Equipoise-Image

#include>
macro authoring utility

version 1.0, 10 March 2002

<u>Background</u>

Equipoise-Image ('**EqPi**') is a Macintosh Classic application that extends the widely-used image processing packages *NIH Image* and *Object-Image*. **EqPi** can run macro programs written in the *NIH Image* and *Object-Image* macro languages without change, and adds several capabilities not available in either of these applications.

To download Equipoise-Image, visit the **Equipoise Imaging, LLC** home page at:

http://www.eqpi.net/eqpi/

<u>Description: #include> macro authoring utility</u>

While it is advantageous for many compiled language development environments to provide compiler directive statements that aid during precompilation, interpreted languages (such as the Image-family macro language) typically do not. In particular, none of the Image-family macro language variants provides an *include* compiler directive. This capability is provided via a separate macro program referred to as the *#include> macro authoring utility*. The utility allows macro authors to load code from a commonly-used function, procedure, or other routine directly into their macros by using a short *#include>* command that indicates the name of a file containing that code. This facilitates and promotes the reuse of standard program modules.

<u>Instructions</u>

<u>1. Preparation</u>

The *#include> macro authoring utility* makes use of information stored in two places:

(i) a folder entitled *'Includes'*, where files to be included are stored. This folder must exist when the *#include>* command is used.

(ii) the *'EqPi_prefs'* file, a text file in *System Folder/Preferences* folder of the startup volume. This file defines the path to the *'Includes'* folder, using the following syntax:

#include>StartupVolume:folder:subfolder:...:subfolder:

A single *'Includes'* folder is typically created in the folder containing the **EqPi** application.

2. Command syntax

The *#include>* command is used with the following syntax in macro language programs:

{#include>filename}

filename is the name of the file that the author wishes to include in the macro program in which the *#include>* command is used. When this line is encountered in a macro program, the *#include> macro authoring utility* will insert the text content of the indicated file into the macro program. Note that the *#include>* command must be enclosed in *{*comment braces*}*. A macro program may contain multiple *#include>* commands.

3. Terminator line

Macro programs that use the *#include>* command must end with the single line:

{!}

4. Use

A. Open and load the *#include> macro authoring utility* text file. The utility is itself a macro language program.

B. Open a text window containing the macro language program under development. Type the *#include>* command into the macro language program under development wherever you would like to have code inserted. Be certain that the command correctly names the file to be included and that this file exists in the *'Includes'* folder.

C. Select the window containing the macro language program under development, and enter the [F12] function key. A new text window named after the development window (but with '.incl' appended) will be created, the macro language program text will be copied to this window, and *#include>* commands in the program will be replaced by the content of files that they name.

5. Example

The path to the *'Includes'* folder stored in the 'EqPi_prefs' file on the startup volume 'HD' is:

```
HD:EqPi folder:Includes:
```

Here is text from a sample macro language program in the 'DevSample' window:

```
macro 'DevSample';
begin
        PutMessage('yep');
end;
{#include>func AppSpc}
{#include>func I2S}
{!}
```

Here is the 'func AppSpc' file content:

```
function AppSpc(theStr: string; theInt: integer;): string;
{appends spaces to a string to reach a fixed length}
var
        int1: integer;
        str1: string;
begin
        str1 := theStr;
        for int1 := 1 to (theInt - length(theStr)) do begin
                str1 := concat(str1, ' ');
        end;
 AppSpc := str1;
end;
```

Here is the 'func I2S' file content:

```
function I2S(theInt: integer;): string;
{converts as integer to string with no leading spaces}
var
        str1: string;
begin
```

```
        str1 := concat(theInt);
        While (ord(str1)<48) do begin
                Delete(str1, 1, 1);
        end;
        I2S := str1;
end;
```

Execution of the *#include> macro authoring utility* (with the 'Development Macro' window active) creates a new 'DevSample.incl' window with the following content:

```
macro 'DevSample';
begin
        PutMessage('yep');
end;
function AppSpc(theStr: string; theInt: integer;): string;
{appends spaces to a string to reach a fixed length}
var
        int1: integer;
        str1: string;
begin
        str1 := theStr;
        for int1 := 1 to (theInt - length(theStr)) do begin
                str1 := concat(str1, ' ');
        end;
 AppSpc := str1;
end;
function I2S(theInt: integer;): string;
{converts as integer to string with no leading spaces}
var
        str1: string;
begin
        str1 := concat(theInt);
        While (ord(str1)<48) do begin
                Delete(str1, 1, 1);
        end;
        I2S := str1;
end;
{!}
```