Using BarFly Macros

In the early days, several BarFly users asked me why the program does not play rolls. The answer is fairly straightforward. The tilde symbol ~ which is normally used to represent a roll in Irish music is actually meant as a general ornament. In other words, what it really means is "play some decoration here which is appropriate to the tradition within which you are playing, to the instrument you are using, and to the position of the note within the tune." While that causes no problems for human musicians it is a tall order for a computer program. If I were to build a set of Irish fiddle rolls into the program, I would surely receive complaints from people who are busy transcribing Patagonian Crwth music, asking why the program plays their ornaments wrongly. Furthermore, even within the context of Irish dance music, the exact timing, and even the pitch of the notes involved in ornaments is a matter of controversy and a subject of endless argument on the newsgroups and mailing lists. I have no intention of getting involved in such arguments; after all I'm just an iggerant guitar player who wouldn't know a roll or a crann if it crawled up his nose.

For all those reasons, BarFly now includes a system of macros which allow the user to specify exactly what is meant by "~G3", or indeed by any other combination of symbols. There are lots of examples here, because that's the easiest way to explain things, but please don't assume that they are in any way "correct". The macro libraries supplied will get you going and show you how the system works, but you are encouraged to write your own. If you come up with better ornaments than those which I have supplied, or if you construct macro libraries for different instruments or different musical traditions please mail them to me and I will include them in the next program update for the benefit of all users.

Static Macros

BarFly currently supports two kinds of macro, called Static and Transposing. Static macros are the easiest to understand. You define a static macro by writing into the tune header something like this:

```
 m: ~G3 = G{A}G{F}G
```

(Actually the m: is a field identifier, and like all such should be written hard up against the left hand margin - in this case I don't want the program to recognise it as a real macro, hence the space)

When you play the tune, the program searches the tune header for macro definitions, then does a search and replace on its internal copy of the text before passing that to the parser which plays the tune. Every occurence of ~G3 in the tune is replaced by G{A}G{F}G, and that is what gets played. Only ~G3 notes are affected, ~G2, ~g3, ~F3 etc are ignored.

You can put in as many macros as you want, and indeed, if you only use static macros you will need to write a separate macro for each combination of pitch and note-length. Here is an example:

(Before you play this tune, open the Player Preferences dialog and check the Enable Macros check box to hear the effect.)

```
X:1
T:Apples in Winter
R:jig
E:9
m: ~g2 = {a}g{f}g
m: ~D2 = {E}D{C}D
I:speed 350
M:6/8
K:D
```

G/2A/2|BEE dEE|BAG FGE|~D2D FDF|ABc ded|
BEE BAB|def ~g2 e|fdB AGF|GEE E2:|
d|efe edB|ege fdB|dec dAF|DFA def|
[1efe edB|def ~g2a|bgb afa|gee e2:|
[2edB def|gba ~g2e|fdB AGF|GEE E2||

Here I have put in two static macros, since there are two different notes in the tune marked with a tilde.

A static macro definition consists of four parts:
¥ the field identifier m:
¥ the target string - e.g ~G3
¥ the equals sign
¥ the replacement string - e.g. G{A}G{F}G

The target string can consist of any string up to 31 characters in length, except that it may not include the equals sign '=' (because the equals sign marks its end) or the letter 'n' (for reasons which will become obvious later). You don't have to use the tilde, but of course if you don't use a legal combination of abc, other programs will not be able to play your tune. In older versions of BarFly, the target string had to include a single note symbol, but that is no longer the case. Because you cannot use the equals sign, you cannot target a note with a natural sign on it.

The replacement string consists of any legal abc text up to 200 characters in length. It's up to you to ensure that the target and replacement strings occupy the same time interval (the program does not check this). Both the target and replacement strings may have spaces embedded if necessary, but leading and trailing spaces are stripped off so m:~g2={a}g{f}g is perfectly OK, although less readable.

Transposing macros

If your tune has ornaments on lots of different notes, and you want them to all play with the same ornament pattern, you can use transposing macros to achieve this. Transposing macros are written in exactly the same way as static macros, except that the note symbol in the target string is represented by 'n' (meaning any note) and the note symbols in the replacement string by other letters (h to z) which are interpreted according to their position in the alphabet relative to n. So, for example I could re-write the static macro m: ~G3 = G{A}G{F}G as a transposing macro m: ~n3 = n{o}n{m}n. When the transposing macro is expanded, any note of the form ~n3 will be replaced by the appropriate pattern of notes. Notes of the form ~n2 (or other lengths) will be ignored, so you will have to write separate transposing macros for each note length.

Here's an example:

X:2
T:Down the Broom
R:reel
Q: 350
M:C|
m: ~n2 = (3o/n/m/ n % One macro does for all four rolls
K:ADor
EAAG~A2 Bd|eg~g2 egdc|BGGF GAGE|~D2B,D GABG|
EAAG ~A2 Bd|eg~g2 egdg|eg~g2 dgba|gedB BAA2:|
~a2ea agea|agbg agef|~g2dg Bgdg|gfga gede|
~a2 ea agea|agbg ageg|dg~g2 dgba|gedB BA A2:|

A transposing macro definition consists of four parts:
¥ the field identifier m:

¥ the target string - e.g ~n3
¥ the equals sign
¥ the replacement string - e.g. n{o}n{m}n

The target string can consist of any string up to 31 characters in length, except that it must conclude with the letter 'n', followed by a number which specifies the note length, and may not include an equals sign.

The replacement string consists of any legal abc text up to 200 characters in length, where note pitches are defined by the letters h - z, the pitches being interpreted relative to that of the letter n. Once again you should ensure that the time intervals match. Where a transposing macro is applied to a note with an accidental on it, the accidental will be placed on the first instance of the principal note in the expansion. This is usually the right thing to do; however, you may find sometimes that you need an accidental on another note too. In this case you will have to write a static macro to deal with that particular case.

Global macros

If you have a file with 500 tunes in it, inserting macro definitions into each tune header would involve a great deal of work, so BarFly allows you to put macros in two other places and have them apply to all the tunes involved.

¥ You can put macro definitions into the file header - that is to say anywhere between the beginning of the file and the X: field of the first tune. These definitions will then apply to all the tunes in that file.

¥ You can place macro definitions in a separate file, and have the program apply them to all tunes played. This opens up the possibility of creating ornament libraries, each containing sets of macros appropriate to a different instrument or musical tradition.

¥ If you re-define a macro it will override the previous definition, so for example you might have a set of global transposing macros for fiddle rolls, but in a particular case you want a triplet to be played instead. Place a static macro defining the triplet in the header of that tune, and for that note only a triplet will be played instead of the globally-defined roll. The order of precedence for macros is as follows : tune header macros > file header macros > ornament library macros. In any list of macros, later definitions take precedence over earlier ones.

Using Ornament Libraries.

You will find a folder named 'Macros' alongside the program, and in it are several files. On startup, the program searches for this folder and installs each of the files within it. The file names appear on the Global Macro File submenu, off the Edit menu. Use this menu to select which of the macro files to use. Older versions of BarFly required you to install the macros individually using the Add a FileÉ command at the top of this menu. This option is no longer available.

Writing your own Ornament Libraries.

This is extremely simple. Ornament libraries are just text files containing macro definitions. Any text which is not preceded by a m: field is ignored, so you can put in as much documentation as you like. If you need to put comments on the same line as a macro use the % comment symbol in the usual way. You can even put a tune at the end of the file as an example, and if you open up the file the macros within it will be interpreted as file header macros and applied only to that tune.

Odds and ends.

The Highlight Note Played option temporarily turns macros off. This is because if they were on, the text which was being interpreted would no longer correspond to that on the screen, and the moving highlight would be invalid.

The Viewer Preferences dialog also has an Enable Macros checkbox, and if you turn this on you can view and print the music with the macros written out in full. You will probably want to keep this turned off normally as it slows down the drawing a little, and makes the music look unnecessarily complicated.

The Expand Macros command in the Edit menu creates a new copy of the current tune in a new window, with all the macros written out in full. You can use this to debug macros you have written yourself, and it's also useful if you want to send the tune to someone who is not using BarFly, so they can hear it play as intended.

Don't change the name of the Macros folder, or the program will complain that it can't find it on startup.

Note that BarFly only reads the file header when opening a file, so if you choose to put macros here, you will have to save, close and re-open the file before the program will take any notice of them.

Finally...

Macros can be used for other purposes besides playing ornaments; over-use of them can, however, make the abc very hard to read, and abc is supposed to be a human-readable language.

Here's an extreme example - see if you can figure out what's going on here.

```
X:3
T:The Twelve Days of Christmas
M:none
m: H = AA | A2 dd d2 cd | efge f3 ||
m: I = g |a2 bg fd e2 | d3 ||
m: J = a2 | ef g2
m: K = f ||
m: L = a2 b^g a4 || agfe d2 g2 | B2 d2 edcB A2 |
m: 2J = JJ
m: 3J = JJJ
m: 4J = JJJJ
m: M = HI HJKI H2JKI H3JKI HLKI HJLKI H2JLKI H3JLKI H4JLKI H4JJLKI H2J4JLKI HJ2J4JLKI
K:D
M
```

If you want to see the music display for this tune, you must turn on Enable Macros in the Viewer preferences dialog, otherwise the program will tell you there's no tune there.

Back to the table of contents:

file://:Table of Contents