{profile = pgfgo}
\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}[sgf size = first]
    \sgf{first}
  \end{goban}
  \begin{goban}[sgf size = second]
    \sgf[to = 100, numbers = false]{second}
  \end{goban}
\end{pgfgolayout}
\sgfinfo{first}{PB} -- \sgfinfo{first}{PW};
\sgfinfo{second}{PB} -- \sgfinfo{second}{PW}.
```



Black – White; Maeda Nobuaki – Segoe Kensaku.

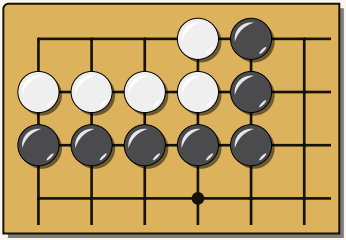
A file with several games: `game` chooses one, in each command or in the name. Each game has its own size.

Problems: the statement, then the solution



```
% First the statement...
\sgfproblem[solutions = false, links,
  goban = {scale = 1.6}]{problem}
% ...and on the next slide,
% the solutions.
\sgfproblem[statement = false, links,
  verdict, comments = false,
  layout = {columns = 4, fit}]
  {problem}
```

`verdict` says whether each variation is right, according to the annotations of the file. `links`: the caption of the statement leads to the solution, and back.



Black to play

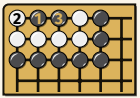
Black to kill.

Problems: the statement, then the solution

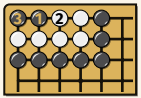


```
% First the statement...
\sgfproblem[solutions = false, links,
  goban = {scale = 1.6}]{problem}
% ...and on the next slide,
% the solutions.
\sgfproblem[statement = false, links,
  verdict, comments = false,
  layout = {columns = 4, fit}]
  {problem}
```

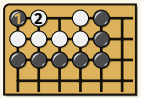
`verdict` says whether each variation is right, according to the annotations of the file. `links`: the caption of the statement leads to the solution, and back.



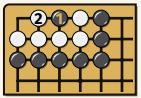
Variation 1 (correct)



Variation 2 (correct)



Variation 3 (wrong)

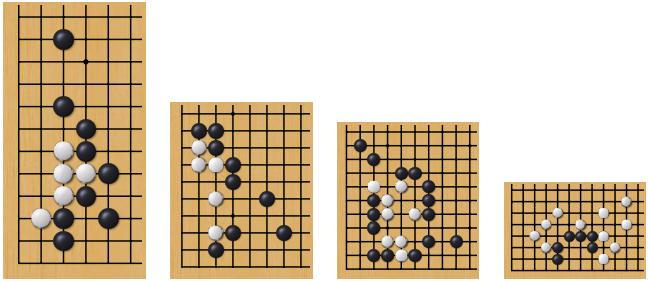


Variation 4

A problem sheet

```
\pgfgoset{profile = shell}

\sgfproblem[
  game = {1:4},
  to play = black,
  comments = false,
  layout = {columns = 4,
    fit, valign =
   bottom}]{hatsuyoron}
```



Problem 1: Black to play

Problem 2: Black to play

Problem 3: Black to play

Problem 4: Black to play

The problems of the *Igo Hatsuyōron* (1713), one per game of the file.

A joseki dictionary

```
\pgfgoset{profile = pgfgo}

\sgfdiagrams[
  variation = {Q16, R14},
  variations = {commented,
    from = 3},
  partial = auto,
  comments = {true, font =
  \tiny},
  caption = {font =
  \scriptsize},
  layout = {columns = 4,
    fit}]{josekis}
```

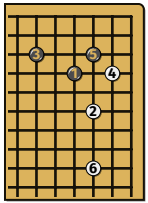


Diagram 1 (1–6)

2: The low approach.
3: A: the knight's move.
4: White slides into the corner.
6: And extends along the side.

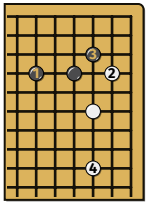


Diagram 1a (variation at 3)

1: B: the one-space jump.

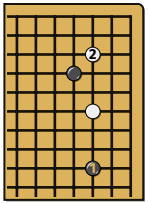


Diagram 1b (variation at 3)

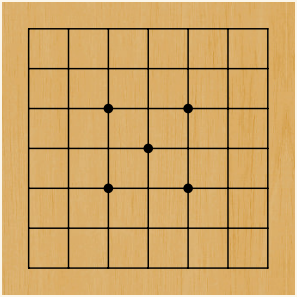
1: C: the low pincer.
2: White invades at the 3-3 point.

The line by its moves; the commented variations from move 3, each one in its own diagram; all with the same part of the board.

\sgfanimation: a game with buttons

```
\usepackage{animate}
...
\sgfanimation[
  comments = {true, height = 3em},
  goban = {captures, scale = 1.2,
    profile = shell},
  animate = {controls, loop}]
  {commented}
```

One frame per move, with the buttons of `animate`. It is seen in Acrobat Reader, Foxit or PDF-XChange; in other viewers, only the first frame.

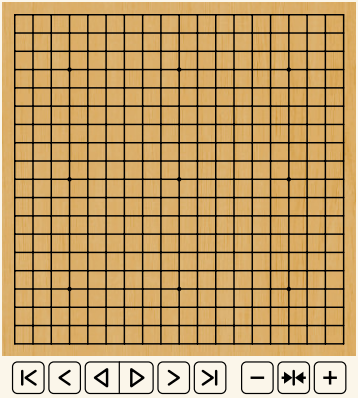


A game on a 7x7 board, with comments.



Animations in layers

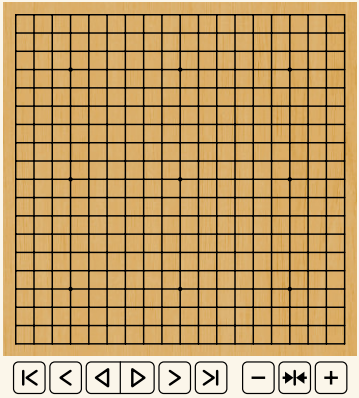
- Without layers, each frame is a whole `goban`: the PDF grows with the square of the moves.



Maeda – Segoe, 1940: 60 frames.

Animations in layers

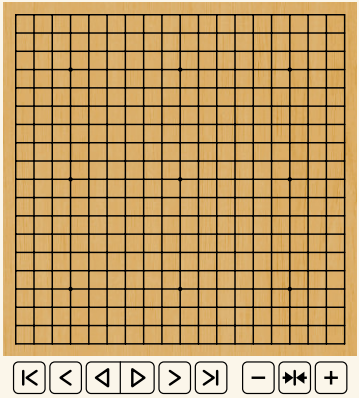
- Without layers, each frame is a whole `goban`: the PDF grows with the square of the moves.
- With `layers` (the initial value), the board is drawn only once, and so is each stone, on a transparency that `animate` shows while the stone is on the board.



Maeda – Segoe, 1940: 60 frames.

Animations in layers

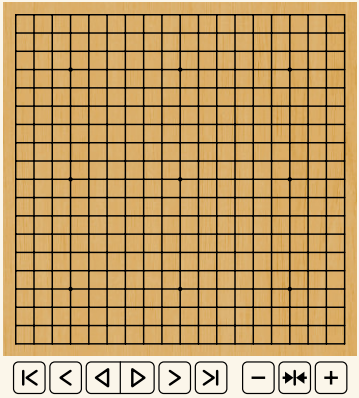
- Without layers, each frame is a whole `goban`: the PDF grows with the square of the moves.
- With `layers` (the initial value), the board is drawn only once, and so is each stone, on a transparency that `animate` shows while the stone is on the board.
- In each frame only what changes is drawn: the mark of the last move, the marks of the file and the comment.



Maeda – Segoe, 1940: 60 frames.

Animations in layers

- Without layers, each frame is a whole `goban`: the PDF grows with the square of the moves.
- With `layers` (the initial value), the board is drawn only once, and so is each stone, on a transparency that `animate` shows while the stone is on the board.
- In each frame only what changes is drawn: the mark of the last move, the marks of the file and the comment.
- A game of 200 moves: less than 1 MB instead of more than 10 MB, and it compiles faster.

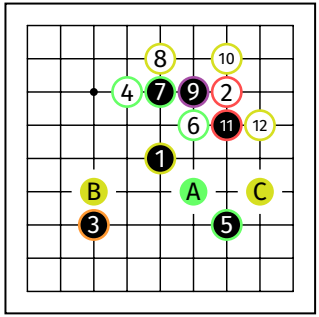


Maeda – Segoe, 1940: 60 frames.

Analyzed games: what the engine proposes

```
% The analysis that KaTrain, Lizzie, q5Go,
% Sabaki and Ogatak save in the file.
\begin{goban}[9]
  \sgf[to = 12, analysis = {colored,
    candidates = 3}]{analyzed}
\end{goban}

\pgfgocandidates
```

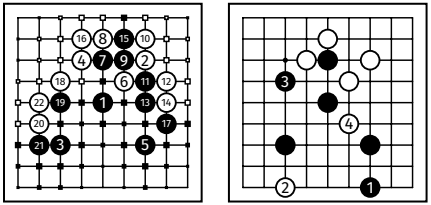


A: 26.7 %, B: 24.8 %, C: 23.0 %

`candidates`: the moves it proposes, with letters. `colored`: from green to violet by what each move loses, as in KaTrain.

Who owns each point and the engine's line

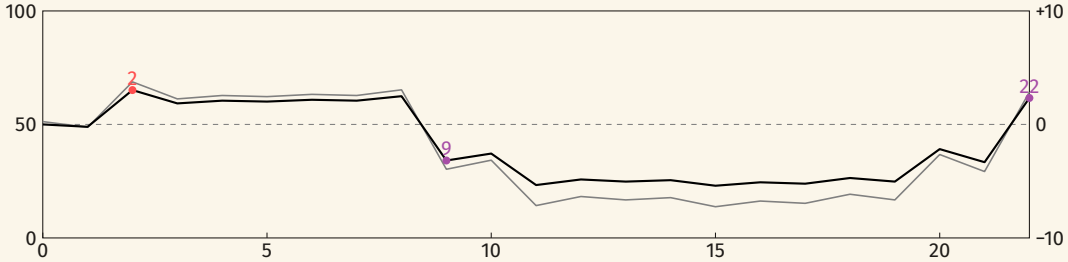
```
\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}[9]
    \sgf[analysis = {ownership}]{analyzed}
  \end{goban}
  \begin{goban}[9, notes]
    \sgf[from = 9, to = 8, analysis =
      {line = {5, from = 1}}]{analyzed}
  \end{goban}
\end{pgfgolayout}
```



`ownership`: a square of the owner's color, on the empty points and on the dead stones. `line`: what the engine expected next.

The graph of the game

```
\sgfgraph[height = 3cm, analysis = {colored, mistakes = 3}]{analyzed}
```



Black's winrate (black, on the left) and the score lead (gray, on the right). The dots are the three moves that lose the most.

A diagram for each mistake

```
\sgfdiagrams[step = 22, layout = {columns = 3, fit},
  analysis = {colored, mistakes = 2, line = 5}]{analyzed}
```

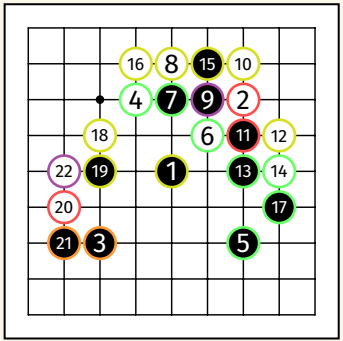


Diagram 1 (1–22)

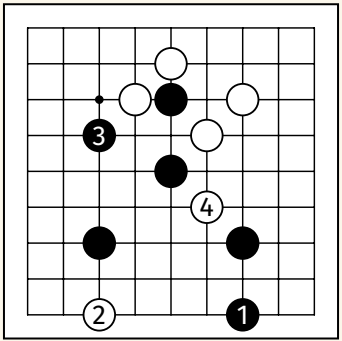


Diagram 1a (AI line at 9)

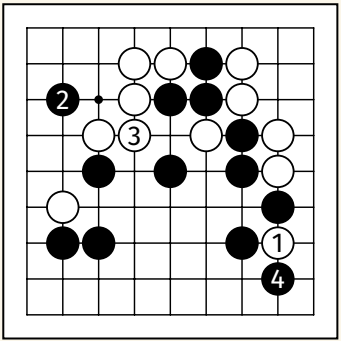
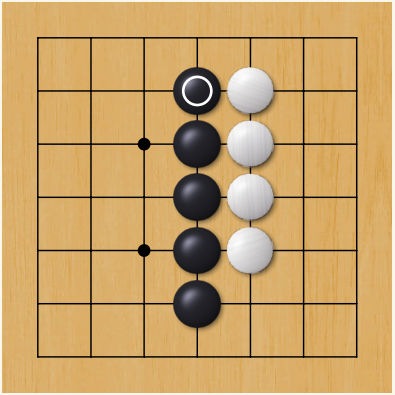


Diagram 1b (AI line at 22)

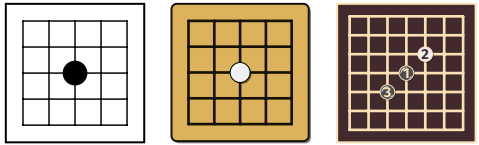
Part 9

Layout



A grid of diagrams

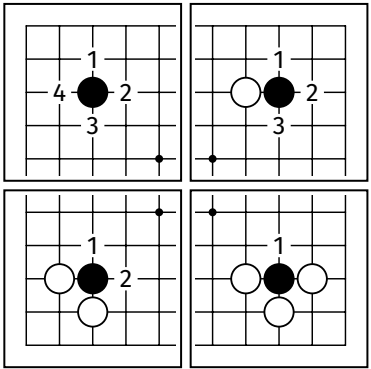
```
\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[5]
    \stones[b]{C3}
  \end{goban}
  \begin{goban}[5, profile = pgfgo]
    \stones[w]{C3}
  \end{goban}
  \begin{goban}[7, profile = wain]
    \moves{D4, E5, C3}
  \end{goban}
\end{pgfgolayout}
```



Columns of the same width; with `fit`, each board is scaled until it fills its own.

width = natural

```
\begin{pgfgolayout}[columns = 2, width = natural,
    column sep = 0.3em, row sep = 0.3em]
  \begin{goban}[partial = A9:E5]
    \stone[b]{C7}
    \marks[sequence]{C8, D7, C6, B7}
  \end{goban}
  \begin{goban}[partial = E9:J5]
    \stone[b]{G7} \stone[w]{F7}
    \marks[sequence]{G8, H7, G6}
  \end{goban}
  \begin{goban}[partial = A1:E5]
    \stone[b]{C3} \stones[w]{B3, C2}
    \marks[sequence]{C4, D3}
  \end{goban}
  \begin{goban}[partial = E1:J5]
    \stone[b]{G3} \stones[w]{F3, G2, H3}
    \marks[sequence]{G4}
  \end{goban}
\end{pgfgolayout}
```



Pages and slides



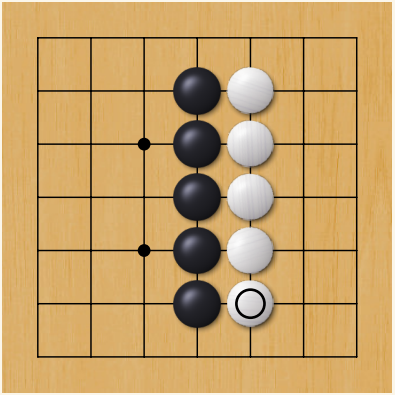
```
% Four diagrams per page.
\sgfdiagrams[step = 50, layout =
  {columns = 2, fit, per page = 4}]{lee}

% One per slide, inside a frame.
\sgfdiagrams[step = 50, layout =
  {per frame}]{lee}
```

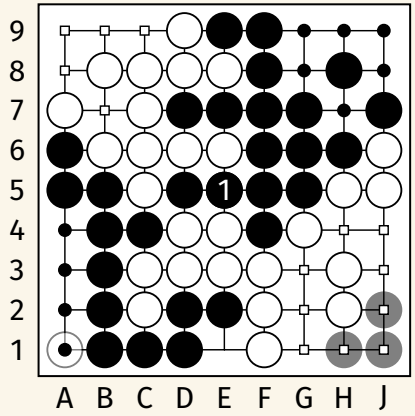
- `per page`: every *n* diagrams, `\newpage`.
- `per frame`: every *n* diagrams, the next slide (it uses `\only<+>`, so it adds to the `\pause` of the frame).
- `halign`: `center`, `left` or `right` inside the column.
- `valign = bottom`: the boards of a row end at the same height; `align captions`: they stay at the top and the captions start on the same line.
- They work in `pgfgolayout` and in the key `layout` of `\sgfdiagrams` and `\sgfproblem`.

Part 10

Profiles and images



Nine profiles



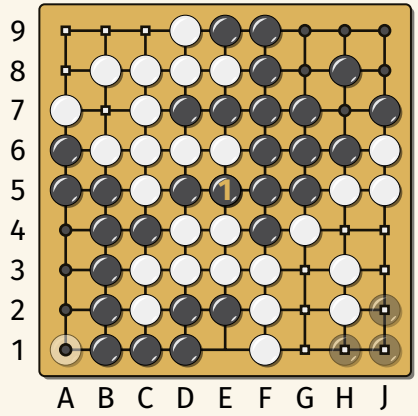
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = default}
```

The initial style: the board and the players go back to it before any profile is applied.

Profile 1 of 9.

Nine profiles



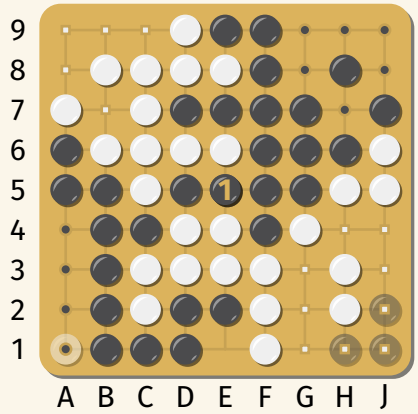
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = pgfgo}
```

The color of the board, with a shadow.

Profile 2 of 9.

Nine profiles



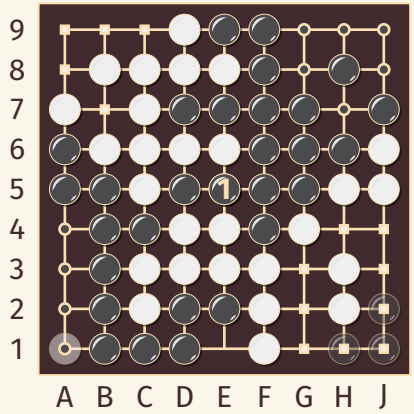
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = pgfgosmooth}
```

Like `pgfgo`, softer.

Profile 3 of 9.

Nine profiles



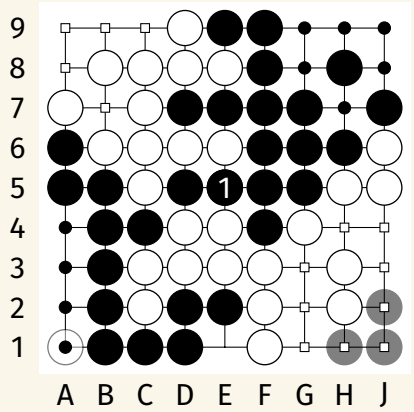
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = wain}
```

The colors of a light wooden board, with shiny stones.

Profile 4 of 9.

Nine profiles



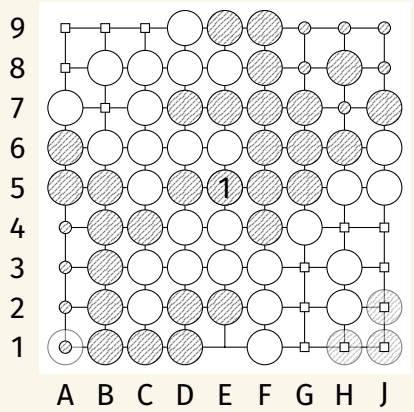
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = simple}
```

For printing: black and white, without ornaments. It also works for views.

Profile 5 of 9.

Nine profiles



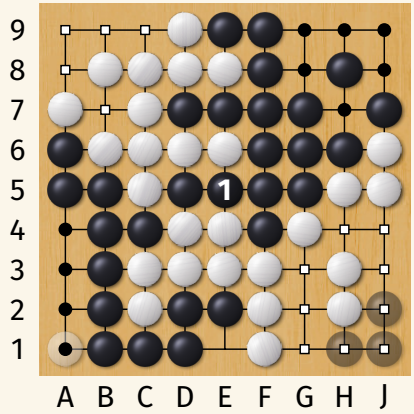
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = draft}
```

Black and white, with hatching where there is a fill: black stones, marks and territory.

Profile 6 of 9.

Nine profiles



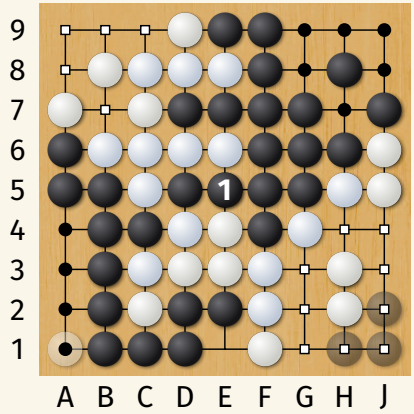
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = shell}
```

Images: wood, slate and five rotated shells, with a soft shadow.

Profile 7 of 9.

Nine profiles



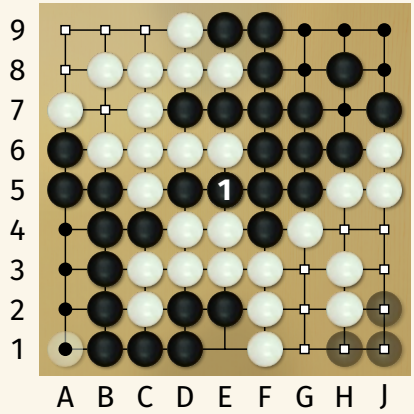
A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = yunzi}
```

Like `shell`, with matte *yunzi* stones.

Profile 8 of 9.

Nine profiles



A profile changes the style of the board and of the three players at once:

```
\pgfgoset{profile = baduktv}
```

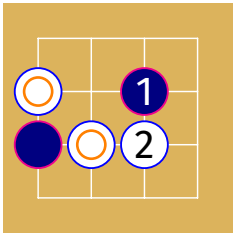
In the style of television broadcasts.

Profile 9 of 9.

The options of each player

```
\pgfgoset{
  goban = {background = {fill = {color = wood},
    line = false}, grid = {line = {color = white}}},
  players = {
    black = {stone = {line = {color = magenta},
      fill = {color = blue!50!black}}},
    white = {stone = {line = {color = blue}},
      mark = {line = {color = orange}}}}}

\begin{goban}[4, scale = 1.6]
  \stone[b]{A2}
  \stones[w, mark = circle]{B2, A3}
  \moves{C3, C2}
\end{goban}
```

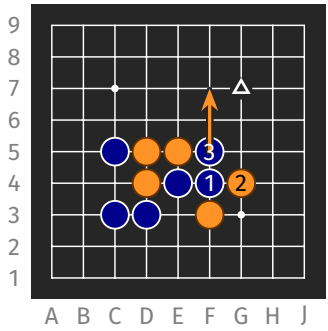


Three players: `black`, `white` and `neutral`; `all` for the three. Each one with its stones, moves and marks.

A profile of your own

```
\pgfgodefineprofile[from = default]{night}{
  goban = {background = {fill = {color = black!85},
      line = false},
    grid = {line = {color = white!60}},
    label = {color = gray},
    arrow = {color = orange!85}},
  neutral = {mark = {line = {color = white!80},
    fill = {color = gray}}},
  black = {stone = {fill = {color = blue!55!black},
    line = {color = white!70}}},
  white = {stone = {fill = {color = orange!85},
    line = {color = orange!40!black}}}}

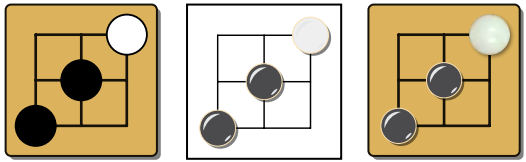
\begin{goban}[9, profile = night, scale = 0.95,
    label = {at = {south, west}}]
  \stones[b]{C3, D3, E4, C5}
  \stones[w]{D4, D5, E5, F3}
  \moves{F4, G4, F5}
  \marks[triangle]{G7}
  \arrow{F5}{F7}
\end{goban}
```



Mixing profiles

```
\pgfgoset{goban = {3}}

\begin{pgfgolayout}[columns = 3, fit]
  % The board of one profile...
  \begin{goban}[goban profile = pgfgo]
    \stones[b]{A1, B2} \stones[w]{C3}
  \end{goban}
  % ...the players of another...
  \begin{goban}[players profile = wain]
    \stones[b]{A1, B2} \stones[w]{C3}
  \end{goban}
  % ...or one player from each.
  \begin{goban}[goban profile = pgfgo,
      players = {
        black = {profile = wain-black},
        white = {profile = baduktv-white}}]
    \stones[b]{A1, B2} \stones[w]{C3}
  \end{goban}
\end{pgfgolayout}
```

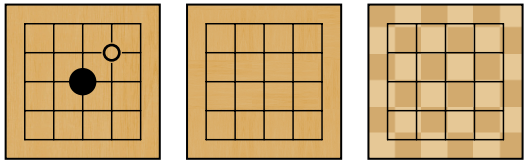


A profile always starts from the initial style: it does not inherit what it does not define.

Images in the background

```
\pgfgoset{goban = {5}}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[background = {fill =
  {image = {
      file = pgfgo-board.jpg}}}]
    \stones[b]{C3} \marks[circle]{D4}
  \end{goban}
  \begin{goban}[background = {fill =
  {image = {
      file = pgfgo-board.jpg, fit = tile,
      width = 1cm, rotate = random}}}]
  \end{goban}
  \begin{goban}[background = {fill =
  {image = {
      file = pgfgo-tile.png,
      fit = tile, width = 0.8cm}}}]
  \end{goban}
\end{pgfgolayout}
```

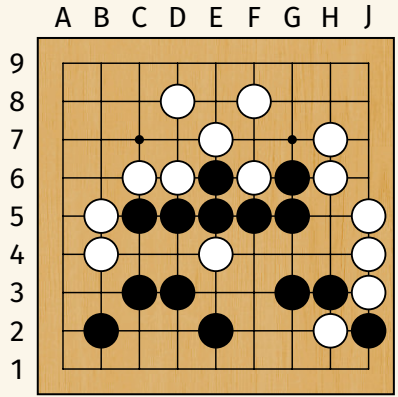


`cover` (initial), `stretch` or `tile`. With `partial`, the corresponding part is seen.

Stones with images, step by step

```
\pgfgoset{
  goban = {background = {fill = {image = {
    file = pgfgo-board.jpg}}}},
  players = {
    all = {stone = {shadow = {true,
      offset = 0.75pt, angle = -45}}},
    black = {stone = {image = {
      file = pgfgo-shell-black.png}}},
    white = {stone = {image = {
      file = {pgfgo-shell-white-1.png, ...,
        pgfgo-shell-white-5.png},
      rotate = -15:15}}}}}
```

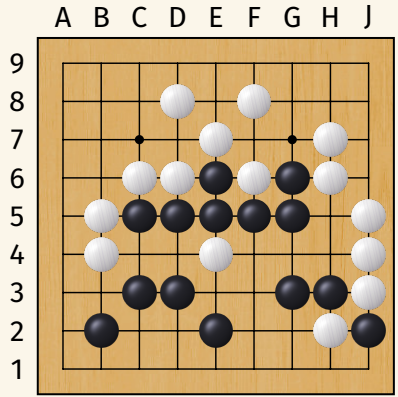
1. The wood in the background.



Stones with images, step by step

```
\pgfgoset{
  goban = {background = {fill = {image = {
    file = pgfgo-board.jpg}}}},
  players = {
    all = {stone = {shadow = {true,
      offset = 0.75pt, angle = -45}}},
    black = {stone = {image = {
      file = pgfgo-shell-black.png}}},
    white = {stone = {image = {
      file = {pgfgo-shell-white-1.png, ...,
        pgfgo-shell-white-5.png},
      rotate = -15:15}}}}}
```

1. The wood in the background.
2. An image for each player.

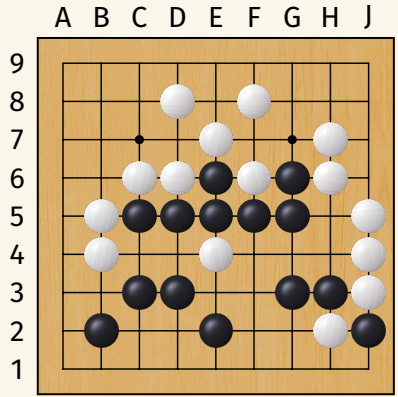


Stones with images, step by step



```
\pgfgoset{
  goban = {background = {fill = {image = {
    file = pgfgo-board.jpg}}}},
  players = {
    all = {stone = {shadow = {true,
      offset = 0.75pt, angle = -45}}},
    black = {stone = {image = {
      file = pgfgo-shell-black.png}}},
    white = {stone = {image = {
      file = {pgfgo-shell-white-1.png, ...,
        pgfgo-shell-white-5.png},
      rotate = -15:15}}}}}
```

1. The wood in the background.
2. An image for each player.
3. Five shells: each point chooses its own.

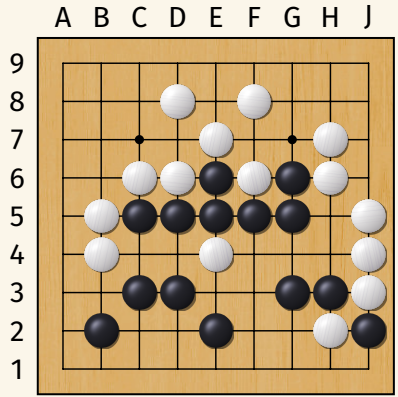


Stones with images, step by step



```
\pgfgoset{
  goban = {background = {fill = {image = {
    file = pgfgo-board.jpg}}}},
  players = {
    all = {stone = {shadow = {true,
      offset = 0.75pt, angle = -45}}},
    black = {stone = {image = {
      file = pgfgo-shell-black.png}}},
    white = {stone = {image = {
      file = {pgfgo-shell-white-1.png, ...,
        pgfgo-shell-white-5.png},
      rotate = -15:15}}}}}
```

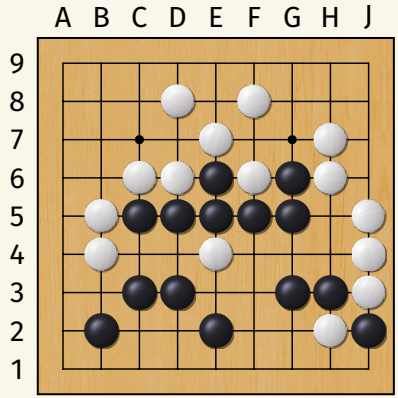
1. The wood in the background.
2. An image for each player.
3. Five shells: each point chooses its own.
4. A shadow with the shape of the image.



Stones with images, step by step



```
\pgfoset{
  goban = {background = {fill = {image = {
    file = pgfgo-board.jpg}}}},
  players = {
    all = {stone = {shadow = {true,
      offset = 0.75pt, angle = -45}}},
    black = {stone = {image = {
      file = pgfgo-shell-black.png}}},
    white = {stone = {image = {
      file = {pgfgo-shell-white-1.png, ...,
        pgfgo-shell-white-5.png},
      rotate = -15:15}}}}}
```

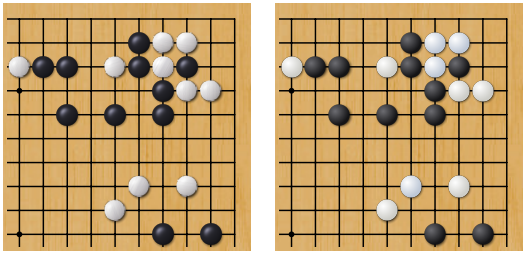


1. The wood in the background.
2. An image for each player.
3. Five shells: each point chooses its own.
4. A shadow with the shape of the image.
5. Each shell rotated, between -15 and 15 degrees.

Ready-made: shell and yunzi

```
\pgfgoset{goban = {sgf size = segoe,
  partial = K10:T19, captures}}

\begin{pgfgolayout}[columns = 2, fit]
  \begin{goban}[profile = shell]
    \sgf[to = 80, numbers = false]{segoe}
  \end{goban}
  \begin{goban}[profile = yunzi]
    \sgf[to = 80, numbers = false]{segoe}
  \end{goban}
\end{pgfgolayout}
```

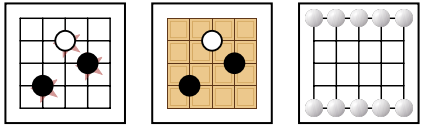


The images come with the package and are used by their name, from any document.

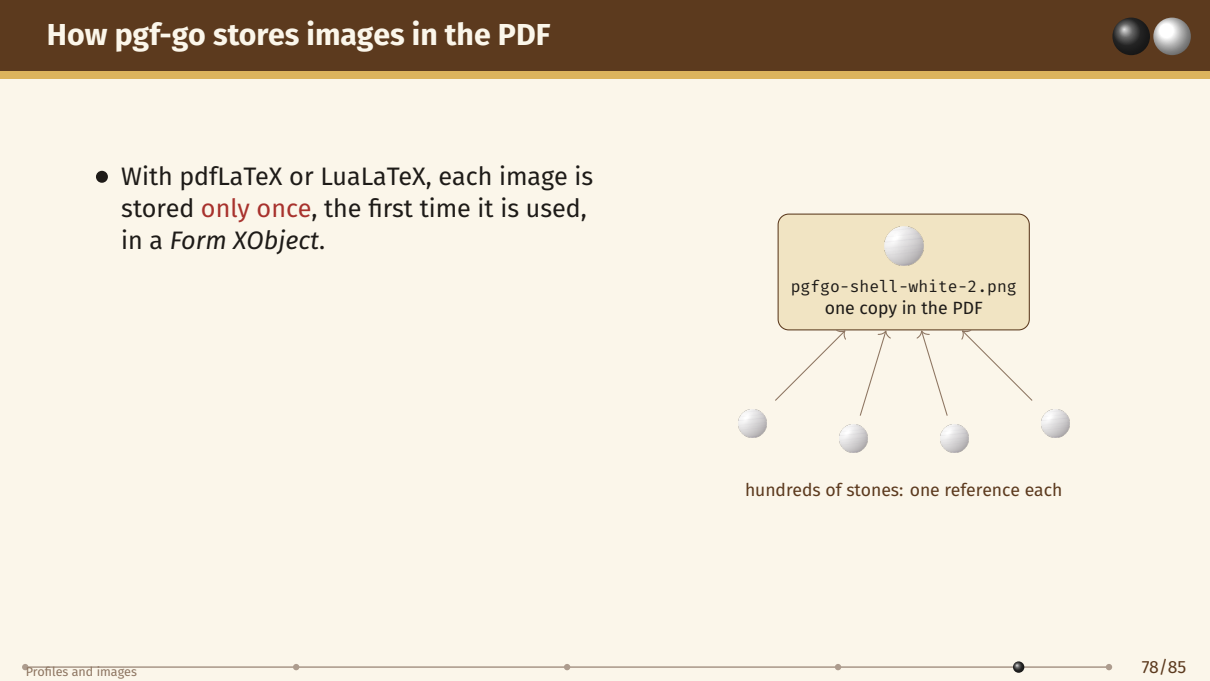
Shadows, hoshi, grid and marks with images

```
\pgfgoset{goban = {5}}
\pgfgodefinemark[image = {file = {
  pgfgo-shell-white-1.png, pgfgo-shell-white-3.png},
  width = 1em}]{shells}

\begin{pgfgolayout}[columns = 3, fit]
  \begin{goban}[stone = {shadow = {true,
      angle = -45, offset = 2pt, color = crimson,
      image = {file = pgfgo-star.pdf,
      width = 16pt, rotate = 18, silhouette}}}]
    \stones[b]{B2, D3} \stone[w]{C4}
  \end{goban}
  \begin{goban}[hoshi = false, grid = {image = {
      file = pgfgo-cell.png, fit = tile}}]
    \stones[b]{B2, D3} \stone[w]{C4}
  \end{goban}
  \begin{goban}[hoshi = {image = {
      file = pgfgo-hoshi.png}, radius = 3pt}]
    \marks[shells]{A:E1, A:E5}
  \end{goban}
\end{pgfgolayout}
```



How pgf-go stores images in the PDF

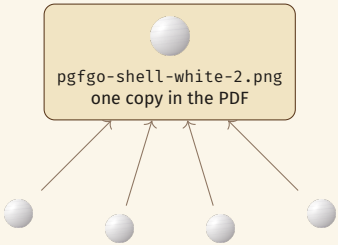


- With pdfLaTeX or LuaLaTeX, each image is stored only once, the first time it is used, in a *Form XObject*.

hundreds of stones: one reference each

How pgf-go stores images in the PDF

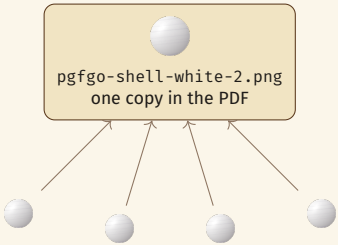
- With pdfLaTeX or LuaLaTeX, each image is stored only once, the first time it is used, in a *Form XObject*.
- After that, each stone, tile or mark that uses it only *names* it: a few bytes.



hundreds of stones: one reference each

How pgf-go stores images in the PDF

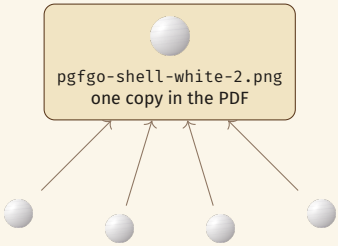
- With pdfLaTeX or LuaLaTeX, each image is stored only once, the first time it is used, in a *Form XObject*.
- After that, each stone, tile or mark that uses it only *names* it: a few bytes.
- Rotating it (`rotate`) does not add another copy, and the shadow with its shape uses the same copy.



hundreds of stones: one reference each

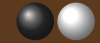
How pgf-go stores images in the PDF

- With pdfLaTeX or LuaLaTeX, each image is stored only once, the first time it is used, in a *Form XObject*.
- After that, each stone, tile or mark that uses it only *names* it: a few bytes.
- Rotating it (`rotate`) does not add another copy, and the shadow with its shape uses the same copy.
- What does add a copy is another size: another `width`, or another `scale`.

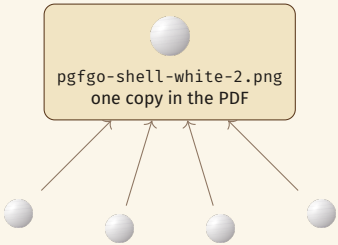


hundreds of stones: one reference each

How pgf-go stores images in the PDF



- With pdfLaTeX or LuaLaTeX, each image is stored only once, the first time it is used, in a *Form XObject*.
- After that, each stone, tile or mark that uses it only *names* it: a few bytes.
- Rotating it (`rotate`) does not add another copy, and the shadow with its shape uses the same copy.
- What does add a copy is another size: another `width`, or another `scale`.
- With XeLaTeX or DVI output, the image is included every time.



hundreds of stones: one reference each

A thousand diagrams, less than 700 KB



```
\pgfgoset{profile = shell}

\begin{pgfgolayout}[columns = 6, fit]
  \foreach \k in {1, ..., 1000} {
    \begin{goban}[9]
      \stones[b]{C3, D4, E5, G3, ...}
      \stones[w]{G7, F4, D6, C7, B4:6}
      \moves{H3, H4, J4}
    \end{goban}}
\end{pgfgolayout}
```

1000 boards of 9×9	
Profile `shell` (images)	285 KB
Profile `pgfgo` (drawn)	615 KB
Pages	21

The wood, the six stones and the shadow are stored once, and each stone only names them: with images, the PDF weighs less than with drawn stones. All the boards have the same scale (`fit`): there are no copies of another size.

What makes a PDF bigger and what does not



It does not make it bigger

- More stones with the same image and the same size.
- Rotating the images, or choosing among several files.
- The shadow with the shape of the image.
- Many marks on a background of tiles: only the tiles the mark touches are redrawn.
- An animation in layers (`layers`).

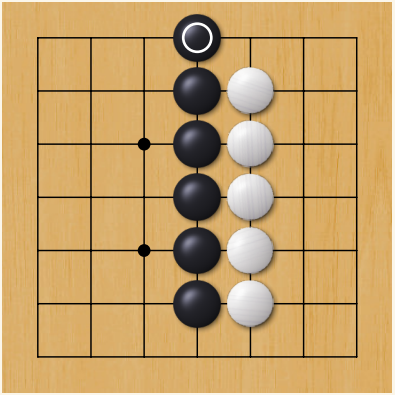
It makes it bigger

- The same image with another size: another `width`, another `scale`, `fit` in different grids.
- Translucent stones (`opacity` less than 1): each one goes in a transparency group.
- Animations with `layers = false`: a whole board per frame.
- XeLaTeX or DVI output with images.

In `\sgfdiagrams` and `\sgfproblem` with `fit`, all the boards use the same scale: the stones have the same size, and each image is stored once.

Part 11

Languages



The texts of the package

```
\pgfgotext{black to play}.
\pgfgotext{very good tesuji}.

\pgfgoset{lang = fr} \pgfgotext{black to play}.

\pgfgoset{lang = de} \pgfgotext{black to play}.

\pgfgoset{lang = eo} \pgfgotext{black to play}.

\pgfgoset{lang = it} \pgfgotext{bad move}.

{\japanese\pgfgoset{lang = ja}
  \pgfgotext{black to play}.}
```

Black to play. Very good tesuji.
Noir joue.
Schwarz am Zug.
Nigro ludas.
Mossa cattiva.
黒番.

Captions, results, annotations and “N at M” notes, in the language of `babel` or `polyglossia` (ten languages), or the one chosen with `lang`.

One diagram in French, another in German

```
\pgfgoset{goban = {scale = 0.65},
  sgf diagrams = {layout = {columns = 2}}}

\pgfgoset{lang = fr}
\sgfdiagrams[from = 1, to = 6,
  step = 6]{game}

\pgfgoset{lang = de}
\sgfproblem[solutions = false]
  {problem}
```

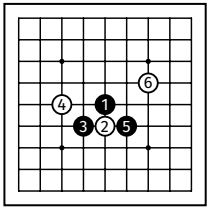
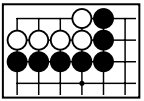


Diagramme 1 (1–6)



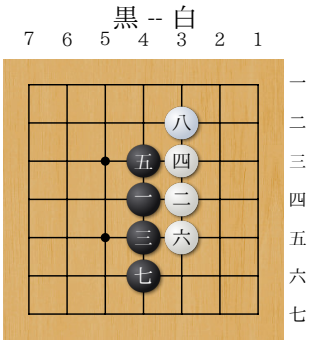
Schwarz am Zug
Black to kill.

The comment of the problem is not translated: it is written that way in the file. Each text is changed with its key or with `\pgfgodefinelanguage`.

A Japanese game

```
{\japanese\sgfinfo{kifu}{PB} -- \sgfinfo{kifu}{PW}}

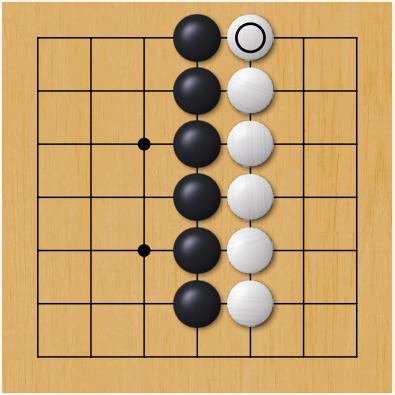
\begin{goban}[sgf size = kifu, profile = yunzi,
    scale = 1.15, label = {at = {north, east},
    style = japanese, font = \kanji},
    move = {label = {format = kanji,
      font = \japanese}}]
  \sgf[to = 8]{kifu}
\end{goban}
```



Japanese-style coordinates and move numbers in kanji. Files in GB2312, Big5, Shift_JIS, EUC-JP or EUC-KR are converted on their own.

Part 12

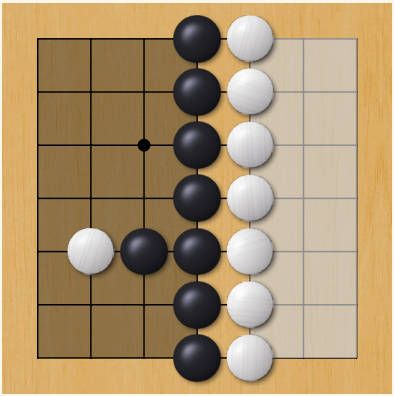
Closing



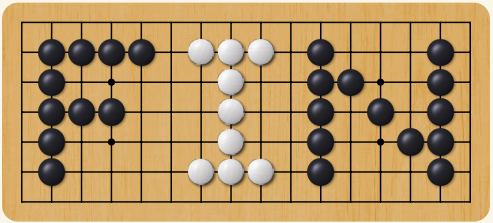
In short

- `goban`, `\stones`, `\moves`, `\marks`: positions by hand, with a syntax to write little.
- `captures`, `notes`, `\territory`, `\pgfgoscore`: the rules.
- `name`, `resume`, `\pgfgoview`, `export`: boards are remembered.
- `\sgf`, `\sgfdiagrams`, `\sgfproblem`, `\sgfanimation`: games, books, problems and animations.
- `analysis`, `\sgfgraph`: the AI analysis saved in the file.
- `profile` and images: the style, in a light PDF.

The manual (`pgf-go-en.pdf`) has fifteen tutorials and the reference of each key; in `doc/examples` there are 40 examples with their code.



Black 21, White 20.5.



Thank you!

```
\stones [b]{E6 -| B2 |- D4}
\stones [w]{[G:J][2, 6], H3:5}
\stones [b]{L2:6 -- P2:6}
```