How To

by "Cajun" David Richard

EV Plug-ins - The Final Chapter

Now that the EV Plug-in FAQ is complete, I wanted to write one more article on plug-in creation and troubleshooting. However, this may not be the last. If I find that folks are having some trouble in a particular area, we may see another article on the subject. Regardless, we hope that the EV Plug-in FAQ will help all of you out there who are baffled by all of the computer gobbledegook you find when you open the EV Bible.

For starters, the EV Plug-in FAQ is available on the Web from our Escape Velocity Add Ons page (http://www.ambrosiasw.com/Products/Add_Ons/EV_AddOns.html) as well as somewhere on the vast Escape Velocity Web site (http://www.escape-velocity.com). It is labeled "EV Plug-in package" and contains the FAQ text file, the EV Bible and the Starter Plug-in. This is a file that anyone can use to begin creating a plug-in. Also, you can find an HTML version of the EV Plug-in FAQ at: http://www.AmbrosiaSW.com/FAQs/EV_Plug-in_FAQ.html.

Areas of confusion

There are a couple of areas that are bound to cause some confusion when creating plug-ins. These are ID conventions and using HEX numbers. We have attempted to provide information in the EV Plug-in FAQ to help, however, I would like to cover these two topics here because of their importance to making plug-ins.

ID conventions

One thing to keep in mind here is that no ID will be lower than 128. This is a standard practice that all programmers use. ID's below 128 are used by the system, and setting an ID lower that 128 is asking for trouble.

So, when you open a resource group in ResEdit, the lowest ID you will ever see will be 128. Sounds easy, right? Well, the trouble comes when in some fields of some resources in EV, you need to place the ID of a related resource. For example, if you have a weap resource for ammunition, the AmmoType field of the weap resource holds the ID of the outf resource (the gun). However, the field in the weap resource starts counting at 1, and resources start counting at 128, so you need to subtract 128 from the outf resource ID to reference it properly. There are formulas in the FAQ to help you.

HEX Numbers

HEX numbers are another way to illustrate binary data. They represent a string of bits that can be turned either on or off. Each bit can tell a program about certain characteristics (often called "flags") and can be used in a variety of different ways. What all this means is that you can set a number of "bits" and hold them in one variable, or in EV's case, one field.

In the EV Bible, you will see different characteristics that can be set in a "Flags" field. Flags can either be 4 or 8 digits long, and are preceded by the characters "0x."

To combine different characteristics, you need to add together the hex flags for each desired characteristic. To add the hex flags together, add up the columns. The trick is, though, you don't carry digits. Rather, 10 = A, 11 = B, 12 = C, 13 = D, 14 = E, 15 = F. So, say you've got the following flags to add:

```
0x0001
0x0002
0x0008
0x0010
0x0040
0x0800

0x085B
```

The last column adds to 11, so put in B) Add the columns and substitute a letter for any value that is over 9. In the EV Bible, the '0x' before the number indicates that it is a HEX number, and is ignored when you add them together. However, when you enter a HEX number in a ResEdit field, replace the '0x' with a '$'.

Troubleshooting your Plug-in

One of the more common problems that is reported concerns a message from EV during the loading process. This message says "EV ran out of memory during a critical operation." There are two main reasons for this: Either EV actually ran out of memory, or EV thinks it ran out of memory due to a badly written plug-in or some corruption in one of EV's files.

If you add sounds or graphics (including sprites) to the game, you need to give EV more memory to accommodate these new items. There are many plug-ins that will require you to increase the memory allocation for the EV application. The Read Me that comes with the plug-in should tell you what you need.

However, sometimes EV will present this message due to an ill written plug-in. I have also found that corruption of files, including the EV Preferences file, can also be the source of this problem. It is usually a good idea to have a copy of the EV Installer on your machine just in case you need to reinstall a fresh copy of the game. If you are developing a plug-in, and you receive this message when you place your plug-in into the Plug-ins folder, do the following, in order, launching EV after each step:

- Remove the plug-in

- Remove the preferences file
- Reinstall a fresh copy of the game
- If you still have problems, remove the Escape Velocity License file (have your code handy).

  This is the only file remaining that can be corrupted.

Computer programs are drones - they will do exactly what you tell them to do. If you tell them to do the wrong thing, they will do the wrong thing. If you find that your plug-in is not behaving as expected, then the best thing to do is to go through the plug-in with a fine toothed comb. There is bound to be something that is incorrect.

Also, use your brain, and try to follow the logic of the game. If you are getting the wrong graphics for a ship, then check all of the factors associated with that ship's graphic. If you know exactly what is going wrong, then you should be able to find the error.

Thanks...

I would like to thank some people who have helped me with the assembly of the EV Plug-in FAQ. Jason Meltzer was kind enough to take it upon himself to write a great deal of text that made this project much easier to complete. Also helping out were Patrick Delahanty, Bill Hazelrig, Jay Tingley, Charles Lawrence. Thanks guys! Hopefully we will see many more plug-ins as a result of this project.

As always, comments can be directed to help@AmbrosiaSW.com.