

→ Read This First

Though the usefulness of NSF files is pretty limited, there are a few people out there like myself who would like to use them in Game Maker. While this is very convenient for easy looping and small file size, it's really not easy to get up and running smoothly. In fact, it was one of the most frustrating aspects of the projects I used it in. So, I compiled the following notes in the hopes that it will help people get NSF files playing the way they want them faster and easier than I did.

It should be noted that all of this worked for me, but I cannot guarantee that it will work for everyone, or that even the behavior of things described will match what others may experience. The following is based solely on my own experiences.

The following notes are for using the best NSF DLL I've found and then more advanced topics, so I assume you're using Shaltif's S-Winamp extension and already have a basic working knowledge of it: <http://gmc.yoyogames.com/index.php?showtopic=280606>

→ Using a Working DLL

After trying every NSF DLL I could find and being met each time with bad sound quality, abundant crashes and annoying pop-up windows, I finally stumbled upon the one included with this file and for unknown reasons, it seems to work pretty well. It does on occasion crash, but very rarely- quite possibly determined by the NSF file you try to play.

The one snag with this DLL is, like many, it pops something up when you use it for the first time. However, you can't see whatever it pops up before it's gone, so it's really like it just takes window priority away from Game Maker.

This only happens to the first track you go to play, so to get around this, insert the following code at the earliest possible moment in the game, before you actually go to play any music:

```
{
    window_set_stayontop(true);
    //Use your Winamp function to play an NSF file
    //Use your Winamp function to stop the NSF instantly
    window_set_stayontop(false);
}
```

This worked well for me and finally gave me a working NSF system, so I hope it works the same for you.

→ Playing the Desired NSF Track

Needless to say it's quite an annoyance when you want to use S-Winamp to play track 06 of an NSF, and not surprisingly, it only plays track 01 when you load up the file. This would of course be no problem if S-Winamp could allow you to specify which track of the NSF you want to play, but alas... it doesn't. So that means more work for you.

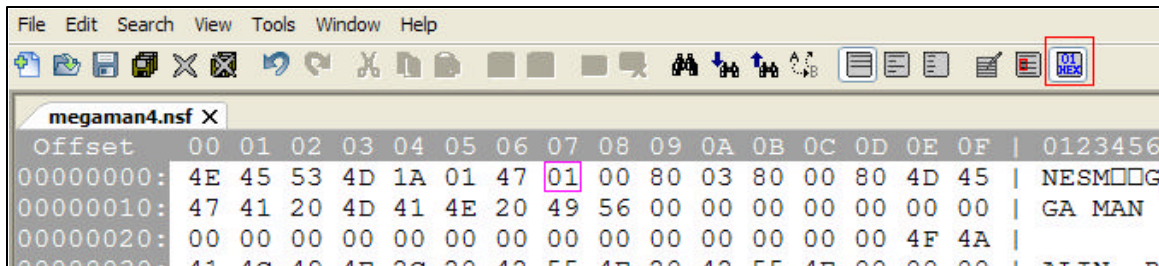
I imagine the most common and easiest way to play the NSF track you want is by using the "NSF Track Ripper" program (Google is your friend). This can take tracks from an NSF and rip them as individual files, which can then be played correctly through S-Winamp.

While this usually works fine, there are some tracks that just simply won't rip properly, and you'll end up with an out-of-sync mess each time you try. When this happens your only option is to hex edit the full NSF file so that it plays the track you want first instead of track 01.

This is actually simpler than it may sound. First step is to obtain a free hex editor if you don't have one (Google is your friend again- I used "MadEdit"), and open up the NSF file.

Now, I know nothing about hex editing, so I won't pretend I do. I'll just be explaining what I know to work for changing the starting NSF track.

From MadEdit's interface, this is what I see when I open up the NSF file for Mega Man 4:



Note the red box around the "01 HEX" button in the upper right- if using MadEdit make sure this is pressed to see the file correctly.

Also note the pink box under "07" that contains the value "01". This is where all the magic happens; as you may guess, "01" is indicating track 01. So if you change that to "06", the NSF file will then open with track "06". Simple as can be, right?

But wait, you wanted track 10, and it ended up playing track... 16? From 10 and up, this is where hex starts to get confusing, and you need to translate your numbers into hex numbers. Here's a conversion chart, with the "Dec" column containing the track numbers you want and the "Hex" column containing the numbers you need to put in the NSF file: <http://www.ascii.cl/conversion.htm>

For obtaining NSF files, check Zophar's Domain: <http://www.zophar.net/music/nsf.html>

(Huge thanks to JMC47 and NMN for showing me how simple this hex edit actually is)

→ Resetting Your Game with S-Winamp

The S-Winamp extension doesn't play nice with Game Maker's `game_restart()` function, which is a shame if you want to give players that option. So to work around it, we'll use Game Maker save states instead.

I'm assuming your S-Winamp will be initialized in the first room of the game, preferably a room that only serves to initialize variables and such before immediately skipping on to the next room, where the opening or title screen kicks in. It's in this second room's creation code that you should place the DLL pop-up catching code mentioned earlier, and in addition to that, make a save state like this:

```
{
    game_save('reset');
}
```

Then, in your game's reset button event, execute the following code:

```
{
    winamp_stop();
    game_load('reset');
}
```

Because the initialization room was skipped over before the player could even see it, they'll never know that they were reset to the second room of the game. The effect is just like the `game_restart()` function while allowing S-Winamp to keep breathing.

→ Where did that S-Winamp Object Go?

You may or may not be aware, but S-Winamp creates a new persistent object for itself as a controller when you initialize it. Sometimes you may want to work with that object in some way; for example, if you use Game Maker's deactivate and activate functions and want to deactivate everything outside of the view, this will also deactivate S-Winamp and cause all sorts of misery for your game, so you need to keep that object activated regardless of where it is.

In order to get a handle on that object, execute the following code after you've called the `winamp_create()` function:

```
{
    execute_string('global.winid = __newobject234.id;');
}
```

****Note** that the number 234 will be whatever the newest object number would be; for example, if the last new object you created in Game Maker was initially called "object567", then the number in the code would be 568.

Once the above code has been executed, you can then use the global variable to work with the object, like so:

```
{
    instance_activate_object(global.winid);
}
```

→ Thanks for Reading

I hope these notes and the included DLL make getting NSF files up and running in Game Maker an easier task for you than it was for me. No credit necessary either way.

-Blyka

<http://blyka.legends-station.com/>

Please do NOT redistribute or claim credit for this file. Share with others by all means if you'd like, but in doing so please link people to the proper download link; among other reasons, this will ensure everyone gets the most up-to-date version of it:

<http://blyka.legends-station.com/dl.php?id=nsfexample>