**OutputFile Format (.plux)**

A music line (1 bar per line) begins with a ‘|’. Empty bar lines are ignored. There are 8 notes in a bar irrespective of the beats.

Bar line data is :

1-5 select string

a-z set fretting and pluck that string this note

+ add hammer on 1 (multiple)

- sub pull off 1 (multiple)

/ slide 1 (multiple)

= Change the width of the slide from 2 notes to 1 note.

. go to next note.

(Chord) set display chord

The + - / modifiers mean the fretting specifies the start position**.** So 2/4 would be 2//.

An information list begins with a ‘.’ and has the format .<name> := <value> (no spaces)

Supported information items are : (default values in brackets)

beats default beats per bar (4)

tempo default beats per minute play speed (80)

step amount tempo increases every play when on speed-up (4)

tuning current tuning. Supports gcgcd and (gdgbd)

others are ignored.

Lines beginning with // are comments and ignored.

**Input File Format (.blue)**

Lines beginning with // are ignored as comments

<key> := <value> are passed through and may affect operation. Defaults are as above.

All other lines are tune information. These are bars seperated by |

Throughout the tune there is a current fretting and current stringing, indicated what fret and string is played on each note (half beat). These are initialised to fretting 00000 and string

None x 8.

Each bar has eight beats. They are numbered from one if the intro key is set to Y. The text is processed with the input fretting and string, and produces an output fretting and stringing when processed.

Fretting : 0123456789 t e w h u f s [represents fretting 0-16]

#fffff Sets the fretting, and nothing else. It is not required to set all

five values, any missing will default to 0. Fretting is as above.

$ssssssss Sets the strings (does not apply retrospectively). S values can be

1-5 or . (for don’t play). There must be eight of them.

\* play one half beat combination of fretting/string, if S is . It is a rest

. play a rest.

(chord) sets a chord for this note.

xxxfff play the current note – x is string skipping, f frets. So x45 would

mean string 1 – nothing string 2 – 4th fret string 3 - 5th fret others

nothing. This is a half beat note. These overwrite the current

fretting.

^^^ vvv /// = Adds pull offs,hammer ons and slides to the last note in this bar.

This is by default 2 notes, e.g. there is a rest added here as well.

However, if the = is present the half beat advance is suppressed.

This only works if only one string is played in the previous note,

otherwise it throws an error.

& This advances to note 2,4 or 6, by adding half beats. This is a

shorthand facility for Wayne Erbsen’s Music.

Modifiers

As well as the ‘as is’ music, multiple versions can be produced. These are done with the option.<name> := definitions key.

A definition is a sequence of modifiers, seperated by a semicolon.

A modifier is a list of bars which can be a combination of \* (all), n (single) or a-b (range) seperated by commas. The modifier is applied once to each bar.

Pre rendering modifiers

These are done before the text is converted to bars.

Each bar has an alternate text which when set overrides that in a tune. This modifier is to <text> and sets that alternate text. This allows specific changes to be made.

Post rendering modifiers

pinch for each half bar, add a pinch if there is a single note.

drone for each half bar, add 2 drones if there is a double note.

chord any chord is played, replacing whatever is in the ‘slot’ (always displayed)

after a chord is played, do not assume fretting.

roll xxxxxxxx/nnn Play one of the rolls below, only on strings nnn

altthumb play x251x251 for both half bars (3 and 4 only), only if 2 single notes.

forward play x15x15x1 (for the 3rd x take the note from half beat 4), 2-4 only.

backward play x21512x1 (3-5 only)

foggy play x1x15x15 (2-4 only)

Rendering

Has a descriptor. Takes fretting and stringing.

Takes a list of [modifiers] or [] the default, for the standard tune).

1) Update the “alternate text” - clear them all and for any to xxxx modifiers set the alternat text.

2) Convert to an 8 x 6 array of fret playing (the 6th is the # notes in the first 5). Each of this array is type Note which represents 1 fret and string sound, which can be modified.

3) Make the post render modifications.

4) Convert to a final code entry.

Naming : if the original song is called Bile em cabbage down, the modified one adds the option name in parenthesis e.g. bile em cabbage down (pluck).

Rendering has to be done in sequence, because of the in/out fretting and stringing requirement. Prior to that, each bar just has a number and its default text. It is not converted to notes on creation or checked syntactically.

Once a bar has been rendered it can be shown as a text sort of bar.

**Different Display Usage**

Minimising the interface for different displays.

*float Interface(command as integer,bar ref as bar,dispInfo ref as dispInfo,params as IFParams)*

Initialise

Load Resources for the display.

Create

(param 1 : bars in tune)

Draw background, set up for specific tune (if full page tab display sizes for example)

Create empty display information records for each bar which are all non-displayed

Destroy

Remove background

(Hide all display resources is done by the caller)

Visible

(bar,param 1 : position in music in bars float)

Returns non-zero if the given bar is partially displayed at that music position.

Show

(bar,param 1 : position in music in bars float)

Move the bar at a given position, creating if necessary, modifying if necessary.

Hide

if drawn , Free up all the bars display resources.

ID allocation (10000 onwards)

+0...+19 Note#1 0..4 note specific 5x3 for each displayed note.

+20..+39 Note#2

+40..+59 Note#3

+140..+159 Note#8

+160..+199 Reserved Allocated for general use.