

Carnatic Music Notation Typesetter (v1.4)

Synopsis

The *Carnatic Music Notation Typesetter* is a web-based application that allows you to create high quality notation sheets for carnatic music. It can generate notations in English, and if appropriate fonts are installed on your computer, it can even generate notation sheets in the following Indic languages: Sanskrit, Telugu, Tamil and Kannada.

Browser Support Firefox is currently the preferred browser to run the typesetter. However, typesetter should work on Internet Explorer and Opera. As indicated above, due to the typesetter being in Beta release, you may still run into problems on these browsers. It is the intention of the author(s) to support the typesetter on Firefox, Opera, IE and Safari (on Mac).

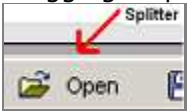
The typesetter requires you to specify the notation for the song to be notated in a specific input format that is simple, and easy to use. The application converts the input into a graphical layout similar to those you see in carnatic music books. This manual describes the input format in detail.

When you open the typesetter application, it should appear as the picture on the left.

The display is divided into two panes

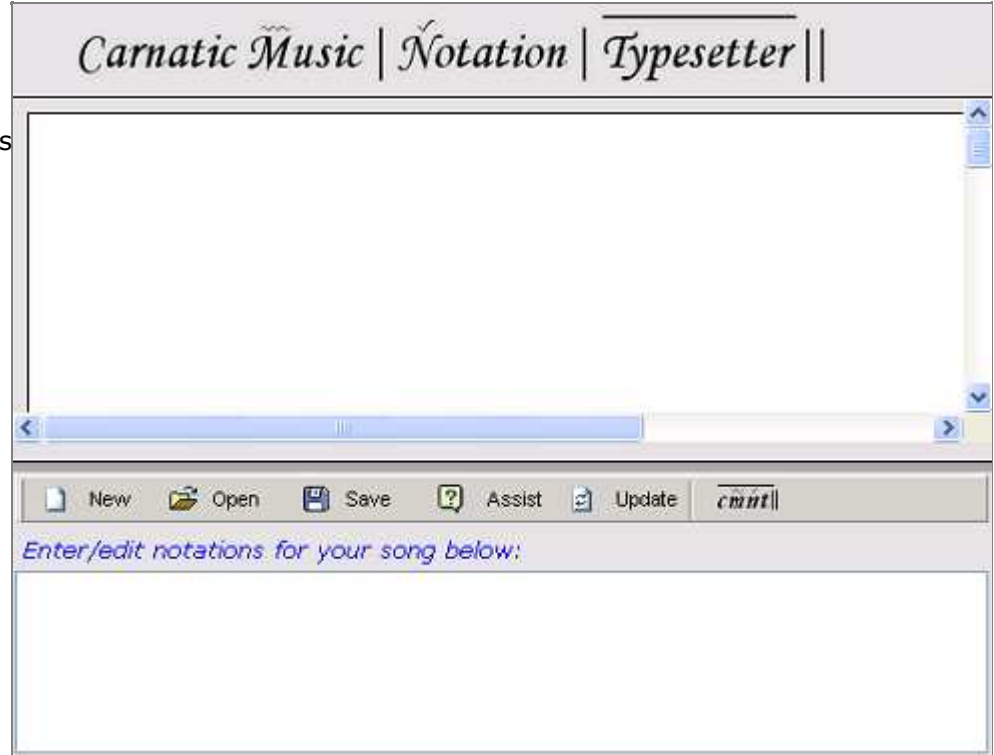
1. The bottom area is the *input area*, where you specify the contents of your song.
2. The top area is the *notation display area* which shows the notation sheet corresponding to your input

Splitter: The spacing of the two panes can be adjusted by clicking on the splitter pane in between and dragging it up or down:








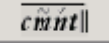
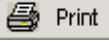
You type in notation content (in the format described in the chapters below) in the text box in the

command area at the bottom below the *Enter/edit notations for your song below:* label.



Command Area Buttons:

- The  **New** button allows you to get started with a new song from scratch. Clicking on this button brings up a dialog where you can specify the song name, raga, tala etc. and this will be converted to the appropriate *directives* (explained below) in the input format.
- The  **Open** button allows you to open a previously saved notation file. Clicking on this button brings up a dialog which allows you to select a previously saved notation file (a plain text file) on your computer.
- The  **Save** button allows you to save your notation content to a file on your computer. The file is saved as a plain text file. Clicking on this button brings up a dialog which allows you to specify the name and location of the file to which the contents should be saved to.
- The  **Assist** button helps you in specifying the various directives. Clicking on this button will provide assistance for entering new directives as well as editing already specified directives, making it easy to specify the notation information without having to memorize the syntax of the various directives. See the **Assistance in entering directives** section for more information.
- The  **Update** button updates/creates the notation based on the input in the text area. So ones you have created your song content, or have made changes to it, click on this button to update the notation display area with those changes.

- The  button brings up a web-page containing information about the typesetter and links to examples, and this manual.
- The  button (not shown in the screenshot above) sends your notation to the printer.

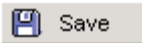
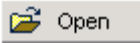
Print your notation, saving as a PDF file

Once you have typed in your notation content in the input area, and made the typesetter generate notation for it by clicking on the **Update** button, you can print the notation by clicking on the **Print** button. If you want to generate a PDF file, you can install a free PDF print driver (there are many available on the internet) and make the browser print to a PDF file.

Note that the **Print** button switches the editor to print view. To get back to the editor, click on the [Back to Editor](#) hyperlink on the print view. **Special note for printing with Firefox:** On Firefox, make sure the *Print Background (colors & images)* checkbox in the *Page Setup* dialog is checked. Otherwise, the notation will not print correctly. This dialog can be accessed by clicking on the *Page Setup ...* button in the Print Preview mode.

Special note for printing with Internet Explorer: If you are using Internet Explorer version 7.x, make sure the *Print background colors and images* checkbox under the *Printing* category in the *Advanced* Tab of the *Internet Options* dialog is checked. On Internet Explorer 6.x, this is not required as the back

Saving your notation

Click on the  button to save your input to a text file on your computer. You can reload and reuse the contents of the file at a later time by clicking on the  button.

Important: Do not hit the refresh/back button of your browser without saving your work, otherwise all your contents would be lost!

Introduction

The input format for the typesetter is a simple ASCII based format i.e. which uses standard (english) letters and signs available in all keyboards. This allows for you to create and maintain the input as a simple text file, and import it into the typesetter as and when needed.

Each line of the input format contains a **directive**. A directive can be specify one of many things:

- Tala of the song
- Headings, which can specify title of the song are be used to indicate separate sections of the songs like *pallavi*, *anu-pallavi* etc.
- Swaras
- Lyrics
- Color of the text for swaras and lyrics
- Font of the text for swaras and lyrics
- Should lines to mark "second speed" etc. are to drawn below or above a swara

Quick Start through an example

To understand and quickly become familiar with the input format, let us look at an example. Here is a short snippet of a song specified using an input format, and how it is rendered by the typesetter:

Input Format	<pre>Tala: Adi DefaultSpeed: 1 Heading: "brOva bhAramA - bahudAri - Adi - tyAgarAja",bold,italic,14,center Heading: "" Heading: "pallavi:",bold,left,12 S: "1)" ; , (p d n ,) pa m ga , g m pa , L: brO _ va _ bhA ra mA _ ra ghu rA , S: "2)" m~ , , (p d n s' n p ;) m ga , g m pa , L: mA . . brO _ vA . . bhA _ ra mA _ ra ghu rA ,</pre>
Typesetter Output	<p style="text-align: center;">brOva bhAramA - bahudAri - Adi - tyAgarAja</p> <p>pallavi:</p> <pre>1) ; , p d n , pa m ga , g m pa , brO va bhA ra mA ra ghu rA ,</pre> <p>~~~~</p> <pre>2) m , , p d n s n p ; m ga , g m pa , mA . . brO vA . . bhA ra mA ra ghu rA ,</pre>

Let us now look at the input line by line.

```
#1: Tala: Adi
```

This line contains a *Tala directive* that specifies catuSra gati Adi tala as the tala for the song. Specifying the tala also controls how the typesetter should layout the notation of the song and when and where tala markers need to be placed. Check out the **Tala Directive** section below for more information on the the various talas that can be specified using the tala directive.

```
#2: Heading: "brOva bhAramA - bahudAri - Adi - tyAgarAja",bold,italic,14,center
```

```
#3: Heading: ""
```

```
#4: Heading: "pallavi:",bold,left,12
```

These line contain *Heading directives*. Line #2 directs the typesetter to place the text *brOva bhArama - bahudAri - Adi - tyAgarAja* in 14 point size, bold, and italic and centered on the notation display area. You can see in the typesetter output area above that this is how the heading appears.

Line #3 specifies an heading directive with an empty heading. This simply results in an empty line in the typeset output, and can be used to introduce empty vertical space wherever needed.

Finally, Line #4 directs the typesetter to place a heading *pallavi*: in 12 point size, bold and aligned at the left edge of the display.

Check out the **Heading Directive** section below for more information on the heading directive. The above examples are very simple and basic. There is a lot of power and flexibility available in specifying a heading, which you can use to specify mixed bold, italic and underlined, text in various colors, text that is tabular (i.e. rendered as a table), or text that is translated to an Indic language, or even insert an image!

```
#5: S: "1)" ; , ( p d n , ) pa m | ga , g | m pa , ||
```

```
...
```

```
#7: S: "2)" m~ , , ( p d n s' n p ; ) m | ga , g | m pa , ||
```

These lines start with **S:**, a *Swara directive*, which specifies notation i.e. swaras for the song. Here are some key points to note in the syntax used to specify swaras:

- **Swara Labels:** *s*, *r*, *g*, *m* etc. indicate swaras, and *sa*, *ri*, *ga*, *ma* etc. indicate dhIrgha or elongated swaras. You can also specify a dhIrgha swara as *n*, instead of *ni*.
- **Tara/Mandra Stayi Swaras:** To indicate a swara in tara stayi, use the forward-tick character as a suffix as in *s'* (as in Line #7) or *sa'*. Similarly to indicate a swara in mandra stayi, use the backward-tick character as *p`* or *pa`*.
- **Pauses, Continuations:** The comma (,) and semi-colon (;) are pauses/continuations, with the latter being twice as long in duration as the former.
- **Switching speed/kAlam:** The parenthesis characters (, and) are used to switch from normal speed to higher speeds. But what is the normal speed? Since the above example did not explicitly specify a layout (see **Layout Directive** below), the default layout is *krithi layout*. In the *krithi layout*, for *catuSra gati* (i.e. as per *tala* specified), swaras by default are in second speed i.e. 2 swaras per akshara. So in the above example, the swaras outside of paranthese are in the default, second speed. However, phrases *p d n ,* on line #5, and *p d n s' n pa ;* on line #7 are in third speed, and thus are rendered as such in the output. Note that the paranthesis must be separated by spaces on either side.
- **Tala Markers:** The special characters |, and || indicate *tala* markers but they are only for clarity in the input for you as the user. They are simply skipped by the typesetter which automatically determines where *tala* markers fall for the current *tala* based on the duration of the swaras. In other words, you dont have to specify these in the input, their presence/absence has no effect on the typeset output, but having them there may allow the input to be more understandable.
- **Sangati Labels:** Both the above swara directives here begin with a quoted string, which is a special indicator telling the typesetter to attach a label to the left of the swara lines (and their associated lyric lines). The labels here are 1) and 2) respectively, and in this example are used to indicate the first and second sangatis of the *pallavi*.
- **Gamaka specifiers:** Following the swara and the stayi suffix, certain characters or patterns are allowed to indicate standard gamakas. In the example above, on Line #7, the *m~*, with the tilde/wavy-line character (~) after *m*, indicates the *kampita* gamaka for that *madhyama* swara. You can see in the rendition that the gamaka is rendered as a glyph above the swara as shown in the typesetter output area above.
 - See the **Swara Directive** section below for more information on how to specify gamakas. You can use special characters like ~ that map to specific predefined glyphs, and also specify anuswaras.

Check out the **Swara Directive** section below for more information.

Let us move on to the next couple of lines in the input.

```
#6: L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,
```

```
...
```

```
#8: L: mA . . brO _ vA . . bhA _ ra mA _ ra ghu rA ,
```

These lines starts with **L:**, a *lyric directive*, which specifies lyrics associated with the swaras for the previous *swara directive* seen. Lyrics for the swaras are separated by one or more spaces. It is important to note that you **must** specify one lyric for every swara (including pauses) in the swara line. An empty lyric can be specified using the underscore character (_) as employed in this example. It is also not uncommon to use a period (.) instead, particularly if the corresponding swara/continuation should be emphasized as with gamaka.

It is not necessary to align the lyrics against the swaras in the corresponding swara directive as done in the example above. The typesetter will automatically align them in the typeset output as shown above. Of course it makes the input a lot more readable if the lyrics are aligned against the swaras using spaces. Note also that in the typesetter program has a button that can automatically align the raw input itself automatically for you.

Check out the **Lyric Directive** section below for more information.

Assistance in entering directives

As mentioned earlier, the input area of the typesetter application has an **Assist** button that can assist you in entering new directives, and editing already specified directives making it easy to specify the notation information without having to memorize the syntax of the various directives. There are two kinds of assistance:

- **Directive specific dialogs/forms:** Here assistance is provided via dialogs/forms specific to a directive.

In the example below, the **Assist** button is pressed when the cursor on an empty line:



Fig A1

As shown above, a dialog that asks you to select a directive that you wish to add is shown. Pick a directive and click OK, and a dialog specific to that directive will be displayed.

In the second example below, after "Heading: " was typed on a new line, the **Assist** button was pressed:

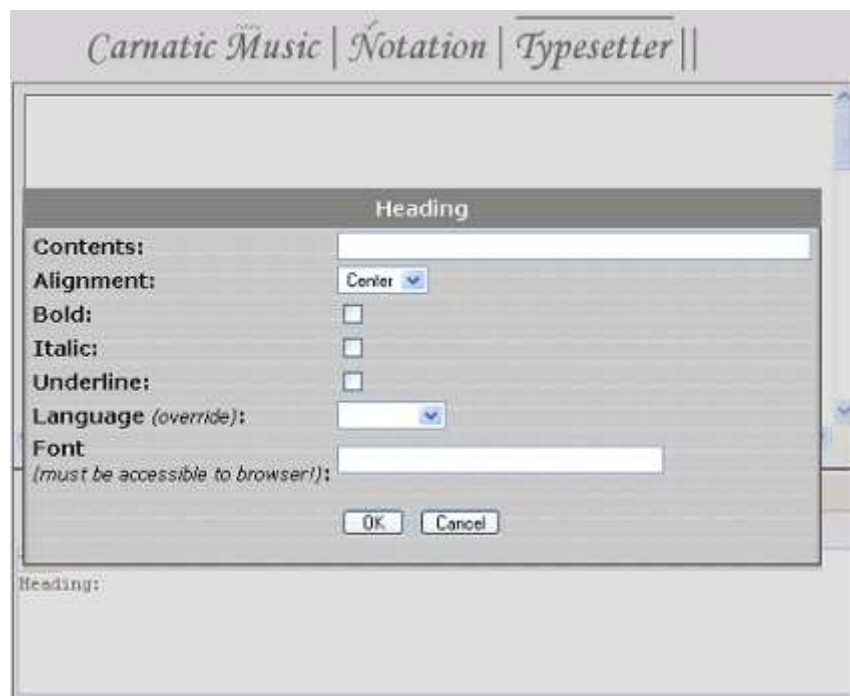


Fig A1

As shown above, a form for entering the heading is shown. Here you can specify the heading text, the alignment, the formatting etc. After entering that, if you click "ok", then the contents of heading directive is appropriately constructed and placed in the text area as shown below:

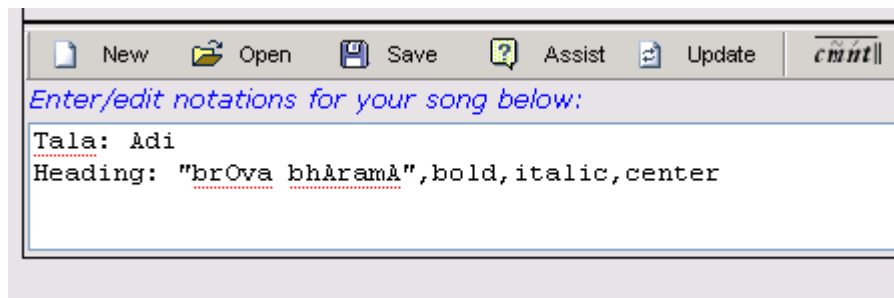


Fig A2

You can bring up the assistance dialogs when the line where the cursor is in the input area is one of the foollowing:

- An empty line. In this case, as explained above a dialog that lets you firsy pick a directive would be shown.
 - Contains just the directive name: The dialog specific to the directive would be shown and you can specify all the parameters for the directive.
 - Contains a complete valid directive: Again, the dialog specific to the directive would be shown, and you can change the already specified parameters.
- **Inline assistance (TAB completion):** Once you get familiar with the syntax of the directives, some may find it faster to type in the raw directives rather than use the forms. You can press the tab key while typing in the directive, and the application should complete the directive information wherever appropriate. For example, if on a new line, you enter just the letter "t" and than press the TAB key (see Fig A3 below), the application will immediatly fill "Tala: " as a beginning of a tala directive (see Fig A4 below). Then for the value for the tala directive, if for example, you enter "k" and press TAB (Fig A5 below), the application will fill in "KhandaCapu" (Fig A6 below).

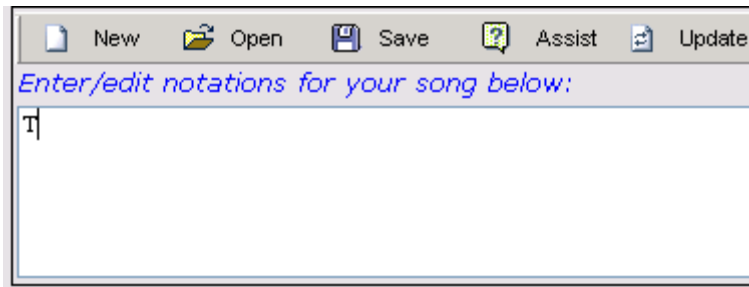


Fig A3

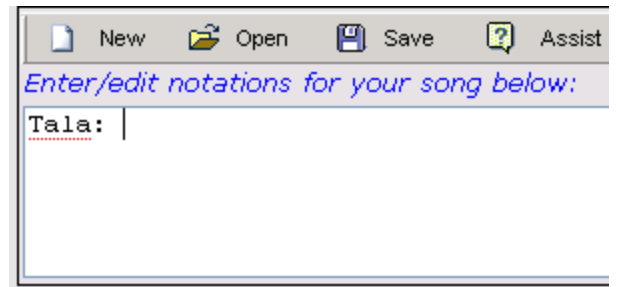


Fig A4

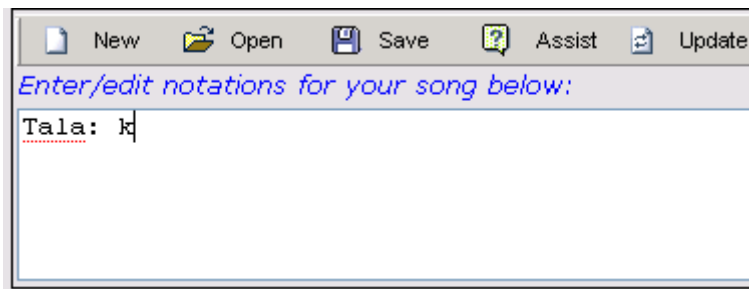


Fig A5

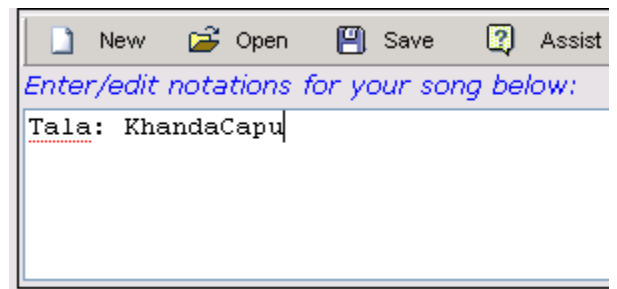


Fig A6

Directives - in detail

Each line of the input format contains a directive. The syntax of a directive is the directive name, followed by a colon, which is then followed by the directive value:

```
directive_name:directive_value
```

For example:

```
Tala: Adi  
S: s r g m | p d | n s
```

Directive names are case insensitive. There can be spaces around the colon separating the directive name and the value. Also in most places, more than one space is \Leftrightarrow one space. So all the following *Tala* and *S* (for swaras) directives are equivalent to each other:

```
tala: Adi  
  tALa: Adi  
  Tala: Adi  
  Tala : Adi  
  Tala:           Adi  
  Tala           :   Adi  
  
S: s r g m | p d | n s  
S: s   r   g   m | p d |   n s
```

Spacing is flexible so that you can give whatever spacing that is needed to make the input format itself readable.

Header vs. Body section: The specification of the song in this format is roughly divided into 2 sections:

- The header portion where you specify overall attributes for the song. Certain *directives* must appear in the header portion only.
- The main body where you specify the swaras and the associated lyrics. You can also specify headings for different portions of the song here. Certain *directives* cannot appear in the body portion and must appear only in the header portion

There is no clear demarcation between the header and the body portion. Basically the first heading or swara or lyric signals the start of the body portion.

Layout Directive

The `Layout` directive indicates the type of song that is being notated which in turn plays a part in determining the layout of the notation. It must be specified in the header section so that the tala for the song is known when the first swara line is encountered.

The `tala` directive has the following syntax:

```
Layout: layout_value
```

The value is one of the following pre-defined layout values:

- `gitam` or `geetam` to indicate a *gItam* layout
- `varnam` to indicate a *varnam* layout
- `kriti` or `krithi` to indicate a *krithi* layout. Note that if no Layout directive is specified, the typesetter presumes a krithi layout.

Please refer to the individual sections below for more information on these layouts.

Geetam Layout

If you specify the value of the Layout directive as `gitam` or `geetam`, then it indicates that the typesetter should layout the notations as if for a geetham.

```
Layout: gitam
```

The gItam layout implies the following:

- The default speed of swaras would be what is colloquially called "first speed", where each swara would span one akshara as it is for a *gItam*. (Note that this can be overridden with **DefaultSpeed** directive)
- An extra space (or tala marker) is provided between each akshara of a tala so that each akshara stands out clearly, and notations across lines (avarthanas) are aligned at each akshara. This is in contrast to the krithi format, where any extra space would appear only at tala anga ends and aksharas within an anga "run together".

Here is an example of a gItam layout of the most popular gItam:

```
Layout: gitam
Tala: rupaka
S: m p d s' s' r' r' s' d p m p
L: sRi . ga Na nA ta sin du ra . var na
S: r m p d m p d p m g r s
L: ka ru Na sA ga ra ka ri va da na .
```

(Note: "Tala" specifies the tala, and "S" and "L" specify Swaras and Lyrics. See Tala directive, Swara Directive and Lyric Directive sections for more information)

```
m p | d s s r || r s | d p m p ||
sRi . ga Na nA ta sin du ra . var na

r m | p d m p || d p | m g r s ||
ka ru Na sA ga ra ka ri va da na .
```

Varnam Layout

If you specify the value of the Layout directive as `varnam`, then it indicates that the typesetter should layout the notations as if for a varnam.

```
Layout: varnam
```

The varnam layout implies the following:

- The default speed of swaras would be what is colloquially called "second speed", where each swara would span two aksharas as it is for a *varnam*. (Note that this can be overridden with **DefaultSpeed** directive)
- An extra space (or tala marker) is provided between each akshara of a tala so that each akshara stands out clearly, and notations across lines (avarthanas) are aligned at each akshara. This is in contrast to the krithi format, where any extra space would appear only at tala anga ends and aksharas within an anga "run together".

Here is an example of a varnam layout of the one more popular varnams:

SpeedMarks Directive

The `SpeedMarks` directive controls whether the typesetter should draw the lines for higher speeds (mEI kAlam) below the swaras or over the swaras. The value of the directive must be either `below` or `above`.

The default setting is to draw the lines above the swaras, and so an explicit `SpeedMarks: above` is not strictly needed.

Here are examples of usage of `SpeedMarks` directive:

<pre>SpeedMarks: above Tala: Adi S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<pre>_____ ; , p d n , pa m ga , g m pa , brO va bhA ra mA ra ghu rA ,</pre>
<pre>SpeedMarks: below Tala: Adi S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<pre>; , <u>p d n</u> , pa m ga , g m pa , brO va bhA ra mA ra ghu rA ,</pre>

Revision History

- **v1.0:** Introduced

PhraseEnds Directive

The `PhraseEnds` directive controls the display of the phrase end markers (-, i.e. dash/hyphen), which can be used at the end of a swara specified in the Swara Directive. This marker is used to group swaras i.e. as in indicate sub-phrases within a phrase, or where a pause or breath should be taken while singing.

The `PhraseEnds` directive has the following syntax:

```
PhraseEnds: <style>
(or)
PhraseEnds: <style>,<trailing_space>
```

where,

- `<style>` specifies the style of the phrase ends in the typeset notation. It can be one of
 - `Hide` - to make the phrase ends invisible in the display.
 - `Show` - to make the phrase ends show as-is i.e. as hyphens.
 - `Handle` - to make the phrase ends show as a handle in the form of a ▬ glyph, aligned to the bottom right of the swara that is at the phrase end.
 - `HandleThick` - to make the phrase ends show as a thick handle in the form of a ▬ glyph, aligned to the bottom right of the swara that is at the phrase end.
- `<trailing_space>` specifies the amount of trailing space to introduce after the phrase (thus separate phrases in the display making them stand out). This can be specified in two ways:
 - A number `n`, where `n` is 0 or greater. This will apply a trailing space of `n` pixels
 - In terms of inches. For example `0.15in` for a spacing of 0.15 inches.

Note that the default style is `Hide`, and default spacing is 0 (i.e. no spacing) Also that the speed marks are always broken at phrase ends to show phrasing.

Here are some examples:

Invisible, no spacing (default)	Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p - n d p m p a
Show as hyphens, no spacing	PhraseEnds: show Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p - n d p m p a
Invisible , spacing of 15 pixels	PhraseEnds: Hide,15 Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p n d p m p a
As hyphens, spacing of 10 pixels	PhraseEnds: Show,10 Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p - n d p m p a
As handle, spacing of 10 pixels	PhraseEnds: Handle,10 Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p n d p m p a
As thick handle, spacing of 0.15 inches	PhraseEnds: HandleThick,0.15in Tala: Adi S: (s' n' d p- n d p m) pa	_____ _____ s n d p n d p m p a

Revision History

- **v1.1:** Added `Handle`, `HandleThick` styles; Also added spacing control

DefaultSpeed Directive

The `DefaultSpeed` directive controls what the default speed of the swaras in a Swara Directive are. The speed here implies the duration of the swara. The typesetter current supports 3 speeds:

- **0** - which implies what is colloquially known as first speed, where each swara spans a full akshara.
- **1** - which implies what is colloquially known as second speed, where e.g. in *catusra gati*, each swara spans half an aksara, or 2 swaras taken up an akshara. Note that in this speed for *tIsra gati*, 3 swaras take up 1 akshara. Similarly for *khaNDa*, *miSra* and *sankIrNa gati*, it would be 5, 7 and 9 swaras per akshara respectively.
- **2** - which implies what is colloquially known as third speed, where e.g. in *catusra gati*, each swara spans quarter of an aksara, or 4 swaras taken up an akshara. For *tiSra*, *khaNDa*, *miSra* and *sankIrNa gati* it would be 6, 10, 14 and 19 swaras.

Note that depending on the layout (see `Layout` directive), an appropriate default speed is automatically set for you. The `DefaultSpeed` directive can be used to override that default. The value of the `DefaultSpeed` directive must be one of the above speeds. This implies that swaras at higher speeds will be shown with speed lines (above or below - see `SpeedMarks` directive). Note that there is no way to specify a speed that is *slower* than the default, and so e.g. if you pick a default speed as 1, then there is no way to specify swaras of speed 0. Hence you should pick such a speed only if there is no such need.

Here are some examples of usage of the `DefaultSpeed` directive. The `varnam` layout (see **Layout Directive**, while it may not be most appropriate to show various speeds, is used here for illustrative purposes, as it makes the typesetter show each akshara clearly:

DefaultSpeed: 0 Layout: varnam Tala: Adi S: s r (s r (g m p d)) r g (r g (m p d n))	s r sr <u>gmpd</u> r g rg <u>mpdn</u>
DefaultSpeed: 1 Layout: varnam Tala: Adi S: sa ri s r (g m p d) ri ga r g (m p d n)	sa ri sr <u>gmpd</u> ri ga rg <u>mpdn</u>
DefaultSpeed: 2 Layout: varnam Tala: Adi S: sa ; ri ; sa ri g m p d ri ; ga ; ri ga m p d n	sa; ri; sari <u>gmpd</u> ri; ga; riga <u>mpdn</u>

Revision History

- **v1.0**: Introduced

Tala Directive

The `Tala` directive specifies the tala for the song and this is one of the parameters that determines how the notations should be laid out. The tala anga markers are also determined based on this and automatically placed by the typesetter. The tala directive must be specified in the header section so that the tala for the song is known when the first swara line is encountered.

Manual Tala: The typesetter also supports a "Manual" tala, which turns off automatic placement of tala anga markers. Instead markers need to be explicitly indicated in the **Swara Directive** (as | and ||). The typesetter also will not start a new line, unless you explicitly use the **SongBreak Directive** (or start a Heading). The Manual mode is useful for laying out notations for snippets of ragas, and also for songs whose layout is more complex than what the typesetter can handle. Note however that the typesetter will not be able to align notations at various anga markers in this mode.

The `tala` directive has the following syntax:

```
Tala: Tala_Name
```

The value is one of the following pre-defined tala names as listed in the table below:

Tala	How to specify <i>(Note: Tala Names are case insensitive)</i>
Adi	Tala: <code>Adi</code> (catusra gati) or Tala: <code>Tisra_Adi</code> (triSra/tiSra gati) or Tala: <code>Adi2Kalai</code> (Adi 2-kalai) or Tala: <code>adi</code>
Roopaka (cApu)	Tala: <code>Roopakacapu</code> or Tala: <code>Rupakacapu</code> or Tala: <code>rUpakacApu</code>
Roopaka	Tala: <code>Roopaka</code> (catuSra jAti roopaka) Tala: <code>Tisra_Roopaka</code> (tiSra jAti roopaka) Tala: <code>Catusra_Roopaka</code> (catuSra jAti roopaka) Tala: <code>Khanda_Roopaka</code> (khaNDa jAti roopaka) Tala: <code>Misra_Roopaka</code> (catusra jAti roopaka) Tala: <code>Sankirna_Roopaka</code> (sankIrna jAti roopaka) <i>(Note: You can specify Rupaka instead of Roopaka in the above)</i>
khaNDa cApu	Tala: <code>KhandaCapu</code> (laid out as 4 avarthanams per row) or Tala: <code>KhandaCapu2</code> (laid out as 2 avarthanams per row)
miSra cApu	Tala: <code>MisraCapu</code>
Eka	Tala: <code>Eka</code> (catuSra jAti eka) Tala: <code>Tisra_Eka</code> (tiSra jAti eka) Tala: <code>Catusra_Eka</code> (catuSra jAti eka) Tala: <code>Khanda_Eka</code> (khaNDa jAti eka) Tala: <code>Misra_Eka</code> (catusra jAti eka) Tala: <code>Sankirna_Eka</code> (sankIrna jAti eka)

Tala	How to specify <i>(Note: Tala Names are case insensitive)</i>
Triputa	Tala: <code>Triputa</code> (tiSra jAti tripuTa) Tala: <code>Tisra_Triputa</code> (tiSra jAti tripuTa) Tala: <code>Catusra_Triputa</code> (catuSra jAti tripuTa) Tala: <code>Khanda_Triputa</code> (khaNDa jAti tripuTa) Tala: <code>Misra_Triputa</code> (catusra jAti tripuTa) Tala: <code>Sankirna_Triputa</code> (sankIrna jAti tripuTa)
Dhruva	Tala: <code>Dhruva</code> (catusra jAti dhruva laid out as 1 avarthanam per row) Tala: <code>Dhruva2</code> (catusra jAti dhruva laid out as 1 avarthanam over 2 rows) Tala: <code>Tisra_Dhruva</code> (tiSra jAti dhruva) Tala: <code>Catusra_Dhruva</code> (catuSra jAti dhruva) Tala: <code>Khanda_Dhruva</code> (khaNDa jAti dhruva) Tala: <code>Misra_Dhruva</code> (catusra jAti dhruva) Tala: <code>Sankirna_Dhruva</code> (sankIrna jAti dhruva)
Matya	Tala: <code>Matya</code> (catuSra jAti maTya) Tala: <code>Tisra_Matya</code> (tiSra jAti maTya) Tala: <code>Catusra_Matya</code> (catuSra jAti maTya) Tala: <code>Khanda_Matya</code> (khaNDa jAti maTya) Tala: <code>Misra_Matya</code> (catusra jAti maTya) Tala: <code>Sankirna_Matya</code> (sankIrna jAti maTya)
Khanda Ata	Tala: <code>Ata</code> (khaNDa jAti aTa) Tala: <code>Tisra_Atata</code> (tiSra jAti aTa) Tala: <code>Catusra_Atata</code> (catuSra jAti aTa) Tala: <code>Khanda_Atata</code> (khaNDa jAti aTa) Tala: <code>Misra_Atata</code> (catusra jAti aTa) Tala: <code>Sankirna_Atata</code> (sankIrna jAti aTa)
Misra Jhampa	Tala: <code>Jhampa</code> (miSra jAti jhampa) Tala: <code>Tisra_Jhampa</code> (tiSra jAti jhampa) Tala: <code>Catusra_Jhampa</code> (catuSra jAti jhampa) Tala: <code>Khanda_Jhampa</code> (khaNDa jAti jhampa) Tala: <code>Misra_Jhampa</code> (catusra jAti jhampa) Tala: <code>Sankirna_Jhampa</code> (sankIrna jAti jhampa)
Manual (you control everything)	Tala: <code>Manual</code>

Here are some examples of usage of tala.

Tala: Adi DefaultSpeed: 1 S: s r g m p d n s' s n d p m g r s	srgmpdns sndp mgrs
Tala: MisraCapu DefaultSpeed: 1 S: s , n d m d n d , m g m d m	s,n dm dn d,m gm dm
Tala: KhandaCapu DefaultSpeed: 1 S: s g s g m g m g m p	sg sgm gm gmp
Tala: RupakaCapu DefaultSpeed: 1 S: s r g r g m g m p m p d p d n d n s'	sr gr gm gm pm pd pd nd ns
Tala: Tisra_Adi DefaultSpeed: 1 S: s r g m p d n s' s' n d m p g r s s' n d p m g r s	srgmpdnssndm pgrssn dpmgrs
Tala: Rupaka DefaultSpeed: 0 S: m p d s' s' r r' s d p m p S: r m p d m p d p m g r s	mp dssr rs dpmp rm pdmp dp mgrs

Revision History

- **v1.0:** Introduced
- **v1.2:** Added support for Khanda Triputa Tala
- **v1.4:** Added support for all 35 talas (catusra gati only), and Manual tala

Heading Directive

The `heading` directive can be used to specify text labels to include in your notation. These can be used to represent headings such as title of the song, raga, and also headings of portions of song as pallavi, anupallavi. Empty headings can be used to add empty space.

The `heading` directive has the following syntax:

```
Heading: "Heading_text", attributes
```

where,


- *Heading_text* (which must be enclosed in double-quotes as shown above) is the text of the heading. This text will be displayed as-is unless the typesetter has been asked to translate to another language via either the **Language Directive** or via a language specified as one the heading attributes (see below).
- *attributes* is a comma separated list of one or more of the following:

font name	<p>For example, <code>Heading: "taLa: Adi", Arial</code> <code>Heading: "taLa: Adi", MS Sans Serif</code></p> <p>If you do not specify a font name, the current default font in your browser is picked. Note that the font name must match a font installed on your computer and hence may not render the same way on your friend's computer if you shared the notation data with him/her. If you intend to share your notation information in raw format, then it is recommended that you pick common fonts or not specify a font.</p>
Point size of font as a number	<p>For example, <code>Heading: "rAga: AbhOgi", Verdana, 12</code> to render the heading using the font Verdana 12 pt size.</p>
bold	<p>For example, <code>Heading: "rAga: AbhOgi", Verdana, 12, bold</code> to render the above heading in bold font</p>
italic	<p>For example, <code>Heading: "rAga: AbhOgi", Verdana, 12, italic</code> to render the above heading in italic font</p>
One of left, right, center to indicate alignment.	<p>For example, <code>Heading: "rAga: AbhOgi", Verdana, 12, bold, italic, center</code> to render the above heading in bold, italic font and centered.</p>
One of tamil, sanskrit, telugu or kannada	<p>This indicate that the actual heading text is in the Unified Transliteration Scheme for Carnatic Music Compositions and should be rendered in a Unicode font in the specified language. This language specification also overrides the default language specified by the Language Directive.</p> <p>For example, <code>Heading: "alai pAyudE kaNNA!", 12, bold, italic, center, tamil</code> would interpret <i>alai pAyudE kaNNA!</i> to specify a tamil text following the Unified Transliteration scheme, and convert it to the appropriate tamil letters and render it.</p> <p>Please note that you must have a font that can render in these languages for the heading to appear correct. Please see the LanguageFont Directive section for specifying different fonts for different languages.</p> <p>Overriding some Language attributes: You can use <i>Extra Language Attributes</i> (see Language Directive section) to override the attributes that specified (or implied by default) by the <code>Language</code> directive. For example, let us say that the <code>Language</code> Directive was specified as <code>Language: Tamil, granthasa</code>, to imply that by default text should be rendered in Tamil, should use ூ for Sa, and should use qualifiers for non-Tamil sounds like kha, gha etc. Then, if you want a specific heading, to be rendered in Tamil, but not use grantha sa, and also not use any qualifiers, you can include the attribute <code>Tamil:nogranthasa:noqual</code>.</p> <p><i>Note: Use of the <code>granthasa</code> attribute requires your system to meet some requirements - please see the Language Directive section.</i></p>

Note that the attributes can be in any order and the values are case-insensitive.

Specifying advanced formatting using Wiki directives

In the above examples, formatting such as bold, italic was specified as a separate directive. You can instead specify much richer format by using special characters in the heading text itself. The syntax here follows the popular Wiki style. Here is a summary of some common formatting constructs you may want to use:

<code>*bold text*</code>	bold text	Enclose text inside two asterisks to render that part of the heading in
<code>_italic text_</code>	<i>italic text</i>	Enclose text inside two underscores to render that part of the heading
<code>fifth</code>	fif th	Enclose a text inside two caret (^) characters to render that part as s
<code>h~2~o</code>	h ₂ o	Enclose a text inside two tilde (~) characters to render that part as s
<code>{c:red}</code>	red colored heading	The general syntax is Heading: "{c:color}heading text", where color specifies a color by
<code>{c:=}</code>	Underlined heading	Start the heading with {c:=} to render the heading as underlined
<code>---</code>	 a horizontal line	A heading text like "---" draws a horizontal line across the notation ar
<code>[img:url]</code>	to insert a image	Example: Heading: "[img:http://www.mysite.com/mypage/myimage.jpg]"
<code>[row1_cell1 row1_cell2 \n row2_cell1 row2_cell2]</code>	A tabular display You can specify a table to display by adhering to the following rules: <ul style="list-style-type: none"> • The table content must be enclosed in square brackets [and] • The content of every cell is specified within two , as in cell1 . <ul style="list-style-type: none"> • If you specify a leading space but no trailing space, content is right ali • If you specify a trailing space but no leading space, content is left ali • Else content is centered • Start a new row by first specifying either \\n (backslash character followed b character followed by a newline. <p>Example: Here is an example of a table with 2 columns, with each row representing a line of indicates section of song as in pallavi, anupallavi etc., which are rendered in bold. rendered in point size of 10. There are 4 rows. Note that the last row specifies an e heading) as just .</p> <pre>Heading: "[*P:* nAdalOluDai brahmAnandamandavE manasA \n *A:* svAdu phalaprada sapta svara rAga nicaya sahita \n *C:* hariharAtmabhU surapati Sarajanma ganESAdi \n varamaunulu (u)pAsincarE dhara tyAgarAju teliyu]",10</pre> <p>Note that the above formatting for bold, italic, underlined, colored text etc. and be formatting of individual cells.</p> <p>Here is how the typesetter would render it:</p> <p>P: nAdalOluDai brahmAnandamandavE manasA A: svAdu phalaprada sapta svara rAga nicaya sahita C: hariharAtmabhU surapati Sarajanma ganESAdi varamaunulu (u)pAsincarE dhara tyAgarAju teliyu</p>	

The typesetter uses a 3rd part convertor called **Wikiy** that handles the Wiki formatting. Please refer to the Wiki website at <http://goessner.net/articles/wiki> for complete information on the various formatting constructs you can use. Please note that since the formatting is done within the framework of the notation layout, not all of them may work as you may expect.

Multi-line heading

You can specify a heading that spans multiple lines. Use `[br]` to introduce a line break as illustrated by the following example:

Heading: "First Line[br]Second Line"
Heading: ""
Heading: "*bold line*[br]_italic on next line_"
First Line Second Line
bold line <i>italic on next line</i>

Revision History

- **v1.0**: Introduced
- **v1.1**: Language specifier can include extra language attributes - see **Language Directive** section

HeadingPrefs Directive

The `HeadingPrefs` directive can be set the default preferences for subsequent Heading directives such as the default font size, and default font/text color. The syntax of the `HeadingPrefs` directive is as follows:

```
HeadingPrefs: attributes
```

where *attributes* is a comma separated list of one or more of the following:

- *font size*: The default point size of the headings as a number
- *color*: The color of the text of the headings. This can either be a color name such as *green*, *blue* or a RGB (red, green, blue) combination.

Here are some examples of usages of the `HeadingPrefs` directive:

<code>HeadingPrefs: 12</code>	Sets the default point size for headings to 12pt.
<code>HeadingPrefs: 12, red</code>	Sets the default point size for headings to 12pt and color to red.
<code>HeadingPrefs: 12, rgb(64, 100, 125)</code>	Sets the default point size for headings to 12pt, and the color to a a color whose red-component is 64, blue component is 100, and green component is 125. The <i>rgb(red, green, blue)</i> of color specification takes a value between 0 and 255 for each of the red, green and blue component. The maximum value of 255 for all three i.e. <i>rgb(255,255,255)</i> is white, the minimum value of 0 for all three i.e. <i>rgb(0,0,0)</i> is black. All the colors of the spectrum fall between these two and take varying values for these three components.

Revision History

- **v1.0**: Introduced

Swara Directive

The `s` (swara) directive can be used to specify notations for the song. You can specify as many such directives as you want - and the typesetter will combine the notations specified in all such directives and lay them out as per the tala specified by the **Tala directive**

The `s` directive has the following syntax:

```
S: notations
(or)
S: "Label" notations
```

where,

- `notations` is a space separated list of notations (swaras), whose syntax is described in detail below.
 - *swara*: Indicates a swara using the syntax indicated below
 - *tala marker*: One of | or || for tala markers. Note however that these are only for clarity in the input for you as the user. They are simply skipped by the typesetter which automatically determines where tala markers fall for the current tala based on the duration of the swaras. In other words, you don't have to specify these in the input, their presence/absence has no effect on the typeset output, but having them there may allow the input to be more understandable. For example, you can specify
`s r g m | p m | g r || f`
- `label` is a label that is applied to the swara (and associated lyric line) and appears on the left. This can be used to specify sangati numbers.

Specifying Notations in a Swara directive

Notations are specified in the swara directive as a space separated list of the following: *Swaras*, *Pauses*, *Speed Change indicators* and *tala markers*.

Swaras

The syntax of a *Swara* can be one of the following:

```
<swara_indicator><optional_stayi_indicator><optional_gamaka_indicator><optional_phrase_end_indicator>
<pause_or_continuation_indicator><;optional_phrase_end_indicator>
```

Where,

- ***swara_indicator*** is one of `s r g m p d n` to indicate normal swaras than span one mAtra in the current speed, or `sa ri ga ma pa da ni` to indicate elongated or dhIrga swaras that span two mAtras in the current speed.
- ***pause_or_continuation_indicator*** is either a comma (,) or a semicolon (;), which can be used to indicate a real pause, or a continuation of a swara when following it (and thus can be an alternate way of representing a dhIrga swara). The comma spans one mAtra in the current speed, and the semicolon spans two mAtras in the current speed.
- ***Empty swara indicator*** simply occupies space without a visual swara. It is specified as either a single underscore (_) or two underscores together (__). This can be useful for certain styles of notation where you want to indent the swaras by the eDuppu or start of point instead of visually indicating pauses. One use is in Ata tala varnams which typically do not start at the tala cycle (i.e. not at *samam*). The pallavi will have 2 avarthanas where the line extends beyond the end of tala into the duration of the eduppu. While typically, people notate both the lines (spanning 2 avarthanas plus eduppu) as a single block. However, some may prefer to notate the first line (i.e. avarthana plus eduppu) as a block, introduce a break, and then notate the second line (indented to eduppu). This allows for the sahitya integrity to be better reflected in the notation. In such cases, for the second line you can use the empty swara indicators for the indent to the eduppu. Here is an example of usage of the empty swara indicators (although the context in which they are used may not be most appropriate):

Explicit pause indicators before eduppu

```
Layout: Varnam
Tala: Adi
DefaultSpeed: 1
Heading: "pallavi:",bold,left,12
S: ; , ( p d n , ) pa m | ga , g | m pa , ||
L: __ brO _ va _ bhA ra mA _ ra ghurA ,
S: m~ , , ( p d n s' n p ; ) m | ga , g | m pa , ||
L: mA . . brO _ vA . . bhA _ ra mA _ ra ghurA ,
S: m~ , ,
L: mA . .
```

pallavi:

```

; , p d n , p , m | ga , g | m p , , ||
brO va bhA .ra mA ra ghurA .,

~
m , , p d n sn p ; m | ga , g | m p , , ||
mA . .brO vA . . bhA ra mA ra ghurA .,

~
m , ,
mA . .
```

Empty swara indicators before eduppu

```
Layout: Varnam
Tala: Adi
DefaultSpeed: 1
Heading: "pallavi:",bold,left,12
S: __ ( p d n , ) pa m | ga , g | m pa , ||
L: __ brO _ va _ bhA ra mA _ ra ghurA ,
S: m~ , , ( p d n s' n p ; ) m | ga , g | m pa , ||
L: mA . . brO _ vA . . bhA _ ra mA _ ra ghurA ,
S: m~ , ,
L: mA . .
```

pallavi:

```

p d n , p , m | ga , g | m p , , ||
brO va bhA .ra mA ra ghurA .,

~
m , , p d n sn p ; m | ga , g | m p , , ||
mA . .brO vA . . bhA ra mA ra ghurA .,















~
m , ,
mA . .
```

- **optional_stayi_indicator** is used to indicate the stAyi (octave) of the swara. It is optional and so if not provided, the swara is taken to in the madya (middle) stayi. If specified, it should be either the forward tick (') for the tara (upper) stAyi (e.g. s' for the tara shadjam), or the backward tick (`) to indicate the mandra (lower) stAyi (e.g. p` for the mandra pancamam). Following standards used in carnatic music books, the typesetter shows tara stAyi swaras by showing a dot above the swara, and mandra stAyi swaras by showing a dot below the swara. For example S: s' n d p m g r s n` will be rendered as follows:

sndp mgrs | n

- **optional_gamaka_indicator** is used to indicate a gamaka for the swara. This can be one of the following:

- A pre-defined symbol which corresponds on a pre-defined glyph (image) that is displayed above the swara. Currently, the following pre-defined swaras are supported:

Gamaka Indicator	Glyph	Example
~ (tilde)		S: m~ ⇔ 
~~ (two tilde)		S: ma~~ , ⇔ 
^ (caret)		S: r g^ m ⇔ 
/ (forward slash)		S: s m/ g r ⇔ 
// (two forward slashes)		S: s ma// g r ⇔ 
\ (backslash)		S: n d\ p m ⇔ 
\\ (two backslashes)		S: n da\\ p ⇔ 

- Any text enclosed in parenthesis as in S: s' n(w) n d, which will make that text appear above the swara as follows:

w
s n n d

- **optional_phrase_end_indicator** is the minus/hyphen - character, and is used to group swaras i.e. as in phrasing. For example, in S: (s' n' d p- n d p m), the hyphen is used after the the pancharam to indicate that the phrase is divided internally into two sub-phrases s' n d p, and n d p m and should be sung/played as such (*Note: the parantheses are speed change markers discussed below*). In many cases, phrasing indicators are used to indicate where a pause for breath should be taken (vocal). The above example would be rendered as follows:

s n d p n d p m

Note that the **PhraseEnds** directive can control whether the hyphen itself is to be displayed or not.

Speed Change Indicators

In the swara directive, the default speed is **always** predetermined. For krithis, this defaults speed is known colloquially as "second speed" where in catusra gati, you can fit 2 swaras in an akshara. If you want to swaras in "third speed" or mEl kAlam, you must enclose them in parantheses as shown below:

S: g , m , (p m g r s n`) s , , ,

In the above example the first 3 swaras g - - are in second speed. The swaras p m g r s n` is in third speed, and the swaras following it are in second speed again. The above notation would be rendered as follows:

g , m , p m g r s n s | , , ,

Tala Markers

You can also use the special characters | and || in the notations of a swara directive to serve as tala markers. However, you should note that these characters are there for clarity in the input for you as the user. They are simply skipped by the typesetter which automatically determines where tala markers fall for the current tala based on the duration of the swaras. In other words, you don't have to specify these in the input, their presence/absence has no effect on the typeset output, but having them there may allow the input to be more understandable.

The equal sign (=), "same as previous sangati" indicator

In notation, often you will find a need to simply indicate that a certain section of the avarthanam is simply the same as the last one. This is especially true in sangatis where each sangati is a minor variation of the previous one. You can use the equal sign character (=) for these purposes.

The equal sign is in reality a swara that takes up a full akshara. This means that if you want to indicate that the first half of the Adi tala in a certain notation line is the same as previous, you need to specify four separate equal signs for the four aksharas as illustrated in the following example:

```
DefaultSpeed: 1
Tala: Adi
S: p ; ( s' n d n ) p d n | s' , , , | , , , , ||
S: = = = = | = = | ( d n s' , - d n s' , ) ||
```

The above example uses 6 equal signs for to indicate that the notation for the first 6 aksharas of Adi tala simply follows the previous line. It would be rendered as follows:

```
_____
p; sndnpdn      | s,,,      | ,,,,      ||
.. .. .. .. | .. .. | dns, dns, ||
```

Revision History

- **v1.0:** Introduced

SwaraPrefs Directive

The `SwaraPrefs` directive can be set the default preferences for subsequent Swara directives such as the default font size, and default font/text color. The syntax of the `SwaraPrefs` directive is as follows:

```
SwaraPrefs: attributes
```

where *attributes* is a comma separated list of one or more of the following:

- *font size*: The default point size of the swaras as a number
- *color*: The color of the text of the swaras. This can either be a color name such as *green*, *blue* or a RGB (red, green, blue) combination.

Here are some examples of usages of the `SwaraPrefs` directive:

<code>SwaraPrefs: 12</code>	Sets the default point size for swaras to 12pt.
<code>SwaraPrefs: 12, red</code>	Sets the default point size for swaras to 12pt and color to red.
<code>SwaraPrefs: 12, rgb(64,100,125)</code>	Sets the default point size for swaras to 12pt, and the color to a a color whose red-component is 64, blue component is 100, and green component is 125. The <i>rgb(red,green.blue)</i> of color specification takes a value between 0 and 255 for each of the red, green and blue component. The maximum value of 255 for all three i.e. <i>rgb(255,255,255)</i> is white, the minimum value of 0 for all three i.e. <i>rgb(0,0,0)</i> is black. All the colors of the spectrum fall between these two and take varying values for these three components.

Note that if no `SwaraPrefs` directive is given, the default font size is taken to be 10pt, and color to be black.

Revision History

- **v1.0**: Introduced

GamakaPrefs Directive

The `GamakaPrefs` directive can be used to set the default preferences for subsequent gamaka markers in Swara directives. Currently, the preferences apply only to text based gamakas i.e. $g(w)$, which results in the text within the parenthesis w here, to be used as gamaka markers. The preferences allow you to specify the size of font, and font/text color. The syntax of the `GamakaPrefs` directive is as follows:

```
GamakaPrefs: attributes
```

where *attributes* is a comma separated list of one or more of the following:

- *font size*: The default point size of the font of the text based gamakas as a number
- *color*: The color of the text of the text based gamakas. This can either be a color name such as *green*, *blue* or a RGB (red, green, blue) combination.

Here are some examples of usages of the `GamakaPrefs` directive:

<code>GamakaPrefs: 12</code>	Sets the default point size of the font of text based gamaka specifiers to 12pt.
<code>GamakaPrefs: 12, red</code>	Sets the default point size for the font of text based gamaka specifiers 12pt and color to red.
<code>GamakaPrefs: 12, rgb(64,100,125)</code>	Sets the default point size of the font for text based gamakas to 12pt, and the color to a color whose red-component is 64, blue component is 100, and green component is 125. The <i>rgb(red,green,blue)</i> of color specification takes a value between 0 and 255 for each of the red, green and blue component. The maximum value of 255 for all three i.e. <i>rgb(255,255,255)</i> is white, the minimum value of 0 for all three i.e. <i>rgb(0,0,0)</i> is black. All the colors of the spectrum fall between these two and take varying values for these three components.

Revision History

- **v1.0**: Introduced

Lyrics Directive

The `L` (lyrics) directive can be used to the lyrics for the previously specified `S` (swara) directive. Note that for a single `S` directive, multiple `L` directives can be specified and these can be used e.g. to specify the lyrics for the various *charanams* of a *krithi* which typically share the same melody (*meTTu* in Tamil).

The `L` directive has the following syntax:

```
L: lyrics
```

, here

- `lyrics` is a space separated list of text specifying each lyric fragment. **You must specify a lyric fragment for every swara in the swara directive.** So if the swara directive specifies 20 swaras (tala markers do not count), then you must have 20 space separate text for the lyrics.

Empty Lyric: For certain swara, you may not want a lyric to appear in the typeset content. This is typical in certain style of notations, and also typical for actual pauses in the melody. For these, you can specify just an underscore letter (`_`) and that will translate to an empty lyric.

Here is an example of use of lyrics:

Input Format	<pre>Layout:Krithi Tala: Adi S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>
Typesetter Output	<pre>_____ ; , p d n , pa m ga , g m pa , brO va bhA ra mA ra ghu rA ,</pre>

In the above example, there are 14 swara specifiers discounting the tala markers `|` and `||`, and the higher speed specifiers `(` and `)`. As you can see there are exactly 14 lyric specifiers. Note that the first two lyric specifiers are underscores, and they translate to empty lyrics in the rendered output.

Revision History

- **v1.0:** Introduced

LyricPrefs Directive

The `LyricPrefs` directive can be set the default preferences for subsequent Lyric directives such as the default font size, and default font/text color. The syntax of the `LyricPrefs` directive is as follows:

```
LyricPrefs: attributes
```

where *attributes* is a comma separated list of one or more of the following:

- *font size*: The default point size of the lyrics as a number
- *color*: The color of the text of the lyrics. This can either be a color name such as *green*, *blue* or a RGB (red, green, blue) combination.

Here are some examples of usages of the `LyricPrefs` directive:

<code>LyricPrefs: 12</code>	Sets the default point size for lyrics to 12pt.
<code>LyricPrefs: 12, red</code>	Sets the default point size for lyrics to 12pt and color to red.
<code>LyricPrefs: 12, rgb(64,100,125)</code>	Sets the default point size for lyrics to 12pt, and the color to a a color whose red-component is 64, blue component is 100, and green component is 125. The <i>rgb(red,green,blue)</i> of color specification takes a value between 0 and 255 for each of the red, green and blue component. The maximum value of 255 for all three i.e. <i>rgb(255,255,255)</i> is white, the minimum value of 0 for all three i.e. <i>rgb(0,0,0)</i> is black. All the colors of the spectrum fall between these two and take varying values for these three components.

Note that if no `LyricPrefs` directive is given, the default font size is taken to be 10pt, and color to be black.

Revision History

- **v1.0**: Introduced

Language Directive

The `Language` directive specifies two things:

1. The language in which to render the swaras (specified by Swara Directives) and lyrics (specified by Lyric Directives).
2. The *default* language in which to render the headings and other text specified by the Heading Directives. Note that *default* here implies that this is the language that will be in effect for an heading unless the Heading directive overrides it. Hence it is possible to include headings in different languages.

Using this directive you can tell the typesetter to show your notations and headings in Sanskrit, Telugu, Tamil or Kannada. Note that for Lyric and Heading Directives, the typesetter would presume that lyrics are specified in the **Unified Transliteration Scheme for Carnatic Music Compositions**, and translate to the specified language for rendering.

If no Language directive is specified, swaras and lyrics are rendered as specified in input i.e. in English.

Here are some examples:

<pre>Tala: Adi Language: sanskrit Heading: "brOva bhAramA",12,center Heading: "(brOva bhAramA)",english,italic,9,center S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<p style="text-align: center;">ब्रौव भारमा (brOva bhAramA)</p> <hr style="width: 10%; margin: auto;"/> <p style="text-align: center;">; , प दनि, पा म गा, ग म पा, ब्रौ व भार मा र घु रा ,</p>
<pre>Tala: Adi Language: telugu Heading: "brOva bhAramA",12,center S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<p style="text-align: center;">బ్రోవ భారమా</p> <hr style="width: 10%; margin: auto;"/> <p style="text-align: center;">; , ప దని, పా మ గా, గ మ పా, బ్రో వ భార మార ఘు రా ,</p>
<pre>Tala: Adi Language: tamil Heading: "brOva bhAramA",12,center S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<p style="text-align: center;">ப்₃ரோவ பா₄ரమా</p> <hr style="width: 10%; margin: auto;"/> <p style="text-align: center;">; , ப தநி, பா ம கா, க ம பா, ப்₃ரோ வ பா₄ர மா ர கு₄ரா ,</p>
<pre>Tala: Adi Language: kannada Heading: "brOva bhAramA",12,center S: ; , (p d n ,) pa m ga , g m pa , L: _ _ brO _ va _ bhA ra mA _ ra ghu rA ,</pre>	<p style="text-align: center;">ಬ್ರೋವ ಭಾರಮಾ</p> <hr style="width: 10%; margin: auto;"/> <p style="text-align: center;">; , ಪ ದನಿ, ಪಾ ಮ ಗಾ, ಗ ಮ ಪಾ, ಬ್ರೋ ವ ಭಾರ ಮಾರ ಘು ರಾ ,</p>

Extra language attributes

The `Language` directive allows you to specify certain extra attributes in the language value, which control how the content in that language is rendered. A language directive with attribute values has the following syntax:

```
Language: your_language:attr1:attr2
```

where,

- `your_language` is one of `Sanskrit` `Telugu` `Tamil` `Kannada` `English`
- `attr1` is the first attribute, and `attr2` is the second attribute etc.

Currently the following attributes are supported:

- One of `noqual` `nohard` to specify a Tamil qualifier scheme - see **Tamil Qualifier Scheme** below for more information.
- `granhasa`, `nogranhasa`, which is applicable only to Tamil. It directs the typesetter to use or not use the ூ character for **Sa** (as in Siva, Sakthi). The default is to not use, which implies ூ₂ is used for **Sa**. Note however that the ூ character is a fairly recent addition to Unicode (in 2006) and hence it is

supported only in a few systems at this time. So you should use the `grantha` specifier only if your system can support it. Here are the current known configurations in which this character is supported:

- On Microsoft Vista, there is support at the system level for this character. The Latha font that is part of the OS supports this font. The system support is by the Unicode rendering engine called *Uniscribe*, which is in the dll *usp10.dll*. The version of the dll that ships with Vista has this support
- On Microsoft XP, if Office 2007 is installed then the Unicode rendering engine that is part of the Office 2007 at this time. However, the rendering engine that is in use for the rest of the system is an older version and does *not* support it. To make say *Firefox* or *Opera* browsers to use the Office 2007's rendering engine, find the *usp10.dll* under the Office 2007 installation directory tree, and copy the it into into the FireFox, Opera installation directory.

You would still need a Unicode font that supports the character. One such font is *Code2000*, which is a shareware font available for download at www.code2000.net.

- A percentage value in the form `n%` This controls the size of the qualifiers shown for languages like Tamil when displaying sounds not natural to the language. The qualifiers are shown as sup-scripted numbers and normally are shown in a point size that is closest 70% of the current font size. This may be too small or too big for some fonts depending on user preference. In such cases, the size of the qualifiers can be adjusted using this attribute in the `Language` directive.

For example `Language: Tamil:grantha:90%` asks the typesetter to use ஸ for **Sa**, and also show qualifiers (e.g. for **kha**, **gha** etc.) with a font whose size is closest to 90% of the current font size for the Tamil language.

Need for explicit specifiers in Lyric directives

It would be common for lyric fragments (syllables) belonging to a word to be specified as separate entities separated by spaces if they are assigned to different swaras (as would be very common) - e.g. `ninnukOri` would appear as `nin nu kO ri`. As a result, when applying interpreting a text in the Unified scheme for translation to a language, the typesetter cannot rely on spaces to imply word delimiters, and you may have to explicitly qualify certain letters so that the typesetter renders them properly. This is explained in this expectation.

Tamil: ற vs. ள.

The Tamil script has two characters for the "na" sound - ற and ள. The latter ள. cannot appear at the beginning of the word. A phrase like say *manam nOgi* is written as மனம் நோகி, wherein both characters figure. Note that at a word start like *nOgi*, ற figures. Usually, in the Unified scheme, if specify *manam nOgi* (say in a Heading Directive), the translator will correctly translate as it would rely on space as word delimiter, and know that for *nO*, it should use ற. However, when appearing as lyrics in a notation as in a Lyric Directive, this phrase could appear as `L: ma na m nO gi` where the syllables are sung as separate swaras. Here, the typesetter cannot rely on spaces to determine word breaks - as that would result in the incorrect conclusion that each syllable is a separate word! Hence when interpreting lyric text, the typesetter always assumes `na` stands for ள, and to specify ற, you need to explicitly specify it as `^n` as in `ma na m ^nO gi`.

Use n2 as opposed to n^

Note that there are two was of specifying ற: `n2` or `n^`. However, since the caret character (^) would be interpret as superscript indicator by **Wiky**, it is safer to always use `n2` in Heading directives

Tamil Qualifier Schemes

As implied earlier, if you specify the language as `Tamil:noqual` then no numeric subscript qualifiers (to distinguish *ka*, *ga* etc.) would be used. Similarly `Tamil:nohard` would specify a qualifier scheme, where qualifiers are used for the softer *ga* (but not for *ka*), and of course for *gha*, and *kha*, which do not exist in tamil.

Note that `Tamil:natural` does exist for specifying a qualifier scheme where qualifiers are used only when sounds deviate from natural tamil language rules (e.g. *gAnam* where the soft *gA* occurs at the beginning of word and is not typical of native tamil words), this scheme's effectiveness is nullified in the lyric line. This is

because the scheme relies on being able to reliably detect word beginnings, and as mentioned above, that is not possible in the lyric line. However, you should be able use it in the Heading directives.

For more information in Tamil Qualifier schemes, please refer to [The Unified Carnatic Music Transliteration Scheme - the legend](#)

Anuswara

In languages like kannada and telugu, the anuswara character ೀ is used instead #n, ~n, n, N and m in certain contexts. For example, in contexts shown in words *sa#ngIta*, *pa~nca*, *candra*, *khaNDa*, *amba* etc. the anuswara character is used instead of the #n, ~n, n and N respectively. Usually, in the Unified scheme, you can specify these words as their fair phoenetic representation rather than as *saMgIta*, *paMca*, *caMdra*, *khaMDa* and *aMba*. In fact, that is the recommended way as it facilitates easy translation to other languages which may have different anuswara rules. For example, Sanskrit has these same words but does not use the anuswara in every case.

When specifying lyrics, as indicated above, a single word may appear as multiple fragments separated by spaces. This means that you have to be careful as to whether the #n, ~n, n, N and m is specified in a separate fragment from its trailing consonant, which actually determines whether anuswara should be used. the translator will display the anuswara only if both are together. So while L: ca ndra will still be rendered correctly with anuswara (*n* and the subsequent consonant *d* are still together), L: ca n dra, L: ca n . d ra etc. will not. In such cases, you would need to explicitly specify the anuswara as in L: ca M dra, L: ca M . dra. Note however that this does compromise the ability to easily switch a notation to a different language.

Revision History

- **v1.0:** Introduced
- **v1.1:** Introduced extra language attributes - support for tamil ூ, qualifier size control

LanguageFont Directive

The `LanguageFont` directive can be used to direct the typesetter to use a specific default font for all notation content that are rendered in a specific language. If your swaras and lyrics are in telugu, and you have a mixture of headings in english, telugu and sanskrit - you can specify multiple `LanguageFont` directives to ask the typesetter to use a specific telugu font, sanskrit font and an english font. Note that the `LanguageFont` is optional, and if not specified the browser simply picks an appropriate font if available. Also whatever font you specify must be available on your system in order for the typesetter to be able to render the content properly.

The syntax of the `LanguageFont` directive is as follows:

```
LanguageFont: attributes
```

where *attributes* is a comma separated list of one or more of the following:

- *language*: The language whose font is being specified by this directive. This is one of English, Sanskrit, Telugu, Tamil or Kannada.
- *font name*: The name of the font

Here are some examples of usages of the `LanguageFont` directive:

<code>LanguageFont: Sanskrit,Akshar Unicode</code>	Sets the font to use for all Sanskrit content to the <i>Akshar Unicode</i> font.
<code>LanguageFont: English,Courier</code>	Sets the font to use for all english content to the <i>Courier</i> font.

Revision History

- **v1.0**: Introduced

Gati Directive

The `Gati` directive can be used to switch *gati/nadai* (gait) of the song. There five different *gatis* in carnatic music: the ubiquitous *catusra*, *tiSra*, *khaNDa*, *miSra*, *sankIrNa*. In the typesetter, the default *gati* is *catusra* unless you are using a tala like *Tisra Gati Adi*.

The `S` directive has the following syntax:

```
Gati: gati
```

where, `gati` is one of `tisra catusra khanda misra sankirna`

Gati switches can at the start of an akshara and thus can be at the following:

- start of a tala cycle i.e. *avarthanam*.
- start of a tala anga within a tala cycle.
- start of an *akshara* within a tala anga.

Here is an example showing gati switches. Note that the Varnam layout allows us to see the # of swaras per akshara clearly for the speed setting of 1.

```
Layout: Varnam
Tala: Adi
DefaultSpeed: 2
S: s r g m p d n s' s' n d p | m g r s |
Gati: Tisra
S: s r g r g m g m p m p d ||
Gati: Khanda
S: s r s r g r g r g m
Gati: Misra
S: s r g s r g m
Gati: Sankirna
S: s r s r g s r g m |
Gati: catusra
S: s r g m p d n s' | s' n d p m g r s ||
```

```
srgm pdns sndp mgrs | srg rgm | gmp mpd ||
```

```
srsrg rgrgm srgrgm srsrgsrgm | srgm pdns | sndp mgrs ||
```

Implicit Default Speed change associated with gati switches

(**Note:** New to 1.3)

On encountering a gati switch, the notation typesetter sometimes may switch the default speed of swaras (see **DefaultSpeed Directive**).

For example, if the default speed is 2, and the gati switches from *catusra* gati to any other gati, then default speed changes to 1. When this happens, four swaras per akshara (*catusra* gati for speed 2), switches to three, five, seven or nine i.e *tisra*, *khanda*, *miSra*, *sankIrna* gatis at speed of 1.

This is done to match what typically happens in practice, where on a gati switch, for the same speed, *tiSra* packs the least number of swaras per akshara (3), followed by *catusra* (4), then *khaNDa* (5), then *miSra* (7), and then *sankIrNa*. The automatic implicit change of default speed thus allows the gati switches in the typesetter to be more inline with their natural usage in practice.

Conversely when switching from non-*catusra* gati at a speed of 1, to *catusra* gati, the default speed is increased to 2. It should also be noted that when the gati switches back to the default gati of the song, the default speed is always restored to the original default speed of the song.

Caveat

The above speed change is done for the specific cases mentioned. If a gati switch is specified for a song in *catusra* gati with a default speed of 1 (rather than 2), then no speed change is done. This can result in *tisram* packing more swaras than *catusra*, which does not match with the natural usage of these *gatis*.

One way to get around this and achieve a switch of four swaras per akshara in *catusra gati* to three swaras per akshara in *tisra gati* (for speed = 2) is by using the special *Empty Swara Indicator* i.e. _ (underscore - see **Swara Directive**) to "blank out" half the swaras. This is illustrated in the following example:

<pre>Layout: Varnam Tala: Adi DefaultSpeed: 1 S: s r g m r g m p Gati: Tisra S: s _ r _ g _ r _ g _ m _</pre>	<pre>sr gm rg mp sr g rg m </pre>
---	---

Due to this caveat, it is recommended that for notations involving gati switches, the default speed be set to 2.

Revision History

- **v1.1:** Introduced
- **v1.3:** Introduced implicit speed changes

Songbreak Directive

The `Songbreak` directive is used to force a break in the current notation line and start a new tala cycle. It is generally not needed as a break is introduced automatically whenever a heading directive is seen. However, there may be situations where you want to explicitly break a notation in the middle of a tala cycle. Also when using "Manual" tala (see **Tala Directive**), this directive is the only way to force the typesetter to start a new line of notations.

There is no value for the directive and so its syntax is as follows:

`SongBreak`

Revision History

- **v1.4:** (Officially) Introduced