Configuring the PostScript Device

The **FSC_PSConfig** program is a program for configuring the IDL PostScript device. The program is similar to (but much more powerful than) **PS_Form**, an older program offered by Fanning Software Consulting. The **FSC_PSConfig** program is written as an object.

In addition to configuring the PostScript device, the **FSC_PSConfig** program allows you to keep track of the current state of the PostScript device at all times. European users will be pleased to know that support for the A4 page and centimeter units are built into the program and can easily be set as program defaults. One of the most useful features of the **FSC_PSConfig** program is its ability to collect user input via a graphical user interface, which--on a Windows machine with a 24-bit graphics display--looks similar to the illustration below.

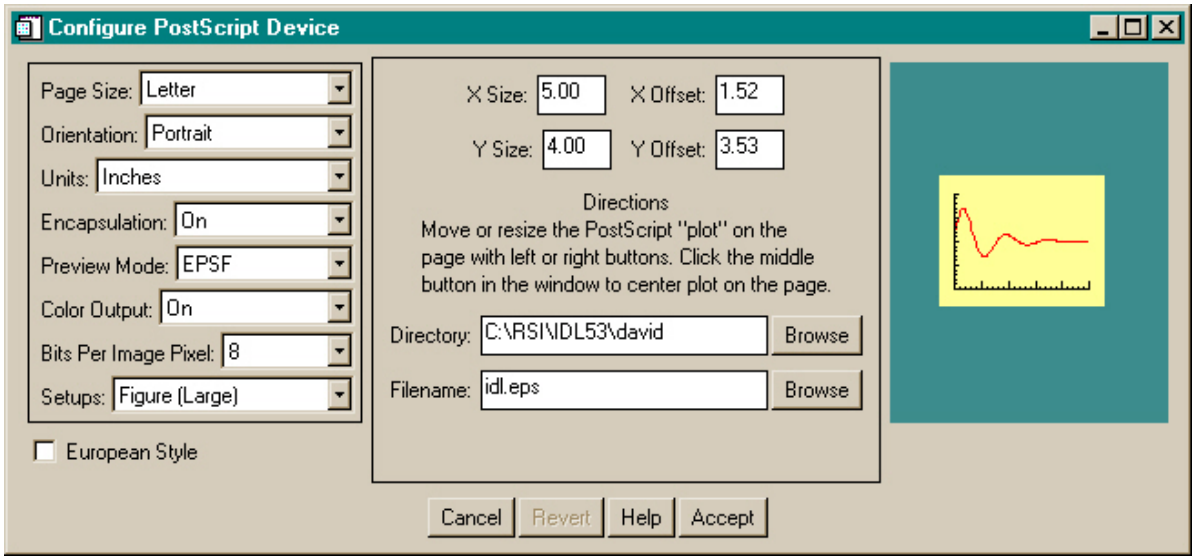


Table of Contents

Program Overview

How you use the **FSC_PSConfig** program is completely up to you. It has been designed to accommodate different programming styles. For example, it can be called from the IDL command line or from within other IDL programs. Typically, the program is used by initializing the object (you can have as many of the objects as you like, so each program can have its own PostScript configuration setup), gathering user input via the built-in Graphical User Interface (GUI), configuring the PostScript device, and drawing your PostScript plot. Here is a typical sequence of commands. Each step will be explained in detail below, but this is a short overview of a typical session. (Users uninterested in the power of the **FSC_PSConfig** object can access most of the functionality through the **PSConfig** wrapper function described below.)

Initializing the Object

The first step is to initialize the object. There are a number of keywords that can be used to set properties of the **FSC_PSConfig** object, but suppose you are going to leave it up to the user to determine exactly how the PostScript device should be configured. Then you might simply initialize the object like this:

```
psObject = Obj_New("FSC_PSConfig")
```

If you typically display PostScript output on an A4 page and prefer size and offset units to be displayed in centimeters rather than inches, then you can initialize the object with the **European** keyword set, like this.

```
psObject = Obj_New("FSC_PSConfig", /European)
```

The A4 page and centimeter units are now the default for this object. Selecting a European style can also be selected via a check box on the graphical user interface.

Gathering User Input

Next, you might gather the user's input into how he or she would like the PostScript device to be configured. You can do this with a graphical user interface in either modal (blocking) or non-modal mode. Suppose you wish the program to wait for the user's input. You would invoke the graphical user interface in blocking mode with this command:

```
psObject->GUI
```

Configuring the PostScript Device

The real purpose of the **FSC_PSConfig** object is to return keywords that are appropriate for the PostScript device. The way the PostScript device is configured is to make the PostScript device the current graphics device, and then pass these keywords to the PostScript device via the **Device** command using the "keyword inheritance" mechanism and the **_Extra** keyword. The keywords are obtained from the object by calling the **GetKeywords** method. A typical sequence of commands looks like this:

```
thisDevice = !D.Name
Set_Plot, "PS"
Device, _Extra=psObject->GetKeywords()
```

Drawing the PostScript Graphics

The PostScript graphics are now written to the PostScript file in the normal way. (Here a simple **Plot** command is used, but the graphics command or commands can be as elaborate as you like.)

```
Plot, data
```

Cleaning Up

The final step is to clean up. Note that you do not have to destroy the object, as shown here. In fact, you may want to keep the object around for the entire IDL session, so you always know the current configuration of the PostScript device. Repeated calls to invoke the graphical user interface will show the current state of the PostScript device. Typical clean-up commands will be similar to these:

```
Device, /Close_File
Set_Plot, thisDevice
Obj_Destroy, psObject
```

[Return to Contents]

Using the Program

The **FSC_PSConfig** program is used by calling the program's object methods. This section provides a detailed description of the object methods available to the user. (There are a number of other methods in the object that are used internally to run the program and should not normally be accessed by the user of the program. These are not described here, although they are extensively documented in the source code.)

Initializing the Program

The object is initialized by creating an object of the **FSC_PSCONFIG** object class:

```
psObject = Obj_New("FSC_PSConfig")
```

At the time the object is initialized, various keywords can be used to set properties of the object. These keywords are typically (but not exclusively) the same keywords that can be used to configure the PostScript device (i.e., the keywords appropriate for the **Device** command when the PostScript device is the current graphics device). Here is a complete description of the **INIT** method keywords you can use.

- **AvantGarde** - Set this keyword to select the AvantGarde font. Turned off by default.
- **Bits_per_Pixel** - The number of image bits saved for each image pixel: 2, 4, or 8. The default is 8.

- **Bold** - Set this keyword to select the Bold font style. Turned off by default.
- **BookStyle** - Set this keyword to select the Book font style. Turned off by default.
- **Bookman** - Set this keyword to select the Bookman font. Turned off by default.
- **Color** - Set this keyword to select Color PostScript output. Turned off by default.
- **Courier** - Set this keyword to select the Courier font. Turned off by default.
- **Debug** - Set this keyword to turn traceback information on for error messages. This is useful for debugging program code. Turned off by default.
- **DefaultSetup** - Set this keyword to the "name" of a default style. Current styles (you can easily create and add your own to the source code) are the following:
  - **"System (Portrait)"** - The normal "default" system set-up. Also, **"System"**.
  - **"System (Landcape)"** - The normal "default" landscape system set-up.
  - **"Centered (Portrait)"** - The window centered on the page. Also, **"Center"** or "Centered".
  - **"Centered (Landscape)"** - The window centered on the landscape page. Also, **"Landscape"**.
  - **"Square (Portrait)"** - A square plot, centered on the page.
  - **"Square (Landscape)"** - A square plot, centered on the landscape page.
  - **"Figure (Small)"** - A small encapsulated figure size, centered on page. Also, **"Encapsulated"** or **"Encapsulate"**.
  - **"Figure (Large)"** - A larger encapsulated figure size, centered on page. Also, **"Figure"**.
  - **"Color (Portrait)"** - A "centered" plot, with color turned on. Also, **"Color"**.
  - **"Color (Landscape)"** - A "centered" landscape plot, with color turned on.
- **Demi** - Set this keyword to select the Demi font style. Turned off by default.
- **Directory** - Set thie keyword to the name of the starting directory. The current directory is used by default.
- **Encapsulate** - Set this keyword to select Encapsulated PostScript output. Turned off by default.
- **European** - Set this keyword to indicate "european" mode (i.e., A4 page and centimeter units). Turned off by default.
- **Filename** - Set thie keyword to the name of the PostScript file. In the source code version the default is "idl.ps", but in the shareware version this keyword cannot be set and is always "coyote.ps".
- **FontSize** - ; Set this keyword to the font size. Between 6 and 36. Default is 12.
- **FontType** - ; Set this keyword to indicate the type of fon you want selected. Hershey Fonts: -1, Hardward Fonts: 0, and True-Type Fonts: 1.
- **Helvetica** - Set this keyword to select the Helvetica font. This is the default selection.
- **Inches** - Set this keyword to indicate sizes and offsets are in inches as opposed to centimeters. Set by **European** keyword by default.
- **Italic** - Set this keyword to select the Italic font style. Turned off by default.
- **Isolatin** - Set this keyword to select ISOlatin1 encoding. Turned off by default.
- **Landscape** - Set this keyword to select Landscape page output. Portrait page output is the default.
- **Light** - Set this keyword to select the Light font style. Turned off by default.
- **Medium** - Set this keyword to select the Medium font style. Turned off by default.
- **Name** - Set this keyword to the "name" of the object. This can be anything you like. Objects that have different names can display their graphical user interfaces simultaneously. The object name appears on the title of the graphical user interface.
- **Narrow** - Set this keyword to select the Narrow font style. Turned off by default.
- **Oblique** - Set this keyword to select the Oblique font style. Turned off by default.
- **PageType** - Set this keyword to the "type" of page. Possible values are:
  - **"Letter"** - 8.5 by 11 inches. (Default, unless the **European** keyword is set.)
  - **"Legal"** - 8.5 by 14 inches.
  - **"Ledger"** - 11 by 17 inches.
  - **"A4"** - 21.0 by 29.7 centimeters. (Default, if the **European** keyword is set.)
- **Palatino** - Set this keyword to select the Palatino font. Turned off by default.
- **Preview** - Set this keyword to select Preview mode: 0 is "none", 1 is "EPSI", or 2 is "EPSF".
- **Schoolbook** - Set this keyword to select the Schoolbook font. Turned off by default.
- **Set_Font** - Set this keyword to the name of a font passed to PostScript with Set_Plot keyword.
- **Symbol** - Set this keyword to select the Symbol font. Turned off by default.
- **Times** - Set this keyword to select the Times font. Turned off by default.
- **TrueType** - Set this keyword to select True-Type fonts. Turned off by default.
- **XOffset** - Set this keyword to the X Offset. Uses "System (Portrait)" defaults. (Note: offset calculated from lower-left corner of page.)
- **XSize** - Set this keyword to the X size of the PostScript "window". Uses "System (Portrait)" defaults.
- **YOffset** - Set this keyword to the Y Offset. Uses "System (Portrait)" defaults. (Note: offset calculated from lower-left corner of page.)
- **YSize** - Set this keyword to the Y size of the PostScript "window". Uses "System (Portrait)" defaults.
- **ZapfChancery** - Set this keyword to select the ZapfChancery font. Turned off by default.
- **ZapfDingbats** - Set this keyword to select the ZapfDingbats font. Turned off by default.

For example, to select a centered, color, encapsulated PostScript file, using the Times Bold PostScript font, and named *myplot.eps*, you can initialize the object like this:

```
psObject = Obj_New("FSC_PSConfig", /Color, /Encapsulate, /Times, /Bold,
Filename="myplot.eps")
```

[Return to Contents]

Description of the Graphical User Interface

The program's graphical user interface is invoked by calling the object's **GUI** method, like this:

```
psObject->GUI
```

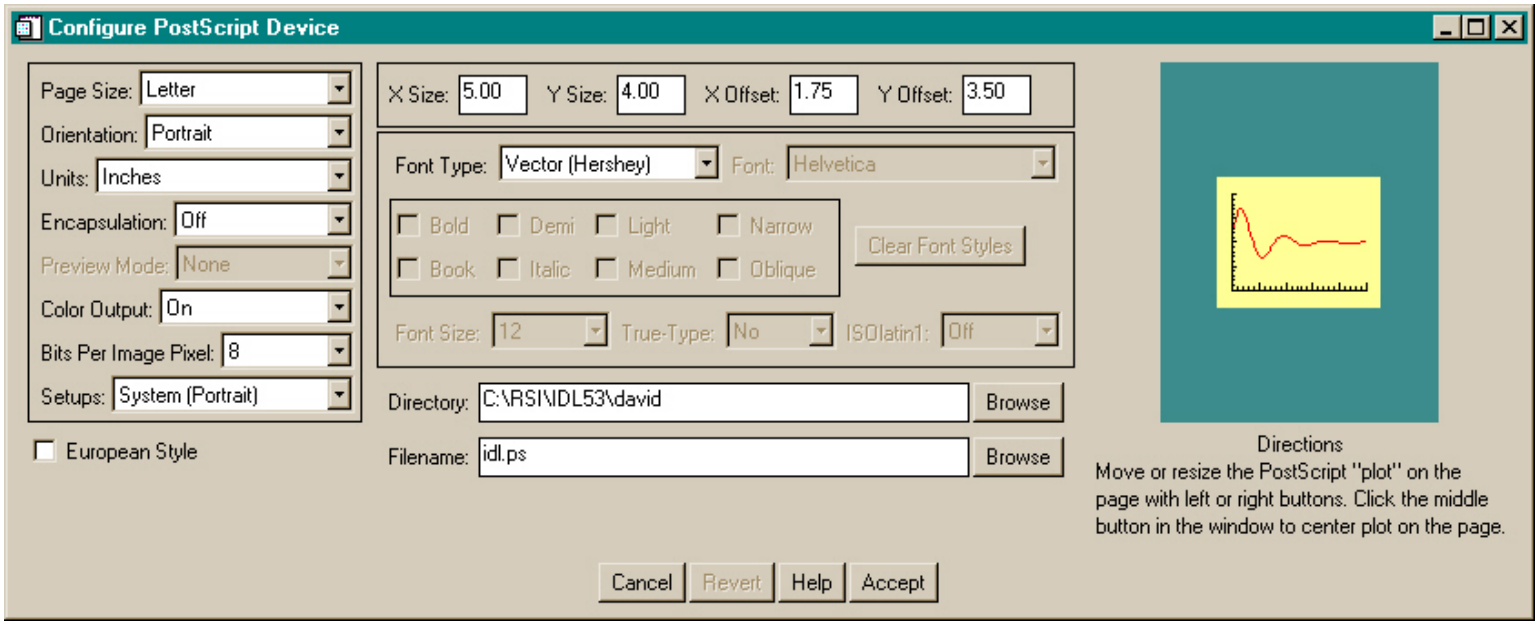
Keywords are available with this method that control how the graphical user interfaces appears and what properties it has. Called as above, the interface is a blocking widget. This means the IDL command line will block and user will be unable to enter IDL commands at the command line until the widget is dismissed by clicking either the *Cancel* or the *Accept* button. Valid keywords for this method are these:

- **Cancel** - Set this output keyword to a named variable that will be set to 1 if the user selected the *Cancel* button, and to 0 if the user selected the *Accept* button. Note that this value only has meaning when the widget is is blocking (or modal) mode. In non-modal mode, the value is always set to 0.

  ```
  psObject->GUI, Group_Leader=event.top, Cancel=cancelled
  ```

- **FontInfo** - This keyword is set if you want the user to be able to configure the PostScript device with font information as well. Please see the description of how to use font information below, but the graphical interface will look similar to this.

  ```
  psObject->GUI, FontInfo=1
  ```



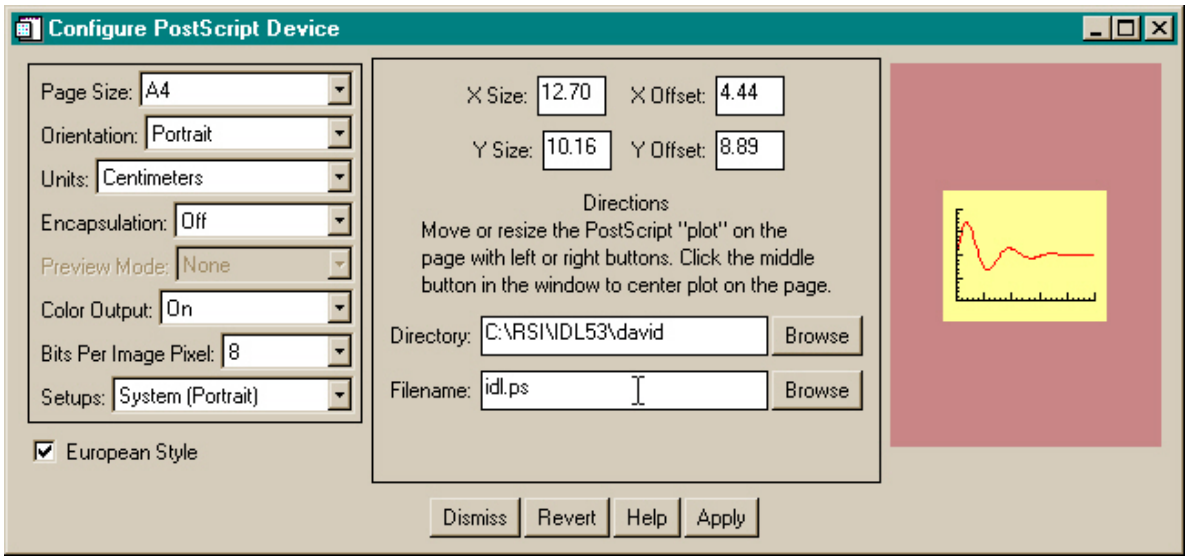
  Users have the ability to select Hershey, Hardware, or True-Type fonts, as well as many font "properties".

- **Group_Leader** - Set this keyword to the identifier of a widget which is the group leader for the graphical user interface. This keyword *must* be set if you want modal (as opposed to just blocking) behavior from the program. In other words, you will probably *always* set this keyword if you are calling the GUI from within another widget program. For example, from within a widget program, the graphical user interface should be invoked like this:

  ```
  psObject->GUI, Group_Leader=event.top
  ```

- **NoBlock** - This keyword is set if you want non-modal, and non-blocking behavior from the program. This is the mode you would use, for example, if you want the graphical user interface to stay on the display so you can change the PostScript set-up at will. Notice that in this mode the *Cancel* button is named *Dismiss* and the *Accept* button is named *Apply*.

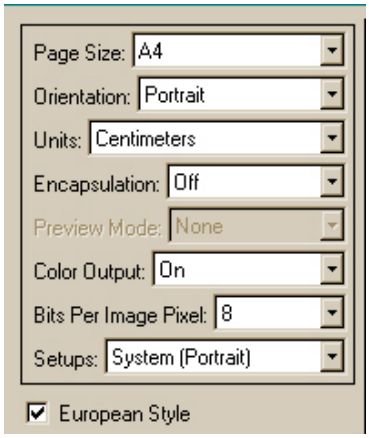
  ```
  psObject->GUI, /NoBlock
  ```



In non-modal mode, selecting the *Apply* button will save the current PostScript configuration. Note that the PostScript graphics window changes background color if the current configuration shown on the form is not currently saved. (The program is shown in its unsaved condition in the illustration above.) Upon saving the configuration, the background color will change back to its default color. The *Revert* button will revert to the last saved configuration.

The Left Panel

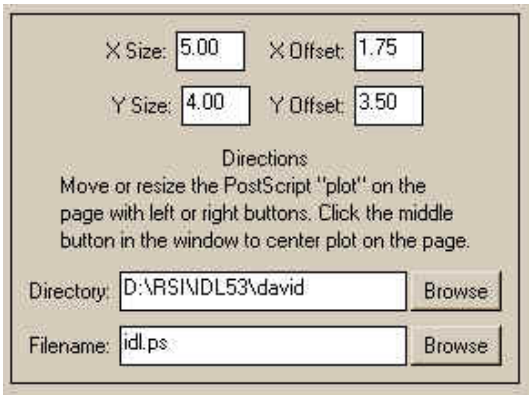
The left panel of the graphical user interface contains eight droplist widgets and looks like this. A description of each droplist follows the illustration.



- **Page Size** - This droplist controls the page size. It should be set to the type of output you intend to print, since offsets and sizes will be based on the page size selected. This will be set to "Letter" by default, unless the **European** keyword has been set, in which case the "A4" option is the default.
- **Orientation** - This droplist controls whether the page will be in portrait or landscape oreintation. The **Portrait** and **Landscape** Device keywords are set from this droplist.
- **Units** - This droplist controls whether the size and offsets are measured in inches or centimeters. The **Inches** Device keyword is set from this droplist.
- **Encapsulation** - This droplist controls whether **Encapsulated** Device keyword is set. If encapsulation is turned *Off*, then the **Preview Mode** droplist is insensitive.
- **Preview Mode** - This droplist selects the value of the **Preview** Device keyword. It is sensitive to user input only if the *Encapsulation* droplist is turned *On* "None" sets the keyword to 0, "ESPI" sets the keyword to 1, and "ESPF" sets the keyword to 2. If you wish IDL to create a preview image for Macintoshs or PCs, then you should choose the "ESPF" option.
- **Color Output** - This droplist controls whether **Color** Device keyword is set.
- **Bits Per Image Pixel** - This droplist controls the value of the **Bits_Per_Pixel** Device keyword.
- **Default Setups** - This droplist selects the various default set-ups, which change multiple program settings at once. The choices in this droplist are easily modified by the user in the source code. Typically they would be choices in use at your facility.
- **European Style** - This checkbox controls the Page Size and Units keywords. If checked, an A4 page size is used, and centimeters are used for Units.

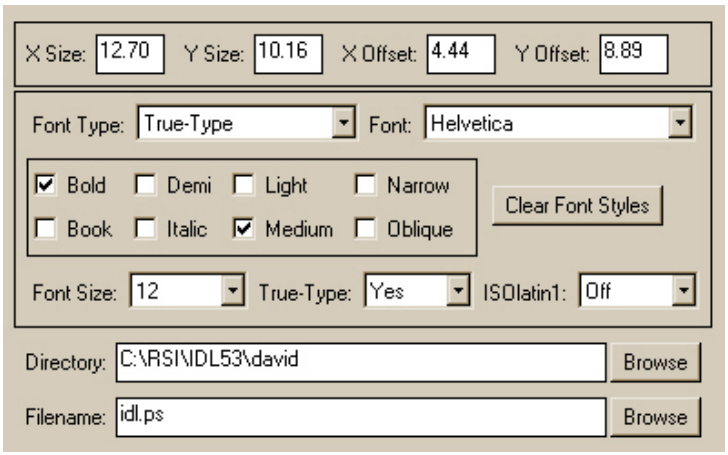
The Center Panel

The center panel of the graphical user interface contains the size and offset information, directions for operating the PostScript "plot", and fields to indicate the output directory and file names. It looks similar to the illustration below. An explanation of each item follows the illustration.



- **X Size** - This text field contains the X size of the PostScript "window". Its value is used to set the **XSize** Device keyword. After entering a value in this field you *must* enter a carriage return in the field for the new information to be transmitted to the plot window in the right panel.
- **Y Size** - This text field contains the Y size of the PostScript "window". Its value is used to set the **YSize** Device keyword. After entering a value in this field you *must* enter a carriage return in the field for the new information to be transmitted to the plot window in the right panel.
- **X Offset** - This text field contains the X offset of the PostScript "window". Its value is used to set the **XOffset** Device keyword. After entering a value in this field you *must* enter a carriage return in the field for the new information to be transmitted to the plot window in the right panel. Offsets are always calculated from the lower-left corner of the page regardless of the orientation, unlike the way you configure the PostScript device. The complexities of PostScript are handled behind the scenes and are not passed along to unsuspecting end-users.
- **Y Offset** - This text field contains the Y offset of the PostScript "window". Its value is used to set the **YOffset** Device keyword. After entering a value in this field you *must* enter a carriage return in the field for the new information to be transmitted to the plot window in the right panel. Offsets are always calculated from the lower-left corner of the page regardless of the orientation, unlike the way you configure the PostScript device. The complexities of PostScript are handled behind the scenes and are not passed along to unsuspecting end-users.
- **Directory** - The name of the directory where the PostScript file is to be written. The *Browse* button following the text field calls the IDL **Dialog_Pickfile** command with the **Directory** keyword set for directory selection.
- **Filename** - The name of the output PostScript file. The *Browse* button following the text field calls the IDL **Dialog_Pickfile** command for file selection. The *Directory* and *Filename* values are put together to constuct the value of the Device **Filename** keyword.

If the **FontInfo** keyword is set when the graphical user interface is invoked, the center panel may also contain graphical elements allowing user input into font selections. These additional fields are described below the illustration.



- **Font Type** - This droplist sets the font type. The other font options are available only if the *Font Type* droplist selection is set to something other than the "Vector (Hershey)". Note that the *Font Type* droplist default selection depends upon the value of the **!P.Font** system variable when the graphical user interface is invoked, although it can be also be set via a **FontType** keyword when the grahical user interface is invoked. For example, the illustration above was produced with this syntax:

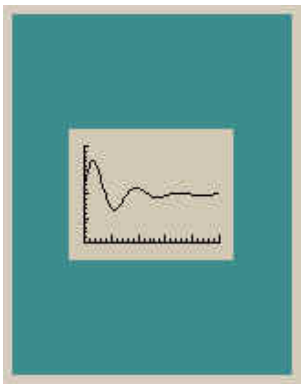
```
IDL> psObject->GUI, /FontInfo, FontType=1
```

- **Font** - This droplist allows selection of one of the 10 font families available on most PostScript printers. A selection here sets the corresponding Device keyword.
- **Font Style Box** - This non-exclusive base allows you to set the eight font styles that can be set for the PostScript device. These font styles are applied to the font selection described above to modify it with the selected style. Note that which font styles are available for which font family is not controlled by IDL, but is a function of the fonts installed on your computer or printer. Thus, you must use this information carefully. The corresponding Device keyword is set according to each style selection.
- **Clear Font Styles** - This button clears all the font style attributes selections described above.
- **Font Size** - This droplist allows you to select the font size in point sizes from 6 to 36. The default size is 12. The **FontSize** Device keyword is set to this value.
- **True-Type** - This droplist sets the Device **True_Type** keyword. If it is turned on true-type fonts are used for output if they are available.

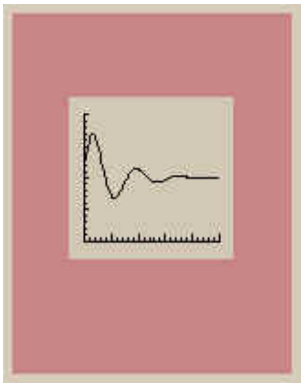
- **ISOlatin1** - This droplist sets the Device **ISOlatin1** keyword. If it is turned on isolatin encoding is used for output. This allows output in many languages other than English.

The Right Panel

The right panel of the graphical user interface contains the representation of the PostScript page. This window will change shape and size as different page sizes and orientations are selected. It looks similar to the illustration below. (Note that the colors you see here will only appear on 24-bit displays. On 8-bit displays the colors are less exciting.)

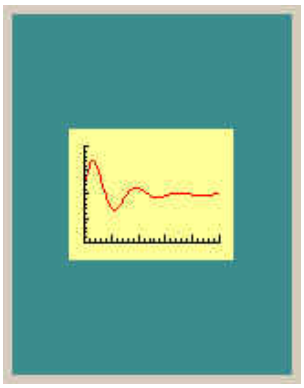


The "plot" in the window can be moved and resized by the user. It represents the "window" on the PostScript page where graphics will be drawn. As such, it controls the **XSize**, **YSize**, **XOffset**, and **YOffset** Device keywords. The plot window can be moved by clicking somewhere inside it and dragging. If you locate the cursor near an edge of the plot, you can click and drag to resize the window. Cursor changes will occur to indicate whether you are in drag or resize mode. Notice as you do that the size and offset fields in the right panel are updated with the new window sizes and offsets. Notice that the background color of the window changes if the current configuration shown on the form hasn't been saved and the program is in non-blocking mode. It will look similar to the illustration below. The backgound color does not change if the graphical user interface is in blocking mode. (Non-blocking mode is selected with the **NoBlock** keyword when the graphical user interface is invoked.)



The plot window can be centered in the page by clicking the middle mouse button anywhere inside the PostScript page representation. Note that the middle mouse button can be simulated if you don't have a three-button mouse. For example, on PCs a *Ctrl-Click* is interpreted as a middle mouse click. *Option-Click* on a Macintosh will do the same.

If the *Color Output* droplist is turned *On*, then the plot is shown in color as a visual clue to the color property.



The Action Button Panel

The action button panel looks different, depending upon whether the graphical user interface is is blocking (modal) mode or in non-blocking mode. Here is what it looks like in non-blocking mode.

Here is how it appears in blocking mode.

Notice that in blocking mode the *Dismiss* button is named *Cancel* and the *Apply* button is named *Accept*, but they have similar functions. The only real difference is that the *Accept* button destroys the widget (as do the *Dismiss* and *Cancel* buttons), while the *Apply* button does not.

- **Dismiss** or **Cancel** - This button destroys the graphical user interface. The last saved configuration of the graphical user interface is in effect.
- **Revert** - This button restores the last saved configuration of the graphical user interface. A configuration is saved when the *Apply* or *Accept* button is selected.
- **Help** - This button provides a Help message to the user and points the user to the program documentation. This action may have been customized for your facility.
- **Apply** or **Accept** - This button saves the current configuration of the graphical user interface. If the button is the *Accept* button, the graphical user interface is also destroyed and the block released.

  Note that the name of this button can be changed by using the **Accept_Button_Name** keyword when the graphical user interface is invoked. For example, you might want to name the button "Print". You would do it like this:

```
IDL> psObject->GUI, Accept_Button_Name='Print'
```

The buttons would now look like this:

[Return to Contents]

Setting Program Properties

The current configuration of the **FSC_PSConfig** object can be set at any time by using the **SetProperty** method of the object. The keywords that can be used to set properties are nearly identical to the keywords avaiable for the **INIT** method. Here is a list of the keywords and thier meaning.

- **AvantGarde** - Set this keyword to select the AvantGarde font.
- **Bits_per_Pixel** - The number of image bits saved for each image pixel: 2, 4, or 8.
- **Bold** - Set this keyword to select the Bold font style.
- **BookStyle** - Set this keyword to select the Book font style.
- **Bookman** - Set this keyword to select the Bookman font. T
- **Color** - Set this keyword to select Color PostScript output.
- **Courier** - Set this keyword to select the Courier font.
- **DefaultSetup** - Set this keyword to the "name" of a default style. Factory styles (you can easily create and add your own to the source code) are the following:
  - **"System (Portrait)"** - The normal "default" system set-up. Also, **"System"**.
  - **"System (Landcape)"** - The normal "default" landscape system set-up.
  - **"Centered (Portrait)"** - The window centered on the page. Also, **"Center"** or "Centered".
  - **"Centered (Landscape)"** - The window centered on the landscape page. Also, **"Landscape"**.
  - **"Square (Portrait)"** - A square plot, centered on the page.
  - **"Square (Landscape)"** - A square plot, centered on the landscape page.
  - **"Figure (Small)"** - A small encapsulated figure size, centered on page. Also, **"Encapsulated"** or **"Encapsulate"**.
  - **"Figure (Large)"** - A larger encapsulated figure size, centered on page. Also, **"Figure"**.
  - **"Color (Portrait)"** - A "centered" plot, with color turned on. Also, **"Color"**.
  - **"Color (Landscape)"** - A "centered" landscape plot, with color turned on.
- **Demi** - Set this keyword to select the Demi font style.
- **Directory** - Set thie keyword to the name of the starting directory.
- **Encapsulate** - Set this keyword to select Encapsulated PostScript output.
- **European** - Set this keyword to indicate "european" mode (i.e., A4 page and centimeter units).
- **Filename** - Set thie keyword to the name of the PostScript file. In the source code version the default is "idl.ps", but in the shareware version this keyword cannot be set and is always "coyote.ps".
- **FontSize** - ; Set this keyword to the font size. Between 6 and 36.
- **FontType** - ; Set this keyword to indicate the type of fon you want selected. Hershey Fonts: -1, Hardward Fonts: 0, and True-Type Fonts: 1.

- **Helvetica** - Set this keyword to select the Helvetica font. This is the default selection.
- **Helvetica** - Set this keyword to select the Helvetica font.
- **Inches** - Set this keyword to indicate sizes and offsets are in inches as opposed to centimeters.
- **Italic** - Set this keyword to select the Italic font style.
- **Isolatin** - Set this keyword to select ISOlatin1 encoding.
- **Landscape** - Set this keyword to select Landscape page output.
- **Light** - Set this keyword to select the Light font style.
- **Medium** - Set this keyword to select the Medium font style.
- **Name** - Set this keyword to the "name" of the object. This can be anything you like. Objects that have different names can display their graphical user interfaces simultaneously. The object name appears on the title of the graphical user interface.
- **Narrow** - Set this keyword to select the Narrow font style.
- **Oblique** - Set this keyword to select the Oblique font style.
- **PageType** - Set this keyword to the "type" of page. Possible values are:
  - **"Letter"** - 8.5 by 11 inches. (Default, unless the **European** keyword is set.)
  - **"Legal"** - 8.5 by 14 inches.
  - **"Ledger"** - 11 by 17 inches.
  - **"A4"** - 21.0 by 29.7 centimeters. (Default, if the **European** keyword is set.)
- **Palatino** - Set this keyword to select the Palatino font.
- **Preview** - Set this keyword to select Preview mode: 0 is "none", 1 is "EPSI", or 2 is "EPSF".
- **Schoolbook** - Set this keyword to select the Schoolbook font.
- **Set_Font** - Set this keyword to the name of a font passed to PostScript with Set_Plot keyword.
- **Symbol** - Set this keyword to select the Symbol font.
- **Times** - Set this keyword to select the Times font.
- **TrueType** - Set this keyword to select True-Type fonts.
- **Update** - Set this keyword if you wish to immediately update the graphical user interface after setting a property. The default is to set the property without updating the display.
- **XOffset** - Set this keyword to the X Offset.
- **XSize** - Set this keyword to the X size of the PostScript "window".
- **YOffset** - Set this keyword to the Y Offset. Uses "System (Portrait)" defaults.
- **YSize** - Set this keyword to the Y size of the PostScript "window".
- **ZapfChancery** - Set this keyword to select the ZapfChancery font.
- **ZapfDingbats** - Set this keyword to select the ZapfDingbats font.

For example, to create a 7 x 5 inch window on a PostScript landscape page, with color output turned on, and immediately update the display, you can type this:

```
  psObject->SetProperty, /Landscape, /Color, /Update, XSize=7, YSize=5, /Inches,
XOffset=1, YOffset=1
```

[Return to Contents]

Setting PostScript Configuration Keywords

The whole purpose of the **FSC_PSConfig** program is to configure the PostScript device by setting the appropriate **Device** command keywords. This is done by obtaining a structure in which the fields of the structure are the names of the appropriate keywords, and the values of the fields are appropriate for configuring the device the way you want it configured. You pass this structure to the **Device** command by means of *keyword inheritance*. In other words, you use the **_Extra** keyword to accept the structure.

The keyword structure is obtained from the object by calling the **GetKeywords** method, like this:

```
   keywordStructure = psObject->GetKeywords()
```

To verify what this structure looks like, type this:

```
   IDL> Help, keywordStructure, /Structure
```

Depending upon how your object is configured, you should see something like this:

```
   ** Structure <15037f8>, 24 tags, length=68, refs=1:
      BITS_PER_PIXEL  INT              8
```

```
COLOR           INT              0
ENCAPSULATED    INT              0
FILENAME        STRING    'D:\RSI\IDL53\DAVID\coyote.ps'
FONT_SIZE       INT             12
INCHES          INT              1
ISOLATIN1       INT              0
PREVIEW         INT              0
TT_FONT         INT              0
XOFFSET         FLOAT           1.75000
XSIZE           FLOAT           5.00000
YOFFSET         FLOAT           3.50000
YSIZE           FLOAT           4.00000
PORTRAIT        INT              1
LANDSCAPE       INT              0
HELVETICA       INT              1
BOLD            INT              1
BOOK            INT              0
DEMI            INT              0
ITALIC          INT              1
LIGHT           INT              0
MEDIUM          INT              0
NARROW          INT              0
OBLIQUE         INT              0
```

Note that your structure may have other keywords defined, depending upon how your object is configured. This is typical.

These keywords are passed to the PostScript device via the **_Extra** keyword to the **Device** command. Be sure to select the PostScript device first. The code will typically look something like this:

```
thisDevice = !D.Name
Set_Plot, "PS"
Device, _Extra=psObject->GetKeywords()
```

Once the PostScript device is configured, you issue your graphics command or commands as normal. They will go into the PostScript "window" you have described on the PostScript page. Then, be sure to close the file. The code might look something like this:

```
Plot, data, Title='Experiment 5A'
Device, /Close_File
Set_Plot, thisDevice
```

A Note About Fonts

Although the PostScript device allows you to set information about PostScript fonts (if you set the **FontInfo** keyword when you invoked the GUI), there is no guarantee that your graphics commands will be rendered in PostScript fonts. In fact, this is a function of the **!P.Font** system variable or the **Font** keyword on a graphics command, and has nothing to do with how the PostScript device is configured *per se*. This makes it a bit tricky to put font property widgets on a graphical user interface, since the unaware user may be surprised that the output doesn't look like the configuration he or she selected.

I've attempted to get around this problem by letting you know of the user's wishes for font type. But this information doesn't come back in the device keyword structure. Rather, it is returned in a **FontType** output keyword from the **GetKeywords** method. In fact, I recommend you always call the method like this when you have asked the user to specify font information in the GUI:

```
thisDevice = !D.Name
Set_Plot, "PS"
Device, _Extra=psObject->GetKeywords(FontType=fonttype)
```

This allows you to set the font type correctly on your graphics commands. For example, you could write code like this:

```
currentFontType = !P.Font
!P.Font = fonttype
Plot, data, Title='Experiment 5A'
Device, /Close_File
Set_Plot, thisDevice
!P.Font = currentFontType
```

Or, even more simply, you could write code like this:

```
Plot, data, Title='Experiment 5A', Font=fonttype
Device, /Close_File
Set_Plot, thisDevice
```

[Return to Contents]

Cleaning Up the Program

Since the **FSC_PSConfig** program is an object, it is persistent in the IDL session. This is one of the huge advantages of objects. For example, you can create an PostScript configuration object in a start-up file and always have it available in your IDL session. You will always have access to the current configuration of the PostScript device.

But as a persistent entity, you must destroy it when you are done with it or you will have heap memory leaking in your IDL session. This is especially true if you create these objects in other program modules. This is no different for other memory-leaking entities: pointers, pixmaps, etc. The way you clean up an object is to destroy it with the **Obj_Destroy** command, like this:

```
Obj_Destroy, psObject
```

[Return to Contents]

Example Program using FSC_PSConfig

Sometimes the best way to see how to use a new program is to see an example. Here is an example program, named PS_Plotter, that uses the **FSC_PSConfig** object to create a window on the PostScript page for two plots: an image plot and a histogram plot of the image data. You will need another useful program from my library to run the program: TVImage. (**TVImage** is needed because the normal IDL **TV** command doesn't honor the **!P.Multi** system variable. Nor is the **TV** command written in the device-independent way required here.) Here is the compete code for the **PS_Plotter** program.

```
PRO PS_Plotter, image, $
   European=european, $   ; Set this keyword if you want European measurements.
   Object=object          ; Output variable to return FSC_PSConfig object.

   ; Get an image if one is not passed in.

IF N_Elements(image) EQ 0 THEN BEGIN
   image = BytArr(360, 360)
   file = Filepath(SubDirectory=['examples', 'data'], 'worldelv.dat')
   OpenR, lun, file, /Get_Lun
   ReadU, lun, image
   Free_Lun, lun
ENDIF

   ; Create the PostScript configuration object.

object = Obj_New('FSC_PSConfig', European=Keyword_Set(european))

   ; We want hardware fonts.

thisFontType = !P.Font
!P.Font = 1

   ; Get user input to PostScript configuration.

object->GUI

   ; Configure the PostScript Device.

thisDevice = !D.Name
Set_Plot, 'PS'
keywords = object->GetKeywords(FontType=fonttype)
Device, _Extra=keywords

   ; Draw the example plots.

!P.Multi = [ 0, 1, 2]
TVImage, image
Plot, Histogram(image), Title='Example Histogram Plot', XTitle='Pixel Value', $
   YTitle='Number of Pixels', XStyle=1, Max_Value=5000
```

```
   ; Clean up.

!P.Multi = 0
Device, /Close_File
Set_Plot, thisDevice
!P.Font = thisfontType

   ; Return the PS_Configuration object or destroy it.

IF Arg_Present(object) EQ 0 THEN Obj_Destroy, object
END
```

If you will be printing on letter-size paper, you want to run the program like this. You will see the graphical user interface of the object in the middle of your display. Fill out the form the way you would like the output file to be configured and click the *Accept* button.

```
IDL> PS_Plotter
```

If you are in a country that uses A4 paper, you might want to call the program like this:

```
IDL> PS_Plotter, /European
```

A PostScript file will be created in the specified directory. You can print it or use a PostScript file viewer (e.g., Ghostview) to view the file.

Called as above, the **FSC_PSConfig** object is created and destroyed when the program exits. However, you may wish to play around with some of the object's properties. You can return the object from the **PS_Plotter** program by using the optional **Object** keyword, like this:

```
IDL> PS_Plotter, Object=psObject
```

If you want to see how the PostScript file was just configured, you can examine the PostScript Device keywords that were used to create the file:

```
IDL> Help, psObject->GetKeywords(), /Structure
```

Be sure to destroy the object when you are finished with it.

```
IDL> Obj_Destroy, psObject
```

[Return to Contents]

A Wrapper Function for FSC_PSConfig

A number of people are still uncomfortable using objects. So for those people, I have written a function named PSConfig, which acts very much like (in fact, can probably be used as a replacement for) the old PostScript configuration program, **PS_Form**.

The **PSConfig** program creates the **FSC_PSConfig** object, calls the graphical user interface in blocking mode, and then returns the device keywords necessary to configure the PostScript device. The program can be used with commands similar to these:

```
keywords = PSConfig(Cancel=cancelled)
IF NOT cancelled THEN BEGIN
   thisDevice = !D.Name
   Set_Plot, "PS"
   Device, _Extra=keywords
   Plot, Findgen(11)
   Device, /Close_File
   Set_Plot, thisDevice
ENDIF
```

You can use the same keywords with **PSConfig** that you use for the **FSC_PSConfig** object.

[Return to Contents]

Customizing Your Source Code

I've tried to write the **FSC_PSConfig** program in such a way that it can be easily customized and extended by you at your site. In particular, it will be easy to add new default set-ups and other page types to the program. The code itself is extensively documented to indicate how this can be done. But, to give you an example, here is how you would add another default set-up, named *Company Viewgraph* to the code.

Step 1: Add the Name to the Set-Up List

Open the *fsc_psconfig__define.pro* file and find the *DefaultList* method. (For example, search for "::DefaultList".) Add your new default name to the list shown there. For example, you would find this code:

```
defaultlist = [ 'System (Portrait)', $
                'System (Landscape)', $
                'Centered (Portrait)', $
                'Centered (Landscape)', $
                'Square (Portrait)', $
                'Square (Landscape)', $
                'Figure (Small)', $
                'Figure (Large)', $
                'Color (Portrait)', $
                'Color (Landscape)' ]
```

and change it to this:

```
defaultlist = [ 'System (Portrait)', $
                'System (Landscape)', $
                'Centered (Portrait)', $
                'Centered (Landscape)', $
                'Square (Portrait)', $
                'Square (Landscape)', $
                'Figure (Small)', $
                'Figure (Large)', $
                'Color (Portrait)', $
                'Color (Landscape)' , $
                'Company Viewgraph']
```

Step 2: Create the New Set-Up

Next, find the *SetDefault* method (it will be just below the *DefaultList* method you just found) and copy one of the defualt setups you find in the CASE statement in that code. For example, here is the *System (Portrait)* method found there:

```
CASE thisDefault OF

   'System (Portrait)': BEGIN
         self.bitsSet = '8'
         self.colorSet = 0
         self.directorySet = directoryName
         self.encapsulationSet = 0
         self.filenameSet = defaultFilename + ".ps"
         self.fonttypeSet = !P.Font
         self.fontsizeSet = 12
         self.fontStyleSet = Replicate(0, 8)
         self.fontnameSet = "Helvetica"
         self.inchesSet = units
         self.landscapeSet = 0
         self.isolatinSet = 0
         self.pagetypeSet = pagetype
         self.previewSet = 0
         self.truetypeSet = 0
         IF self.european THEN self.xoffsetSet = 1.61 ELSE self.xoffsetSet = 0.75
         IF self.european THEN self.yoffsetSet = 14.65 ELSE self.yoffsetSet = 5.0
         IF self.european THEN self.xsizeSet = 17.80 ELSE self.xsizeSet = 7.0
         IF self.european THEN self.ysizeSet = 12.70 ELSE self.ysizeSet = 5.0
         self.defaultsSet = 'System (Portrait)'
      ENDCASE
```

Modify each of the fields you want to change there, and add your changed values to the CASE statement. For example, your *Company Viewgraph* set-up might be defined like this. Note that you will have to provide both inches and centimeter units for the sizes and offsets.

```
   'Company Viewgraph': BEGIN
         self.bitsSet = '8'
         self.colorSet = 1
         self.directorySet = "C:\Company\Viewgraphs"
```

```
        self.encapsulationSet = 0
        self.filenameSet = "viewgraph_1.view"
        self.fonttypeSet = 1
        self.fontsizeSet = 18
        self.fontStyleSet = Replicate(0, 8)
        self.fontnameSet = "Helvetica"
        self.inchesSet = units
        self.landscapeSet = 0
        self.isolatinSet = 0
        self.pagetypeSet = pagetype
        self.previewSet = 0
        self.truetypeSet = 0
        IF self.european THEN self.xoffsetSet = 0.75 * 2.54 ELSE self.xoffsetSet =
0.75
        IF self.european THEN self.yoffsetSet = 0.75 * 2.54ELSE self.yoffsetSet =
0.75
        IF self.european THEN self.xsizeSet = 7.0 * 2.54 ELSE self.xsizeSet = 7.0
        IF self.european THEN self.ysizeSet = 9.5 *2.54 ELSE self.ysizeSet = 9.5
        self.defaultsSet = 'Company Viewgraph'
     ENDCASE
```

[Return to Contents]

Other FSC Programs Needed to Run this Program

This **FSC_PSConfig** program uses a number of other programs from the Coyote library. Here are the programs you will need.

- fsc_psconfig__define.pro - The main PostScript configuration object program. (What I call **FSC_PSConfig**.)
- fsc_field.pro - A CW_Field replacement compound widget, written as an object.
- fsc_fileselect.pro - A file selection compound widget, written as an object.
- fsc_droplist.pro - A droplist compound widget, written as an object.
- fsc_plotwindw.pro - A utility object program required by **FSC_PSConfig**.

In addition, you may want the following wrapper and example programs, respectively:

- psconfig.pro - A wrapper function for the **FSC_PSConfig** object.
- ps_plotter.pro - An example program that uses the **FSC_PSConfig** object.
- tvimage.pro - A device-independent image display program, which honors the !P.Multi system variable.

If you would like to download all these programs at once, you can download this zip file (56KB).

[Return to Contents]

Disclaimer and Licensing Information

Warranties

Fanning Software Consulting makes no warranties, either express or implied, as to the condition of the software described here or its fitness for any particular purpose. All software code is provided as is. IDL is a registered trademark of Research Systems, Inc. for the computer software used to created these programs. A licensed version of IDL is required to run these programs.

Licensing the Software

All programs written by Fanning Software Consulting and described in this article are issued under a license certified and approved by the Open Source Initiative. The following license is attached to the header of each program. Please do not removed the license information if you distribute this source code in any form.

```
;##########################################################################
;
; LICENSE
;
; This software is OSI Certified Open Source Software.
; OSI Certified is a certification mark of the Open Source Initiative.
;
```

```
; Copyright © 2000 Fanning Software Consulting
;
; This software is provided "as-is", without any express or
; implied warranty. In no event will the authors be held liable
; for any damages arising from the use of this software.
;
; Permission is granted to anyone to use this software for any
; purpose, including commercial applications, and to alter it and
; redistribute it freely, subject to the following restrictions:
;
; 1. The origin of this software must not be misrepresented; you must
;    not claim you wrote the original software. If you use this software
;    in a product, an acknowledgment in the product documentation
;    would be appreciated, but is not required.
;
; 2. Altered source versions must be plainly marked as such, and must
;    not be misrepresented as being the original software.
;
; 3. This notice may not be removed or altered from any source distribution.
;
; For more information on Open Source Software, visit the Open Source
; web site: http://www.opensource.org.
;
;##########################################################################
```

Reporting Program Bugs

Every attempt is made to keep these programs accurate and bug free. If you find a bug, I will be happy to attempt to fix it. Please report any bugs or feature requests to:

David Fanning
Fanning Software Consulting
1645 Sheely Drive
Fort Collins, CO 80526 USA
Phone: 970-221-0438
Fax: 970-221-4762
E-Mail: david@dfanning.com

[Return to Programs Page]

Last Updated 30 October 2002