Go Books File Format

Anders Kierulf

March 16, 2026

This document describes the file format used to include go-specific data and diagrams in SmartGo apps that offer digital books about the game of go.

History

SmartGo Kifu for the iPad pioneered a new way of presenting annotated SGF. Instead of forcing users to replay the game move by move, its book view split the game into diagrams and showed comments next to diagrams. Readers used to such diagrams from go books can often grasp a diagram at a glance. Unlike printed books or PDF documents, readers can replay moves inside the diagram to explore the sequence in detail, and expand a partial diagram to see it in the context of the whole board.

The Go Books app took this a step further, offering complete digital go books. At first, those books were created with extensions of SGF. SGF has always been able to associate comments with moves; the SGF Kifu extensions added richer comments and more sophisticated flow of diagrams and text. However, experience with laying out twenty go books using SGF Kifu highlighted the mismatch between the tree structure of SGF and the more linear structure of a book. Forcing a linear layout into SGF made the process of creating books cumbersome and error prone.

This new format for go books more closely mirrors the structure of a book, significantly improves performance, and is more extensible for the future. It started out as an XML representation (closely resembling HTML for some aspects), but that turned out to be too hard to edit. It ended up as a simpler text-based format, taking cues from text-to-HTML formats like Textile and Markdown, and inheriting properties from SGF.

This file format is work in progress, and will evolve over time, but it has already proven itself in a shipping product (SmartGo One for iPhone, iPad, and Mac) that includes more than 160 go books with a total of more than 150,000 figures and diagrams and 15,000 problems. SmartGo One also has tools for this format, e.g. allowing you to convert annotated SGF to this format and export the resulting books as an interactive EPUB (see appendix 4).

Goals

The goals for the Go Books file format were as follows:

- Make it easy for go players and authors to create, edit, and proofread go books.
- Capture the structure of go books while providing for flexible layout.

- Use a text-based format that makes it easy to split a go game into annotated figures.
- Make it easy to get existing go content into digital form.
- Make it easy to share diagrams between multiple language editions.

This file format is specifically designed for interactive go books that contain mainly text and diagrams; it's not intended for storing go games or as a replacement for SGF.

Example

A short example will demonstrate the main points of the format.

```
::book(#example) title="Example Title" author="Some Body"

::chapter(#mustHaveID)

::h1 break=none #title#

::s1 #author#

::img url="https://www.gobooks.com/images/gobooks-icon.png"

Some text describing this book.

::chapter(#contents)

::h2 Table of Contents

::p href=ch1 Link to First Chapter

::p href=dia2 Link to Diagram 2

::chapter(#ch1)

::h2 First Chapter

Start of a new chapter with ID ch1.

::go mv="D4 F4 H4 K4 M4 O4 Q4"

Simple sequence of moves, starting from the empty board, Black playing first,
alternating players. Go data usually contains all the moves of a game.

Paragraphs are separated by an empty line. Line breaks matter:
This is one line.
This is another line.

::fig at=1 to=3 A=D6

This figure shows moves 1 to 3, with the letter A shown on the board at D6.

::dia at=2 mv=F6H6 ca="Dia. 1"

Diagram with caption __Dia. 1__ showing the position before move 2 in the
game, with an alternative sequence for moves 2 and 3.
```

```
::h3 Minor heading

::fig at=4 to=7 vw=A1T10

Figure showing moves 4 to 7 of the go data. Only a partial view of the board
is shown.

::dia(#dia2) at=5 mv=M6O6Q6 ca="Dia. 2" vw=A1T10

__Dia. 2__ shows a sequence of three moves starting at move 5. They'll be
numbered 1 to 3. It has an ID so it can be referenced later.

This paragraph contains an inline diagram. <dia at=2 mv=O7Q7>When you tap
here, you'll see an alternative sequence.</dia>

::dia base=dia2 at=2 mv=O8Q8 ca_en="Dia. 3 English" ca_ja="Dia. 3 Japanese"
vw=A1T10

__Dia. 3__ is a variation based on Dia. 2. At the second move in the
variation, it shows two alternative moves. Japanese and English will show
different captions.

::dia base=none ab=C3C4D5 aw=D3D4 pl=w mv=E5 a=E6 vw=A1G8
ca="Self-contained diagram"

::p align=center Some centered text with --- em dash, **bold**, __italic__,
**bold __italic__**, and __italic **bold**__.

Go Books can contain translations, and the user can select to have the book
presented in a specific language or multiple languages.
<a href="de">Tap here to switch to German.</a>
<a href="en">Tap here to switch back to English.</a>

::de Dieser Paragraph wird nur auf Deutsch angezeigt.

::en This paragraph is only shown when English is selected.

This example explored just a few features of this format. For more
information check the <a href="https://www.smartgo.com">SmartGo web site</a>.
```

Book structure and layout

The Go Books format specifies a linear flow of content: text, diagrams, and images.

Structure

- Each file contains one book. The file extension is `.gobook`.
- A book consists of chapters.
- Each chapter consists of headings, paragraphs, diagrams, images, and go data.
- Go data consists of board positions and move sequences played at those positions.
- Diagrams are based on go data. Go data can be separate or part of a diagram.

- The paragraphs following a diagram describe that diagram.
- Diagrams can be embedded inside a paragraph, and can be shown by the user on demand. These are called inline diagrams.
- Each paragraph and diagram can have a list of notes attached. A note can be either a paragraph or a diagram. Notes are designed to be stored in a separate file, using the same file format.

Layout

The structure of paragraphs and diagrams determines what will be shown, and it gives clues for the layout, but it doesn't specify exactly how everything will be laid out. The layout depends on device capabilities and orientation, as well as user settings such as font, font size, and multi-column layout. Where possible, diagrams will be placed near associated text, but there is no guarantee that a text will be visible at the same time as its associated diagram.

Zipped book with images

To make it easy to create a single file containing a whole book (including cover image), you can create a zip archive containing the book (with extension `.gobook`) as well as any images referenced in the book (extension `.jpg`, `.png`, or `.gif`), and then rename the file with extension `.gobk` so applications know this is a go book package.

MIME types

The following MIME types should be used:

- **`.gobook:`** application/x-go-book
- **`.gobk:`** application/x-go-gobk

File structure and elements

Go Books files are text files, in utf-8 encoding, no BOM, with Unix/Mac style line endings (LF). Windows style (CRLF) line endings should work, but are not as well tested.

Elements

The file is divided into elements; each element can have various attributes and a value.

Elements are separated from each other by an empty line. Lines containing only spaces or tabs are treated as empty lines. (Thus line breaks within paragraphs indicate line breaks in the text; empty lines indicate paragraph breaks.)

Elements start with a double colon, the element name (with optional languages appended using underscores and optional #ID in parenthesis), followed by attributes in

the form `attributeName=attributeValue`, followed by the value of the element. No spaces are allowed around the equal sign separating attribute and attribute value. For example:

```
::h2_de(#someID) attr1=simpleValue attr2="a b c" Table of contents
```

Attributes are either `attr=valueWithoutSpaces`, or `attr="value with spaces"`. Quotes are necessary when the value contains spaces. Within quotes, backslash and quote characters need to be preceded by a backslash. Tabs and new line can be added using `\t` and `\n`.

If an element doesn't start with a double colon, it is assumed to be a paragraph without any ID or attributes.

ID and languages

In addition to the specific attributes listed for each element type, each element can have an id and languages:

- **`id:`** The ID of an element is used to refer to that element, e.g. a specific heading or diagram. IDs used as link destinations must be unique. IDs used to refer to a base diagram use the most recent element with that ID if duplicated (unique IDs best).
- **`language:`** The language or languages of this element (see details below).

None of the attributes listed for an element are required. The default type if not specified is text.

Element, attribute, and ID names are case sensitive. Element and attribute names all start with an ASCII letter and contain only ASCII letters and digits, plus underscores for languages. ID names consist of Unicode text and can contain underscore, dash, and period, but no other symbols. ID names can't consist entirely of language tags.

Languages

A book can include multiple translations. The reader can select which language or languages to see; only elements that are language-neutral or that are tagged with one of those languages will be shown. The language setting is inherited from the parent element.

In practice, figures and diagrams will be shared among all translations, while each text paragraph will be marked with a specific language.

Each language is identified by a two-letter code: **`cn`** (Simplified Chinese), **`cs`** (Czech), **`de`** (German), **`en`** (English, default), **`es`** (Spanish), **`fr`** (French), **`it`** (Italian), **`ja`** (Japanese), **`ko`** (Korean), **`nl`** (Dutch), **`po`** (Polish; note pl = player), **`pt`** (Portuguese), **`ru`** (Russian), **`tr`** (Turkish), **`tw`** (Traditional Chinese), **`uk`** (Ukrainian), and **`ui`** (the current user interface language).

Multiple languages can be separated by an underscore. For example, a **`book`** element for English and Japanese would be specified as **`book_en_ja`**. Some attributes can also be language-specific.

Note that the language doesn't really specify the language of the text, but rather the book language for which a specific element should be displayed. However, standard text will be inserted using that language.

Go coordinates

Some of the attributes contain lists of coordinates. All coordinates are expressed in standard coordinates (`A1` .. `T19`, letter 'I' skipped), with `A1` as the bottom left corner. In lists of coordinates, optional white space (space, tab, line break) can be used to separate the coordinates.

For lists of moves, `Pass` and `Tenuki` are allowed instead of a point. They differ in that `A1PassB1` will be numbered as 1, 2, whereas `A1TenukiB1` will be numbered as 1, 3, with '2: elsewhere' shown in the list of moves. Pass is useful if you are interested in showing how many moves it takes one player to capture a group rather than a move sequence of both players.

Book

If present, the **`book`** element must be the first non-comment element in the file, and must only occur once. (Note that files designed to be included don't have a **`book`** element.) The **`id`** of the book should be unique; in a library of books, it would typically be the SKU (stock-keeping unit) of the book, e.g. SmartGo uses `sg0006_ki_invin` for "Invincible" from Kiseido and `sg0069_ss_wrk5` for "Workshop Lectures vol. 5" from Slate & Shell.

The id can be used to store information associated with the book, such as current reading position and notes.

Notes

TODO: Notes are supported in the Go Books app, but not in SmartGo One yet. It's not clear whether this is a good way to handle notes; may be changed.

Notes are designed to be stored in a separate file, as a book using this same file format. They are recognized as notes with the **`notes`** attribute. To know which book the notes are for, set the **`id`** of the **`book`** element to match the one of the book. (A book sample and the corresponding full book should use the same **`id`**, so any notes created while reading the sample will carry over to the full book.)

The notes elements should all include the **`loc`** attribute to identify where the notes are attached. Each note should be assigned a unique **`id`** when it's created, to make syncing notes across multiple devices easier.

Book attributes

- **`format:`** (integer) The version of the file format (default: 0). This will be incremented if there are substantial changes to the format that are not backwards-compatible.

- **`revision:`** (integer) The revision of this book (default: 0). Incrementing the revision when the content of the book gets updated allows apps to reload the book as needed. (**Note:** This is not currently used in the Go Books app. Instead, the server knows the length and checksum of each book file, and the app will update the book anytime the checksum changes.)
- **`title, subtitle, author, publisher, copyright:`** (language-specific texts) The main title, subtitle, authors, publisher, and copyright of the book. These texts can be language-specific, e.g. you can specify both **`title_en`** and **`title_ja`** to provide different titles in English and Japanese. Specify the texts as they would be displayed in a list of books, e.g. don't include 'by' in front of author. Use `&` to separate multiple authors. Add N dan after author name for pro players. **Note:** You can embed these in the text using e.g. `#title#`.
- **`fullWidth:`** (0 or 1) Usually diagrams are shown slightly smaller than figures. Setting this forces all diagrams in this book to be shown at full width (default: 0), unless overridden by the chapter or diagram setting. TODO: not currently used in SmartGo One, all diagrams use full width. Might want to change this to a user setting in the app, rather than an attribute of the book.
- **`circleAsDot:`** (0 or 1) Show all circles as gray dots, on both stones and empty points (default: 0). Normally, circles are shown as dots on empty points and circles on stones. (The legacy Go Books app shows all circles as dots.)
- **`notes:`** (0 or 1) Whether these are notes instead of a book (default: 0).
- **`series:`** (text) An identifier of the series, e.g. "Elementary Go Series".
- **`volume:`** (integer) The volume in the series, starting at 1.
- **`tags:`** Keywords associated with this book, typically identifying topics covered in depth and the intended audience, designed to make it easier for readers to find the right book for them. Keywords must be separated by comma and space. For example, Invincible has the tags `"intermediate, pro-games, history"`. The standard tags used by SmartGo are listed in Appendix 1.
- **`games:`** A list of game IDs referenced in this book, separated by comma and space. SmartGo uses GoGoD game IDs.

Chapter

The chapters are represented by **`chapter`** elements. Each chapter contains go data, paragraphs, diagrams, and images.

Note that chapters are a grouping mechanism, making it easy for e.g. a table of contents to refer to a chapter, and can allow apps to navigate by chapter; chapters don't add anything to the content flow.

Chapters must have an ID, and the ID must be unique within the book. The ID for the table of contents chapter should be `"contents"` so that the app can provide navigation to that chapter.

Each chapter is exported as a separate file when creating an EPUB, so chapters should not be too long.

Chapter attributes

- **`fullWidth:`** (0 or 1) Same as for the **`book`** element, but applied to this chapter. — TODO: not currently used in SmartGo One

Go data

Go data is represented by **`go`** elements. Go data contains a starting position, an optional sequence of moves, and optional game information. Go data can be based on other go data. The scope of go data is until the end of the chapter (see also the definition of **`base`** below).

Note that go data does not add any text or visuals to the content flow.

Board position attributes

- **`sz:`** (integer) Board size (square boards only). Defaults to 19.
- **`ab`**, **`aw:`** (point list) List of black or white stones to add to the position. Use **`ab`** to set up any handicap stones. The points in **`ab`** and **`aw`** must be distinct; if added to an existing position, they override that position (e.g. **`ab`** changes a white stone to be black).
- **`pl:`** (`"b"` | `"w"`) Whose turn it is to play at that position (default: black).
- **`koPoint:`** (point) Point that is forbidden because a ko just got played.
- **`mn:`** (integer) Move number of first move. Defaults to 1 for go data, **`dia`**, and **`prb`**; defaults to **`at`** for **`fig`** and **`var`**.

Instead of explicitly describing the board position, a **`go`** element can also reference another go data or diagram element and get its data from the position at a specific move.

- **`base:`** (`ID` | `"none"`) ID of the **`go`** or diagram element that this element is based on. This inherits attributes like board size from the base data. Given an ID, the base is the most recent go data with that ID in the current chapter. For diagrams, the default base depends on the kind of diagram (see further down); set to `none` in order to not base the diagram on any go data. **Note:** The base needs to be defined before it's used, no forward references.
- **`at:`** (integer | `"end"`) Move number (default: starting move number of the **base**, or 1 if no base). Designates the position **before** that move gets played in the sequence; for example, for a sequence of 3 moves, `at=2` specifies the position after move 1, `at=4` specifies the position at the end of the sequence (which can also be specified with `end`). This also updates whose turn it is to play (**`pl`**). The move number is in the numbering scheme of base. Note that passes in the move sequence are not counted.
- **`ae:`** (point list) Stones to remove from this position. Must not overlap **`aw`** or **`ab`**. **Note:** This is only intended for diagrams where the author wants to show the position as it

would have been if some earlier moves had not been played. Do not use it to create arbitrary positions from other positions.

If a position is specified using **`at`**, the **`ab`** and **`aw`** properties can be used to add stones to that position. Again, this should only be used if it reflects the author's intent; for example, a diagram may show the consequence of playing an exchange earlier.

Move sequence attributes

The move sequence can be specified either 1) explicitly with **`mv`** with or without **`at`** or 2) by using **`at`** and **`to`** in combination to designate a range of moves in the data this element is based on. (The two ways of specifying a sequence are mutually exclusive.)

- **`mv:`** (list of: point | `"Pass"` | `"Tenuki"`) Sequence of moves, alternating players, starting with the player given by **`pl`**. `Pass` and `Tenuki` both change whose turn it is to play; `Tenuki` is shown in the diagram caption as 'elsewhere'.
- **`to:`** (integer | `"end"`) Specifies a sequence of moves from the base. This range is inclusive: for example, `at=3 to=5` means moves 3, 4 and 5 are included. Use the same number for **`to`** and **`at`** to include a single move. Use `end` to include moves to the end of the sequence.

When replaying a move sequence, moves that are illegal (e.g. ko) will be treated as legal.

Game info attributes

The game info associated with go data is not added to the content flow, but may be shown by the app. Any game info to appear as text in the book should be added directly using regular paragraphs.

- **`pb`** (black player), **`br`** (black rank), **`pw`** (white player), **`wr`** (white rank), **`km`** (komi), **`ha`** (handicap), **`re`** (result), **`tm`** (time limit), **`dt`** (date), **`pc`** (place), **`ev`** (event), **`ro`** (round), **`us`** (user), **`an`** (commentary), **`cp`** (copyright), **`so`** (source): Like the corresponding properties in SGF.
- **`gn`** (game name): This should match the game's ID in the GoGoD game collection. Each game mentioned in the book should also be included in the **`games`** attribute of the **`book`**.

Display elements

Diagrams, paragraphs, and images constitute display elements that share a few common attributes.

- **`align:`** (`"left"` | `"center"` | `"right"`) Alignment of the diagram, paragraph, or image. Default is `"left"` for paragraphs. If no alignment is specified, diagrams and images are usually centered, but the layout algorithm may e.g. left-align a diagram to leave room for text next to the diagram. (Justified text is not supported because it usually doesn't look good at the narrow column widths that can be shown on mobile devices.)
- **`href:`** ID of an element within this book to navigate to when the user taps this element, a language to switch to, or a normal https link for going to a web site. When the

element has an **href**, all other interactions with this element are disabled. (Within a paragraph, use standard <a> links. See the section on paragraph links for examples.)

- **break:** ("page" | "column" | "line" | "none") Start this element at a new page or column. For paragraphs, line forces it to start on a new line (ends text wrapping around diagrams). Default is page for **h1** and **h2**, column for **h3**, and none for other paragraphs and elements.
- **loc:** (id "+" offset) The location this element is attached to; for notes only. The location is given by the chapter ID plus an offset, e.g. "ch2+5" means the fifth element in the chapter with ID "ch2". In multi-lingual books, successive elements for different languages are not counted when computing the offset, thus adding a language to a book does not invalidate offsets.

Diagrams

A diagram describes a specific board position and move sequence to be shown as an image. In interactive media, the move sequence can be animated.

Diagrams are an extension of go data, and can contain anything that go data can contain. Unlike go data, diagrams are part of the content flow, and can contain markup to be shown in the diagram.

Diagrams are expressed by **dia**, **fig**, **var**, and **prb** elements. The four elements are identical except for their defaults, reflecting their intended usage:

fig: Main figures of a game, based on a go game given as go data, with move numbers reflecting the moves in the game, and the move numbers listed below the figure. Shown at full size.

dia: Diagrams showing alternative move sequences. Move numbering starts at 1. Usually shown at a slightly smaller scale than the figures. (This default setting can be changed with the **fullWidth** attribute.)

var: Variation based on another diagram. By default, continues the move numbering of that diagram. Based on the most recent diagram rather than go data.

prb: Diagrams showing a problem diagram. Problems are shown at full width, and move input will react differently, giving feedback on correct or wrong solutions.

element	base	number	full width	showFromTo	showToPlay
fig	go data	at	1	1	0
dia	go data	1	0	0	0
var	diagram	at	0	0	0
prb	go data	1	1	0	1

Diagram attributes

- **`vw:`** (point point) Rectangular section of the board to be shown. The **`vw`** attribute is inherited from the base diagram and defaults to the full board if there's no base diagram.
- **`a, b, ..., z, A, B, ..., Z:`** (point list) Mark board points with a single letter. For example, `A=K5 b=C4C5` will mark K5 with a capital A and C4 and C5 with a lowercase b.
- **`A1, A2, ..., A19, B1, ..., T1, ..., T19:`** Show text at the point with that coordinate. For example, `B5=23` would show 23 at point B5.

When displaying the diagram, text added to a point overrides the numbers shown from numbering the move sequence. However, while replaying that sequence, the move number may be shown instead of the added text.

- **`tr`** (triangle), **`ma`** (cross), **`sq`** (square), **`rg`** (diamond), **`cr`** (circle), **`tb`** (black area), **`tw`** (white area), **`ln`** (line), **`ar`** (arrow): (point list) List of points for markup on board, like in SGF. For lines and arrows, consecutive pairs of points define the end points. For circles, a dot is shown on empty points, a circle on stones (note the **`circleAsDot`** book attribute for books that need dots on stones).
- **`hideInReplay:`** (point list) List of points at which any markup should be hidden during replay. Letters (a-z, A-Z) on empty points are automatically hidden during replay, except for problems (letters may mark multiple choices, still want to show those after we play a wrong choice). Also, any marks that are on stones played during replay will not be shown, as those are usually used in the caption to refer to moves played under the stones. TODO: The points for hideInReplay usually exactly match the marked points, a switch to hide or show all might be better.
- **`width:`** (`"full"` | `"dia"` | `"half"` | `"third"`) Usually diagrams are shown slightly smaller than figures. Setting this to `full` forces diagrams to be shown at full width. The **`fullWidth`** attribute sets the default for a whole book or chapter. Two `half` or three `third` width diagrams can fit side-by-side. (The alignment of half- and third-width diagrams needs to be set, so either left-right or left-center-right.) TODO: possibly remove dia and fullWidth, simplify?
- **`coord:`** (`"none"` | `"auto"` | `"standard"` | `"sgf"` | `"j1"` | `"j2"` | `"j3"`) Coordinate system to be shown around the board. The default `auto` leaves that setting to the user. For rare cases, authors can force coordinate display off (`none`) or turn on A1 to T19 (`standard`); this overrides the user setting for this diagram only. Other coordinate systems: `sgf`, `j1` (Japanese), `j2` (iroha), and `j3` (Segoe tsumego dictionary).

Caption attributes

- **`ca:`** (language-specific text) The caption to show below the diagram. Simply **`ca`** is language neutral; specify **`ca_en`**, **`ca_ja`**, **`ca_de`** for language-specific captions.
- **`showToPlay:`** (0 | 1) Whether to add the current player in the caption (default 0, except 1 for **`prb`**).

- **`showFromTo:`** (0 | 1) Whether to show move numbers in caption (default 1 for **`fig`** with no caption, 0 for **`dia`**, **`var`**, **`prb`**, and **`fig`** with caption). If there is no move or move sequence, no move numbers are added.
- **`gaps:`** (text | `"auto"` | `"none"`) Moves that can not be shown as numbered stones on the board (because they're on top of other moves or captured stones) are mentioned below the diagram, in a separate caption line. The default is `auto`, causing the program to generate an appropriate text like "12 at 7", "ko: 5, 8", or "9: connects". If that is not good enough, just specify the text to show. Specify `none` to not list move gaps.

Problem attributes

The following attributes only apply to diagrams marked as **`prb`**.

- **`correct:`** (move sequences) Move sequences that are correct to play in this position. Default: If the problem is based on go data that includes a move sequence, and this diagram doesn't show that move sequence, then assume that move sequence is correct.
- **`wrong:`** (move sequences) Move sequences that show what happens when you play the wrong move.

For both right and wrong answers, multiple move sequences can be separated by semicolons. Each move in the move sequence can be followed by text in parenthesis that will be shown when that move has been played. For example:

```
::prb sz=9 pl=b correct="E1E2D1(White is dead.);E1D1E2(Seki.)"
```

TODO: the comments are not currently working in SmartGo One

The value of a **`prb`** diagram can contain text that will be shown at the start of the problem. While playing through the problem interactively, that text will be replaced by any text specified in **`correct`** or **`wrong`**. The text for the user's move will be shown only briefly when answered by a computer move. **Note:** These texts should be short, as they may be shown in limited space above the diagram.

TODO: FUTURE: need problem with choices "A|B|C", "Black|White", etc with correct answer marked

Inline diagrams

A diagram may be embedded inside a paragraph. In that case, it needs to be delimited using XML syntax.

Inline diagrams are based on the diagram referenced in the containing paragraph, thus if no base is set, they refer to the most recent diagram or go data in the flow. For example:

```
See what happens <dia at=2 mv=C4B5D6>if Black plays 2 at 3</dia> instead.
```

In this example, the text "if Black plays 2 at 3" can be tapped to show a diagram with three moves starting at move 2 in the diagram associated with this paragraph. The details of how the diagram is shown will vary on different devices.

The **`vw`** attribute is inherited from the diagram the inline diagram is based on.

Restriction: Inline diagrams can't be used as a base for other diagrams.

Paragraphs

The elements **p**, **h1**, **h2**, **h3**, **h4**, and **h5** are interpreted as in HTML; **s1** and **s2** are used for subtitles. Regular paragraphs (**p**) are the default for any text that doesn't specify an element type. Paragraphs are by default associated with the most recent diagram or go data in the flow.

Paragraphs for particular languages can simply use the language tag as the element. For example, **de** is equivalent to **p_de** to denote a German paragraph.

The following table shows how the headings are intended to be used, and their size and style as implemented in SmartGo:

heading	typical use	scale	align	style	break	spacing in 1/10 em before / after
h1	book title	1.65	center	bold	page	0 / 12
s1	subtitle to **h1**	1.30	center	bold	none	14 / 16
h2	chapter title	1.30	center	bold	page	12 / 8
s2	subtitle to **h2**	1.22	center	bold	none	12 / 12
h3	section title	1.05	left	bold	column	16 / 4
h4	minor section title	1.0	left	bold	none	16 / 4
h5	minor title	1.0	left		none	12 / 4

Paragraph style

The **style** attribute (`"normal"` | `"block"` | `"hanging"` | `"indent"` | `"bullet"` | `"text"`) is used to format paragraphs on a high level. The style is inherited until the end of the chapter, until the next heading (except `text`), or until the next paragraph with style `normal`.

- `block`: Indents the paragraph from both left and right by a standard amount. This can also be set using the **leftIndent** and **rightIndent** attributes.
- `hanging`: Indents all lines of the paragraph except the first by a standard amount. This can also be set using the **hangingIndent** attribute.
- `indent`: Indents the paragraph by the standard amount.

- `bullet:` Looks for tabs in the text of each consecutive bulleted paragraph, finds the maximum amount of indentation, and creates a hanging indent to match. Note that this does not add any bullet characters, it just uses the text up to the tab as a bullet.
- `text:` Separates paragraphs by indenting the first line (by 1 em) instead of adding extra leading, except that the first paragraph after a heading or diagram is not indented. This is more appropriate for long passages of text rather than a mix of text and diagrams. Unlike the other styles, this style is not terminated at headings.
- `normal:` Normal paragraph used to reset inherited styles.

Standard indentation is 2 em (1 em = current font size). For `block`, `hanging`, and `indent` styles, if the paragraph starts with tabs, the paragraph is indented by the number of tabs times the standard indentation.

Translated paragraphs should use the same style settings as the original paragraph, so that turning a language on or off doesn't change the style or spacing.

Paragraph attributes

- **`keepWithNext:`** (0 | 1) Keep this with the next paragraph if possible. Default is 0, except 1 for titles (**`h1`**, **`h2`**, **`h3`**, **`h4`**, **`h5`**) and subtitles (**`s1`**, **`s2`**).
- **`hint:`** (0 | 1) This paragraph is a hint that should be hidden on first reading; the reader can uncover the hint interactively.
- **`ref:`** (`ID` | `"none"`) ID of the diagram that this paragraph is describing. Can be `none`, in which case this paragraph is not associated with any diagram (useful to avoid detecting coordinate references in paragraphs). (Default is the most recent diagram in the flow; the reference is empty at the beginning of a new chapter.)
- **`scale:`** (floating-point number) The font size as a multiple of the current user setting. Regular text uses scale 1.0; headings use a larger scale (see table above).
- **`paraSpacing:`** (integer) The space before and after this paragraph, in 1/10 em. This is inherited until the next chapter. Can be overridden by **`spaceBefore`** and **`spaceAfter`** for specific paragraphs. The default depends on the style: for `bullet` and `hanging`, the spacing is about half the regular paragraph spacing. (In SmartGo, `bullet` has 0.5 em before, 0.2 em after; `hanging` has 0.2 em before, 0.2 em after.)
- **`spaceBefore, spaceAfter:`** (integer) The space before or after the paragraph, in 1/10 em. The space between paragraphs is the maximum of the space after one paragraph and the space before the next paragraph. Headings have default settings for these attributes. To add an empty line of space before a paragraph, use `spaceBefore=8` (the default setting for paragraphs separated by empty space).
- **`firstIndent, hangingIndent, rightIndent:`** (integer) The indentation of the first line, the remaining lines, or the right side, in 1/10 em.
- **`leftIndent:`** (integer) Sets **`firstIndent`** and **`hangingIndent`**.
- **`blockIndent:`** (integer) Sets **`firstIndent`**, **`hangingIndent`**, and **`rightIndent`**.

- **`box:`** (`0` | `1`) Draw a box around this paragraph (default `0`). If consecutive paragraphs are marked this way, a single box will be drawn around all of them. (Restriction: This currently only works well for text that is also using blockIndent.)

Paragraph value

The value of a paragraph is the text in the paragraph. However, a paragraph may also contain `<dia>` elements (where the value of the diagram becomes tappable text). For more on text formatting see below.

As paragraphs are delimited by empty lines, line breaks can be added directly in the paragraph. Line breaks can also be added inline using `\n`, `<br>` or `<br />`.

Rich text in paragraphs

Text can include markup and abbreviations that will be displayed in a richer way.

Text style

****bold text**:** Text surrounded by double asterisks will be shown in **bold**.

__italic text__: Text surrounded by double underscores will be shown in *italic*.

Bold and italic can be nested, e.g. *__italic text with **__bold italic__**__* or ****bold text with *__bold italic__*****. However, bold and italic can't span multiple paragraphs.

As headers (except **`h5`**) are bold, adding italic makes them bold italic; don't add extra bold.

==underlined text==: Text surrounded by double equal signs will be underlined. **Note:** Use of underline is discouraged; they should only be used when better looking stylistic options have been exhausted.

Symbols

Go books often use special symbols, some of which are not in Unicode. You can insert the following symbols into the text:

- `^T` → triangle: △ (**`tr`** attribute).
- `^D` → diamond: ◇ (**`rg`** attribute).
- `^O` → black or white stone (shown as small circle when not followed by b or w).
- `^S` → square: □ (**`sq`** attribute).
- `^X` → cross mark: × (**`ma`** attribute).

Each of these can be followed by a lowercase `b` or `w` to more specifically refer to a black or white stone with such a mark. When possible, these should be shown that way in the text, e.g. `^Tb` should be shown as a black circle with a triangle on it.

Coordinates

A1 to T19 refer to coordinates on the board. If there is an associated diagram, that diagram must have a letter or number at that point. The coordinate in the comment can then be replaced by that letter; for example, “invade at F8” will be changed to “invade at A” if F8 on the board is marked as A in the diagram. (If there is no currently active diagram, i.e. `ref=none`, no coordinate substitution takes place.)

There is no exception for coordinates B1 to B19. Informal game comments sometimes use B1, W2, B3 to refer to moves Black 1, White 2, Black 3, but in the context of go books, these will always be interpreted as coordinates.

Typography

Text in paragraphs can use simple typography; go books apps must improve the typography to make it easier to read and more like a high-quality book:

- Straight quotation marks (single & double) → curly quotation marks. (These may vary by user language.)
- Double hyphens → en dash.
- Triple hyphens → em dash.
- Triple periods → ellipsis.
- Double spaces → single space. (Triple spaces are not changed.)
- Hyphen between two digits → en dash.
- Hyphen between two spaces or at end of paragraph → en dash.

Early implementations also converted (c) to ©, but that could conflict with actual text, so use the Unicode copyright symbol directly in the text instead.

Links

Tappable links can be embedded in the text using HTML syntax:

```
<a href="https://www.smartgo.com">SmartGo web site</a>
<a href="mailto:support@smartgo.com">Contact support</a>
```

You can link directly to a given chapter, paragraph, or diagram by linking to the ID of that element. Note that IDs used as link destinations need to be unique. If an ID is not found, the link should be disabled and e.g. shown in gray (this can be used in the table of contents for a book sample).

```
<a href="ch2">Tap here to go to chapter 2.</a>
```

If the link consists of a language tag, the book language is switched to that language:

```
<a href="ja">Tap here to switch to Japanese.</a>
::p href=en_de align=center English & German / Englisch & Deutsch
```

Image bullets

Images can be used as bullets. Like other bulleted paragraphs, the image is measured and consecutive bulleted paragraphs are indented by the same amount. For example:

```
::p style=bullet <img url="image.png" href="https://smartgo.com"></img>\tSome paragraph text.
```

TODO: this is only used for help?? check where used

Standard Text

A number of specific text strings within hash tags will be replaced by text in the language of the current paragraph. For example, #btp# will be replaced by “Black to play.” if the element is marked as English, by “A Noir de jouer.” if it’s marked as French, and by the corresponding text in the current user interface language if it’s marked as language neutral.

The main purpose of these standard texts is to define language-independent go problems. The following texts are currently defined; ^ needs to be replaced by `b` for Black or `w` for White:

```
#^tp#  : “^ to play.”
#^tn#  : “^ to respond.”
#^tk#  : “^ to kill.”
#^tl#  : “^ to live.”
#^tc#  : “^ to capture.”
#^tf#  : “^ to fight.”
#^tu#  : “^ to cut.”
#^to#  : “^ to connect.”
#^tx#  : “^ to extend.”
#^tr#  : “^ to reduce.”
#^ti#  : “^ to invade.”
#^ta#  : “^ to attack.”
#^td#  : “^ to defend.”
#^te#  : “^ to escape.”
#^ts#  : “^ to save the marked stones.”
#^tw#  : “^ to win.”
#^twf# : “^ to win the fight.”
#^twd# : “^ to draw or win.”
#^tgk# : “^ to get a ko.”
#^tgr# : “^ to get the best result.”
#^tpe# : “^ to play the best endgame.”
#^tms# : “^ to make shape.”

#^do#  : “What can ^ do?”
#^wi#  : “^ wins.”

#^he#  : “^ has escaped.”
#^hnk# : “^ has no ko threat.”

#^id#  : “^ is dead.”
#^ia#  : “^ is alive.”
#^ias# : “^ is alive in sente.”
#^iddk#: “^ is dead in double ko.”
#^iadk#: “^ is alive in double ko.”
```

```
#fe#   : “False eye.”
#ko#   : “Ko.”
#ok#   : “Only ko, not good enough.”
#os#   : “Only seki, not good enough.”
#se#   : “Seki.”
#sn#   : “Snapback.”
```

The following are intended for use in titles of problem books, e.g. “#Problem# 17”:

```
#Problem# : “Problem”
#Answer#  : “Answer”
#Solution#: “Solution”
```

The special strings above work in paragraphs as well as captions. In captions, the ^ gets replaced by the current player. Standard texts with a language tag appended get replaced with the string in that specific language (e.g. `#wtms_de#` or `#btw_ja#`).

The special strings `#app#` and `#ver#` get replaced by the name and version number of the current app. The strings `#title#`, `#subtitle#`, `#author#`, `#publisher#` and `#copyright#` can be used to insert the corresponding attribute from the book element, in the right language. Game info from the go data associated with a paragraph can be inserted in the text using `#pb#`, `#br#`, `#pw#`, `#wr#`, `#km#`, `#ha#`, `#re#`, `#tm#`, `#dt#`, `#pc#`, `#ev#`, `#ro#`, `#us#`, `#an#`, `#cp#`, `#so#`. (These substitutions only work in paragraphs for now, not in captions.)

Images

Images can be inserted in the flow using the **`img`** element (extension `.jpg`, `.png`, or `.gif`):

```
::img url="https://smartgo.com/img/icon_smartgo_kifu.png" scale=0.75
```

Image attributes

- **`url:`** The image to insert, either the file name of an image that’s included with the `.gobk` package, or an https link to an external image. Default: an **`img`** without a url shows the cover.
- **`scale:`** (floating-point number) Normally, the image will be drawn either at its natural size, or filling the width of the column, whichever is smaller. This would be the default (1.0). A smaller value of scale can be used to reduce the image size.
- **`ca:`** (language-specific text) Caption (as mentioned above for diagrams).
- **`border:`** (0 | 1) Whether to draw a thin dark border around the image. Defaults to 1 for jpg images, otherwise 0.
- **`shadow:`** (0 | 1) Whether to draw a shadow behind the image. Defaults to 1 for jpg images, otherwise 0.
- **`width:`** (`"full"` | `"half"` | `"third"`) Sets the image at a specific width instead of full width (like diagrams). Scale is computed off that width.

Buy book button

A button for buying the current book can be inserted using the **`buy`** element. The interpretation of this element is app-specific. For example, in SmartGo, this may show up as BUY BOOK if the book is available for purchase, and as PURCHASED if the book has already been bought.

Attributes

- **`sku:`** The SKU (stock-keeping unit) of the book to buy. Default: ID of this book.

Include standard chapters

Text used in several books can be included with the **`include`** statement:

```
::include url="about.gobook"
```

This includes all the elements from the given url as if it was directly inserted in the text. (This is currently limited to `"about.gobook"`, `"buy.gobook"`, `"kiseido.gobook"`, `"brettstein.gobook"`, and “`fairbairn.gobook"`, which are standard sections included with SmartGo.)

Note that the **`include`** statement itself must not define any ID; the included elements may define IDs.

Attributes

- **`url:`** The text to insert. For now, this can only be a file name that SmartGo knows about. TODO: SmartGo should be able to include files in the same folder, just like images

Comments

Comments are elements that will be read and written, but don’t change the book in any way. Comments have a value but no attributes. Some examples:

```
::c This is a comment.

::c ::include url="about.gobook" This include is commented out.

::c attr=value ERROR: comments can’t have attributes.
```

Comments are used to extract a sample from a full book; see appendix 3.

Appendix 1: Tags

The **`tags`** in the **`book`** element are used to classify its level and main topics.

Level

The skill level a book is aimed at. These tags are exclusive, and should denote the minimum level at which readers would benefit from this book.

- `introductory`: For players who know nothing about the game. Introduces you to the rules and basic concepts.
- `beginner`: For players who have just started with the game and have read an introductory book. Good second books to learn more about go.
- `elementary`: Double-digit kyu players (20k-10k).
- `intermediate`: Single-digit kyu players (9k-1k).
- `advanced`: Dan players.
- `any-level`: Suitable for players of any rank.

Content

The main topics of the book – each book may contain several of these tags.

- `rules`: The rules of go, introductory content.
- `basics`: The basic principles and techniques of the game.
- `fundamentals`: Fundamental knowledge and principles applicable to different parts of the game.
- `pro-games`: Analysis of professional games.
- `amateur-games`: Analysis of amateur games.
- `problems`: Collection of problems.
- `strategy`: Larger-scale strategy and concepts such as aji and thickness.
- `history`: Historical background information surrounding the games.
- `other`: Some other content (not directly go related).

Game phase

- `opening`: Discussion of early moves of the game, until middle-game fighting starts.
- `joseki`: Standard move sequences in the corner.
- `sanrensei`: Opening pattern with three star-points on a side.
- `handicap`: Special strategies for playing games with or against handicap stones.
- `middle-game`: Middle portion of the game, fighting for survival of groups.
- `fight`: Aggressive strategies during the middle game.

- `endgame:` Final part of the game, finalizing the boundaries between groups.

Various tactics

- `tactics:` Local, small-scale fighting.
- `life-and-death:` Tactics of making groups alive and killing opponent groups.
- `tesuji:` Tactics of locally sharp moves.
- `semeai:` Capturing race.
- `sabaki:` Creating a light shape within the opponent's area.

Special topics

- `ko:` Situations that involve ko (rule that avoids infinite repetition).
- `shape:` Standard patterns that occur in many games.
- `9x9:` Playing on 9×9 board.
- `post-ai:` Based on knowledge gained after AlphaGo and other AIs came on the scene.

Availability

- `exclusive:` Books that are only available from SmartGo.
- `included:` Included for free in SmartGo.
- `out-of-print:` Books that are no longer available as printed books.

Appendix 2: Conventions for reading/writing gobook format

Apps that read or write gobook format should adhere to the following conventions. This minimizes formatting changes that make it harder to review differences.

Reading

- Lines starting with a double colon should be interpreted as the start of a new element, even without a preceding empty line.
- In attributes, accept lowercase standard coordinates as well as lowercase coordinates in SGF format (but not in paragraphs).

Writing

- Write all points using standard uppercase coordinates.
- Write three empty lines before **`chapter`**, two empty lines before headings (**`h1`** – **`h5`**, **`s1`**, **`s2`**), and one empty line before all other elements.
- Write attributes in the same order as they were read. Preserve whether an attribute starts on a new line or not.

- Always use quotes around **`ca`** and **`url`** attributes, even if not strictly required. Also use quotes around values that contain characters other than ASCII letters, digits, underscore, dash, plus sign, period, or semicolon.
- Start all **`book`** and game info attributes on a new line.
- Always write **`pl`** attribute for go data that specifies a position.

Appendix 3: Automatic creation of book samples

The books in SmartGo are marked up to enable automatic extraction of book samples from the full book. The markup defines where the sample starts and ends, and may define elements that are only in the sample.

In order to keep the book file valid even while containing hints about where the sample starts and ends, all the sample hints are in comments.

```
::c sample-start
::c sample-end
```

Going through the file element by element, each element is included in the sample if sample mode is on. Sample mode is on by default, and `sample-start` and `sample-end` turn sample mode on and off.

```
::c sample-only
```

If a line is starts with `::c sample-only`, the rest of that line (minus the space) will be inserted into the sample (regardless of the current mode).

- TODO: add sample as a book attribute, and this gets written when creating the sample? better than using the file name to distinguish sample?

Appendix 4: Go books format support in SmartGo One

SmartGo One for iPhone, iPad, and Mac provides the best support for the Go Books format.

- In My Files, files with .gobook or .gobk extension are shown as go books. Images in the same folder as the .gobook file will be properly included in the book.
- Tap and hold on an SGF file, then choose Create Book to create a book based on the game or game collection in that SGF file. In particular, annotated games with variations are converted into appropriate figures, diagrams, and text. This is a great way to create go data in the right format, and get a good start on a book.
- You can also view a game as a book by tapping on the More menu in the bottom left of the board, then tap the book icon in the View row, and it will be shown as a book. You can then share and select GOBOOK as the format.
- Books can be exported as EPUB, either interactive (replay moves, solve problems) or static (for improved compatibility with various EPUB readers).

- On Mac and iPad, you can turn on Settings > Layout & Appearance > Add Go Book Menu Items. This adds Edit > Copy as Go Book and Copy Position as Go Book, which can be useful when creating books.

Appendix 5: Possible future additions

- Make sure all the game info properties of SGF are covered: ru (rules, conflicts with Russian), bt, wt (team), gc (game comment)
- Allow a whole diagram to be duplicated inline by reference (`clone` attribute).
- Book and chapter attribute `shortTitle` to avoid truncation of book and chapter titles (e.g. in navigation controls).
- Diagram attribute `header` to add a header above the board, in addition to the caption below.
- More control over text: Serif/sans-serif fonts, small caps, etc.
- More general default settings for book. (The `fullWidth` attribute captures the most common case for now.)
- Allow moves to be marked with ! and ? like in SGF. Bad move (?) most important as it could help with problem sequences. (Richard Hunter: As previously mentioned, I think ? means questionable not definitely bad. I think bad/wrong should be X and correct/good should be check/tick mark ✓. I use ! for tesuji often deeper in the move sequence but sometimes first.)
- Allow positions to be marked with B+, W+, B=W like in SGF.
- Allow katakana for labels on the board.
- Vertical writing for Japanese.
- Allow full answer trees in problem diagrams instead of move sequences.
- For symbols, show the symbols using the Go font, and show regular expression for symbols.

TODO: add appendix on creating a book to be included in SmartGo One:

- tags, games, sample directives
- buy button as sample only
- provide example

Feedback welcome

Thanks for valuable feedback from everybody who has worked with early versions of this format, in particular John Mifsud, Richard Hunter, and Marek Jasovsky.

The newest version of this file can be found at:

https://smartgo.com/pdf/gobookformat.pdf

This is work in progress. Please send your thoughts to anders@smartgo.com.

SGF definition

https://www.red-bean.com/sgf/