

Beginner's Guide for Hindustani Classical Music

Aditee Badge, Amruta Gokhale, Kuhoo Gupta

Computer Science and Engineering
IIT Bombay
{aditee,amruta,kuhoo}@cse.iitb.ac.in

November 28, 2004

logo

Overview

An interactive tool for learning Hindustani Classical Music

- Software intended for the naive users
- Teaches the basic concepts of the ragas

logo

Overview

An interactive tool for learning Hindustani Classical Music

- Software intended for the naive users
- Teaches the basic concepts of the ragas

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Proposed Features

- Introduction to music notations
- Raga lessons
- Simple compositions based on ragas
- Practice sessions
- Lyrics search
- Interactive music synthesis

Languages and Libraries

- Language : Java
- JFugue - Java API for music programming
 - Makes music programming incredible easy
 - Useful for applications in which music is generated at run-time



Languages and Libraries

- Language : Java
- JFugue - Java API for music programming
 - Makes music programming incredible easy
 - Useful for applications in which music is generated at run-time



Languages and Libraries

- Language : Java
- JFugue - Java API for music programming
 - Makes music programming incredible easy
 - Useful for applications in which music is generated at run-time



Languages and Libraries

- Language : Java
- JFugue - Java API for music programming
 - Makes music programming incredible easy
 - Useful for applications in which music is generated at run-time



JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

- Instrument changes
- Multiple voices
- Tempo

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

- Instrument Change
- Multiple Voices
- Tempo

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

-
-
-

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

-
-
-

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

- Instrument changes
- Multiple voices
- Tempo

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

- Instrument changes
- Multiple voices
- Tempo

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

Additional Features

- Instrument changes
- Multiple voices
- Tempo

JFugue API

Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombined
- Dynamically changing pattern of music permitted
- Music can be played at runtime, or saved in MIDI files

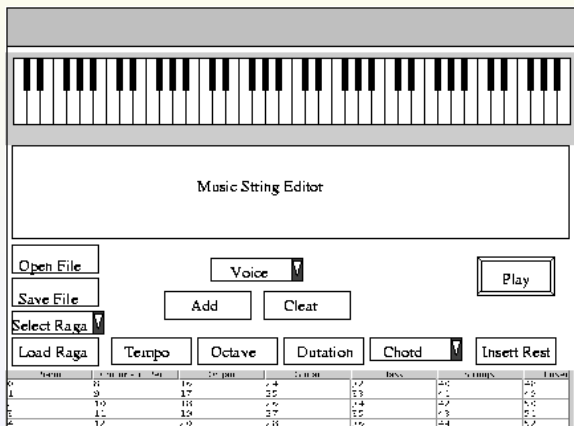
Additional Features

- Instrument changes
- Multiple voices
- Tempo

Project Description
API Description
 Limitations of JFugue
 Remarks
 References
 Figures

Features & Programming
Description of Features
 Some Technical Details & Representations
 Notes, Chords and Rests
 Tempo
 Voice
 Instrument Change
 Class Diagrams

GUI



logo

Learning Notations

- As the user plays the piano, the corresponding music pattern is displayed in the box below
- Music is heard at the same time as the piano is being played

Learning Notations

- As the user plays the piano, the corresponding music pattern is displayed in the box below
- Music is heard at the same time as the piano is being played

logo

Raga Lessons

- A Collection of some popular Ragas provided
- Each Raga stored in separate directory
- Directory contains a text file for every composition of Raga
- When user clicks on Load Raga, a window is displayed which shows all the compositions of that Raga
- User can listen to any of those compositions immediately

Raga Lessons

- A Collection of some popular Ragas provided
- Each Raga stored in separate directory
- Directory contains a text file for every composition of Raga
- When user clicks on Load Raga, a window is displayed which shows all the compositions of that Raga
- User can listen to any of those compositions immediately

Raga Lessons

- A Collection of some popular Ragas provided
- Each Raga stored in separate directory
- Directory contains a text file for every composition of Raga
- When user clicks on Load Raga, a window is displayed which shows all the compositions of that Raga
- User can listen to any of those compositions immediately

Raga Lessons

- A Collection of some popular Ragas provided
- Each Raga stored in separate directory
- Directory contains a text file for every composition of Raga
- When user clicks on Load Raga, a window is displayed which shows all the compositions of that Raga
- User can listen to any of those compositions immediately

Raga Lessons

- A Collection of some popular Ragas provided
- Each Raga stored in separate directory
- Directory contains a text file for every composition of Raga
- When user clicks on Load Raga, a window is displayed which shows all the compositions of that Raga
- User can listen to any of those compositions immediately

Lyrics Search

- User can search lyrics of a song based on raga, taal or song name
- Lyrics will be stored in the database as image file
- Entry in the database has the form : Song name,Taal, Raga, Lyrics

Lyrics Search

- User can search lyrics of a song based on raga, taal or song name
- Lyrics will be stored in the database as image file
- Entry in the database has the form : Song name,Taal, Raga, Lyrics

Lyrics Search

- User can search lyrics of a song based on raga, taal or song name
- Lyrics will be stored in the database as image file
- Entry in the database has the form : Song name,Taal, Raga, Lyrics

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

-
-
-
-

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

- Notes, Chords, and Rests
- Tempo
- Voice
- Instrument Change

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

- Notes, Chords, and Rests
- Tempo
- Voice
- Instrument Change

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

- Notes, Chords, and Rests
- Tempo
- Voice
- Instrument Change

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

- Notes, Chords, and Rests
- Tempo
- Voice
- Instrument Change

Using Features Provided in API

Music String

The Music String is a string of characters, where each group of characters represents a musical command.

Commands available in a Music String

- Notes, Chords, and Rests
- Tempo
- Voice
- Instrument Change

Octave

- MIDI is capable of playing notes spanning 10 octaves
- Default is Octave 5

Singer should be well conversed with at least 3 octaves

- Octave 4 - Mandra Saptak
- Octave 5 - Madhya Saptak
- Octave 6 - Taar Saptak

Octave

- MIDI is capable of playing notes spanning 10 octaves
- Default is Octave 5

Singer should be well conversed with at least 3 octaves

- Octave 4 - Mandra Saptak
- Octave 5 - Madhya Saptak
- Octave 6 - Taar Saptak

Octave

- MIDI is capable of playing notes spanning 10 octaves
- Default is Octave 5

Singer should be well conversed with at least 3 octaves

- Octave 4 - Mandra Saptak
- Octave 5 - Madhya Saptak
- Octave 6 - Taar Saptak

Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
- Values are placed after the octave, or directly after the note if octave is not specified
- Default is a quarter note



Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
- Values are placed after the octave, or directly after the note if octave is not specified
- Default is a quarter note



Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
- Values are placed after the octave, or directly after the note if octave is not specified
- Default is a quarter note



Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
- Values are placed after the octave, or directly after the note if octave is not specified
- Default is a quarter note

- Specifies how long to play the note
- Corresponds to "Aakaar" in Classical music

Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
- Values are placed after the octave, or directly after the note if octave is not specified
- Default is a quarter note

- Specifies how long to play the note

- Corresponds to "Aakaar" in Classical music

Duration

- Following values are allowed
 - w - whole duration
 - h - half duration
 - q - quarter duration
 - i - eighth duration
 - Values are placed after the octave, or directly after the note if octave is not specified
 - Default is a quarter note
- Specifies how long to play the note
 - Corresponds to "Aakaar" in Classical music

Velocity

JFugue can use the velocity to indicate

- How hard a note is struck
 - Set the attack velocity using the *a* indicator, followed by a value from 0 to 127
- How quickly the note is released
 - Set the decay velocity using the *d* indicator, followed by a value from 0 to 127
- Default attack velocity and decay velocity for each note is 64

Taal is a cycle of beats, starting with a stress point called the Sam and ending with a release point called the Khali. JFugue can use the velocity feature to render taal.

Velocity

JFugue can use the velocity to indicate

- How hard a note is struck
 - Set the attack velocity using the *a* indicator, followed by a value from 0 to 127
- How quickly the note is released
 - Set the decay velocity using the *d* indicator, followed by a value from 0 to 127
- Default attack velocity and decay velocity for each note is 64

Taal is a cycle of beats, starting with a stress point called the Sam and ending with a release point called the Khali. JFugue can use the velocity feature to render taal.

Velocity

JFugue can use the velocity to indicate

- How hard a note is struck
 - Set the attack velocity using the *a* indicator, followed by a value from 0 to 127
- How quickly the note is released
 - Set the decay velocity using the *d* indicator, followed by a value from 0 to 127
- Default attack velocity and decay velocity for each note is 64

Taal is a cycle of beats, starting with a stress point called the Sam and ending with a release point called the Khali. JFugue can use the velocity feature to render taal.

Velocity

JFugue can use the velocity to indicate

- How hard a note is struck
 - Set the attack velocity using the *a* indicator, followed by a value from 0 to 127
- How quickly the note is released
 - Set the decay velocity using the *d* indicator, followed by a value from 0 to 127
- Default attack velocity and decay velocity for each note is 64

Taal is a cycle of beats, starting with a stress point called the Sam and ending with a release point called the Khali. JFugue can use the velocity feature to render taal.

Notes

- Note command begins with the note name or chord root, or the rest character: A, B, C, D, E, F, G, or R
- Represent a sharp or flat note by using the # and b characters
 - e.g. F#, Bb
- Specify octave & duration information after the note name
 - e.g. F#5q (5 is the 5th octave & q is the quarter duration)

Notes

- Note command begins with the note name or chord root, or the rest character: A, B, C, D, E, F, G, or R
- Represent a sharp or flat note by using the # and b characters
 - e.g. F#, Bb
- Specify octave & duration information after the note name
 - e.g. F#5q (5 is the 5th octave & q is the quarter duration)

Notes

- Note command begins with the note name or chord root, or the rest character: A, B, C, D, E, F, G, or R
- Represent a sharp or flat note by using the # and b characters
 - e.g. F#, Bb
- Specify octave & duration information after the note name
 - e.g. F#5q (5 is the 5th octave & q is the quarter duration)

Chords

- Specify the root of the chord & its structure
- All notes in a chord are played using the same instrument and in the same voice
- Some chord structures recognized by JFugue are
 - maj - Major
 - min - Minor
- Chord indicator goes directly after the root, and before the octave or duration
 - e.g. C-major, 5th octave, quarter note would be Cmaj5q

Generally it is "Saa Ga Pa" played simultaneously

Chords

- Specify the root of the chord & its structure
- All notes in a chord are played using the same instrument and in the same voice
- Some chord structures recognized by JFugue are
 - maj - Major
 - min - Minor
- Chord indicator goes directly after the root, and before the octave or duration
 - e.g. C-major, 5th octave, quarter note would be Cmaj5q

Generally it is "Saa Ga Pa" played simultaneously

Chords

- Specify the root of the chord & its structure
- All notes in a chord are played using the same instrument and in the same voice
- Some chord structures recognized by JFugue are
 - maj - Major
 - min - Minor
- Chord indicator goes directly after the root, and before the octave or duration
 - e.g. C-major, 5th octave, quarter note would be Cmaj5q

Generally it is "Saa Ga Pa" played simultaneously

Chords

- Specify the root of the chord & its structure
- All notes in a chord are played using the same instrument and in the same voice
- Some chord structures recognized by JFugue are
 - maj - Major
 - min - Minor
- Chord indicator goes directly after the root, and before the octave or duration
 - e.g. C-major, 5th octave, quarter note would be Cmaj5q

Generally it is "Saa Ga Pa" played simultaneously

Chords

- Specify the root of the chord & its structure
- All notes in a chord are played using the same instrument and in the same voice
- Some chord structures recognized by JFugue are
 - maj - Major
 - min - Minor
- Chord indicator goes directly after the root, and before the octave or duration
 - e.g. C-major, 5th octave, quarter note would be Cmaj5q

Generally it is "Saa Ga Pa" played simultaneously

Rests

- Inserts rest period in the music string
- Specify the duration in the same way as of notes
 - e.g. Rw (w is the whole duration)

Rests

- Inserts rest period in the music string
- Specify the duration in the same way as of notes
 - e.g. Rw (w is the whole duration)

Combining Notes

- Plus (+) character can be used to play multiple notes in at the same time (in harmony)
 - e.g. C5q+E5q+G5q will play the C, E, and G notes, quarter duration, at the same time
- Underscore (_) character can be used to play notes in order (in melody) when the melody is being played with a harmony
 - Used to play multiple notes at the same time with mixed durations
 - e.g. C5h+E5q_G5q will play C note for half duration simultaneously with E followed by G, each for quarter duration

Combining Notes

- Plus (+) character can be used to play multiple notes in at the same time (in harmony)
 - e.g. C5q+E5q+G5q will play the C, E, and G notes, quarter duration, at the same time
- Underscore (_) character can be used to play notes in order (in melody) when the melody is being played with a harmony
 - Used to play multiple notes at the same time with mixed durations
 - e.g. C5h+E5q-G5q will play C note for half duration simultaneously with E followed by G, each for quarter duration

Sample Note Commands

Some examples are -

- **A** - Play an A note, fifth octave (default), quarter duration (default)
- **Rw** - A whole-duration rest
- **Cmaj3w** - Play a C-major chord, octave 3, whole duration
- **D4q+F4q+A4q** - Plays the notes D, F, and A together
- **C5w+E5h_G5h+Dmaj3w** - Plays a C note, fifth octave, whole duration; at the same time, plays an E, fifth octave, half duration, followed by a G, fifth octave, half duration; at the same time, plays a D-major chord, third octave, whole duration.

Sample Note Commands

Some examples are -

- **A** - Play an A note, fifth octave (default), quarter duration (default)
- **Rw** - A whole-duration rest
- **Cmaj3w** - Play a C-major chord, octave 3, whole duration
- **D4q+F4q+A4q** - Plays the notes D, F, and A together
- **C5w+E5h_G5h+Dmaj3w** - Plays a C note, fifth octave, whole duration; at the same time, plays an E, fifth octave, half duration, followed by a G, fifth octave, half duration; at the same time, plays a D-major chord, third octave, whole duration.

Sample Note Commands

Some examples are -

- **A** - Play an A note, fifth octave (default), quarter duration (default)
- **Rw** - A whole-duration rest
- **Cmaj3w** - Play a C-major chord, octave 3, whole duration
- **D4q+F4q+A4q** - Plays the notes D, F, and A together
- **C5w+E5h_G5h+Dmaj3w** - Plays a C note, fifth octave, whole duration; at the same time, plays an E, fifth octave, half duration, followed by a G, fifth octave, half duration; at the same time, plays a D-major chord, third octave, whole duration.

Sample Note Commands

Some examples are -

- **A** - Play an A note, fifth octave (default), quarter duration (default)
- **Rw** - A whole-duration rest
- **Cmaj3w** - Play a C-major chord, octave 3, whole duration
- **D4q+F4q+A4q** - Plays the notes D, F, and A together
- **C5w+E5h_G5h+Dmaj3w** - Plays a C note, fifth octave, whole duration; at the same time, plays an E, fifth octave, half duration, followed by a G, fifth octave, half duration; at the same time, plays a D-major chord, third octave, whole duration.

Sample Note Commands

Some examples are -

- **A** - Play an A note, fifth octave (default), quarter duration (default)
- **Rw** - A whole-duration rest
- **Cmaj3w** - Play a C-major chord, octave 3, whole duration
- **D4q+F4q+A4q** - Plays the notes D, F, and A together
- **C5w+E5h_G5h+Dmaj3w** - Plays a C note, fifth octave, whole duration; at the same time, plays an E, fifth octave, half duration, followed by a G, fifth octave, half duration; at the same time, plays a D-major chord, third octave, whole duration.

Tempo

Tempo - How fast or slow the song should be played

- Tempo value represents "Pulses Per Quarter" (PPQ), i.e. how many "pulses", or clock cycles, to give a quarter note
- Default value is 120
- Have to specify the tempo once in the music string
- The command is a T, followed by a number from 0 to infinity.
e.g. T120

Tempo

Tempo - How fast or slow the song should be played

- Tempo value represents "Pulses Per Quarter" (**PPQ**), i.e. how many "pulses", or clock cycles, to give a quarter note
- Default value is 120
- Have to specify the tempo once in the music string
- The command is a T, followed by a number from 0 to infinity.
e.g. T120

This corresponds to laya. Can play dugun, chaugun etc.

Tempo

Tempo - How fast or slow the song should be played

- Tempo value represents "Pulses Per Quarter" (**PPQ**), i.e. how many "pulses", or clock cycles, to give a quarter note
- Default value is 120
- Have to specify the tempo once in the music string
- The command is a **T**, followed by a number from 0 to infinity.
e.g. **T120**

This corresponds to laya. Can play dugun, chaugun etc.

Tempo

Tempo - How fast or slow the song should be played

- Tempo value represents "Pulses Per Quarter" (**PPQ**), i.e. how many "pulses", or clock cycles, to give a quarter note
- Default value is 120
- Have to specify the tempo once in the music string
- The command is a **T**, followed by a number from 0 to infinity.
e.g. **T120**

This corresponds to laya. Can play dugun, chaugun etc.

Tempo

Tempo - How fast or slow the song should be played

- Tempo value represents "Pulses Per Quarter" (**PPQ**), i.e. how many "pulses", or clock cycles, to give a quarter note
- Default value is 120
- Have to specify the tempo once in the music string
- The command is a **T**, followed by a number from 0 to infinity.
e.g. **T120**

This corresponds to laya. Can play dugun, chaugun etc.

Voice

- Gives the ability to play multiple melodies at the same time
- The command is a V, followed by a number from 0 to 15. e.g. V5
- There are 16 voices, numbered 0 through 15.

Can have two voices , one playing the chord and other playing song over this chord

Voice

- Gives the ability to play multiple melodies at the same time
- The command is a **V**, followed by a number from 0 to 15. e.g.
V5
- There are 16 voices, numbered 0 through 15.

Can have two voices , one playing the chord and other playing song over this chord

Voice

- Gives the ability to play multiple melodies at the same time
- The command is a **V**, followed by a number from 0 to 15. e.g.
V5
- There are 16 voices, numbered 0 through 15.

Can have two voices , one playing the chord and other playing song over this chord

Voice

- Gives the ability to play multiple melodies at the same time
- The command is a **V**, followed by a number from 0 to 15. e.g.
V5
- There are 16 voices, numbered 0 through 15.

Can have two voices , one playing the chord and other playing song over this chord

Instrument Change

- This command tells JFugue to play the following notes with the given instrument number or name
- The command is an I, followed by either a number from 0 to 127, or the name of an instrument enclosed in brackets
- e.g. I9 or I[Guitar]

Instrument Change

- This command tells JFugue to play the following notes with the given instrument number or name
- The command is an **I**, followed by either a number from 0 to 127, or the name of an instrument enclosed in brackets
- e.g. I9 or I[Guitar]

Instrument Change

- This command tells JFugue to play the following notes with the given instrument number or name
- The command is an **I**, followed by either a number from 0 to 127, or the name of an instrument enclosed in brackets
- e.g. **I9** or **I[Guitar]**

Instruments supported in API

Name	Code
Instrument names	
PIANO	0
HARMONICA	22
GUITAR	24
VIOLIN	40
FLUTE	73
SITAR	104
Percussion names	
HAND_CLAP	34
LOW_BONGO	61

Class Diagrams

Pattern

```
add(Pattern pattern)
add(String musicString)
addElement(JFugueElement element)
getMusicString()
setMusicString(String s)
```

Voice

```
getVoice()
musicString()
setVoice(byte voice)
```

Limitations of JFugue

- No continuity between two notes
- Changing of base frequency of a note not provided
- No support for Tabla

Limitations of JFugue

- No continuity between two notes
- Changing of base frequency of a note not provided
- No support for Tabla

Limitations of JFugue

- No continuity between two notes
- Changing of base frequency of a note not provided
- No support for Tabla

Remarks

- Uses the domain knowledge of music extensively
- Requires a separate domain expert to explain the music concepts to the programmer
- Programmer has to take care of all the technicalities & map all of them by using appropriate features of JFugue

Remarks

- Uses the domain knowledge of music extensively
- Requires a separate domain expert to explain the music concepts to the programmer
- Programmer has to take care of all the technicalities & map all of them by using appropriate features of JFugue

Remarks

- Uses the domain knowledge of music extensively
- Requires a separate domain expert to explain the music concepts to the programmer
- Programmer has to take care of all the technicalities & map all of them by using appropriate features of JFugue

References

- www.jfugue.org
- www.batish.com/archives/arcgloss.html

