Personal Designer

User Programming Language (UPL)

Revision 6.0

User Reference Guide

Appendix H

Writing Personal Designer Commands

Writing Personal Designer Commands

It is possible to write a UPL program so that it behaves exactly like a built-in Personal Designer command. This is done by accessing the portions of Personal Designer which implement the user interface. UPL supplies routines to access the Modifier Processor, Getdata Processor, and the On-Line Help system. Note that it is not necessary to access the Verb/Noun processor (VNP). Instead, the UPL program name can be added to the Verb/Noun tables and invoked like any Personal Designer command. Otherwise the Personal Designer command RUN will process the verb/noun portion.

General

The normal Personal Designer command syntax is:

```
INS LIN ANG 45 COLOR 9 : ent d,dig dd
| -VNP- || -------- MP -------- || - Getdata - |
```

The VNP section will be handled by Personal Designer before the UPL program is invoked. The Modifier Processor section is handled by the UPL program with the DefineModifier, AskModifiers, and GetModifier procedures. The Getdata section is handled with the GetEnt, GetDig, and GetEnd procedures.

On-line help in the Modifier Processor and Getdata sections is handled by using the SetHelp procedure immediately before using Askmodifiers, GetEnt, GetDig, or GetEnd.

One of the important aspects of making a UPL program behave like a Personal Designer command is how the command syntax is handled after using AskModifiers, GetEnt, GetDig, or GetEnd, The way in which the command syntax should be handled for each section is described in the following paragraphs.

Verb I Noun Processor

There is actually nothing a UPL program needs to do to handle the VNP portion of the command. The program will either be invoked by the RUN command or by adding the program name to the verb/noun table. Since a VNP command invokes the UPL program, the VNP portion of the command is already done.

However, your program should properly handle the character that terminated the VNP portion of the command. The three possible characters are: a space (32), a ":" (58), or a <CR> (13). Use the **LastChar** UPL system variable to determine this character. Normally this character is handled in the following way:

space	enter the modifier processor **(AskModifiers)**
":"	enter get data (GetDig, GetEnt, or GetEnd)
<CR>	if possible, do some default action, then exit the program.

You may install your UPL program into Personal Designer so it can be invoked with a set of verb/noun command words of your choice. Run the UPL program UPLCMD while in Personal Designer. It will prompt you for all the needed information. The UPL program UPLCMD.UPL is supplied on the UPL distribution diskette. It should be compiled and the resulting UCD file stored in the \PD6 directory. In addition, the file UPLCMD.DEF should be processed with the BLDF utility and stored with with UPLCMD.UCD. This will build the on-line help file for UPLCMD.

Modifier Processor

There are three UPL routines that handle the Modifier Processor section. The first is the **DefineModifier** procedure. It is used to define each word that the user can enter in the Modifier Processor portion of the command. This needs to be called once for each modifier at the beginning of the UPL program and after the UPL Send command is used. After using the Send command and before using the **AskModifiers** procedure, you will need to redefine your Modifier Processor words.

The next procedure is the **AskModifiers** procedure. It is used to actually enter the Modifier Processor and get the values of the modifiers from the user. The user terminates the Modifier Processor section by entering a ":", a <CR> (13), or ^C (3). This character will be stored in the **LastChar** system variable. If you want your program to use on-line help, the SetHelp procedure should be called before calling AskModifiers. This call to SetHelp will set up on-line help. However, it is not necessary for the AskModifiers procedure to work correctly.

The **GetModifier** procedure can then be used to return the value of each modifier word. Since **DefineModifiers** supplies default values, a valid value will be returned by GetModifier whether the user has selected the modifier or not.

Getdata

The Getdata section of the command allows the user to pick entities and/or enter locations. This can be done in a UPL prograrn by calling the GetEnt, GetEnd, and GetDig intrinsic procedures. If you want on-line help available to the user, the **SetHelp** procedure should be called before these procedures are called .

The GetEnt, GetEnd, and GetDig procedures will return when either the user:
inputs the maximum number of entities or digitized points
enters a to return to the modifier processor
enters a to end the eurrent section of get data input
enters a <CR> to execute and then exits the command or
enters a ^C to abort the command.

LastChar can be used to determine which character caused the Getdata input to end. This should be used in conjunction with the number of entities, ends, or digitizes input to determine the appropriate action by the UPL program.

On-line Help

The powerful on-line context sensitive help system used by Personal Designer is available to the UPL programmer to provide help text to the user.

The on-line help file for a UPL program is made the same way it is made for Personal Designer. First, a help definition text file (.DEF) is made. Then, using the BLDF utility, the file is "built" into the forrnat (.HLP) that is used by the on-line help system.

See the INSBCIR.DEF file on the distribution disks for an exampie help file for the INSBCIR command.

Writing Personal Designer Commands

The help system can be accessed by the user in four ways:

1) the Personal Designer command HELP
2) via entering a in the VNP portion of a command
3) via entering a in the modifier processor portion or
4) via entering a in the Getdata portion.

Once in the help system, the user can browse through all of the help screens that the help documenter has given the user access to.

The help system is based on index numbers. Help index numbers point the help system to blocks of Help information. These blocks may have numbers which point the help system to other blocks of help information. An hierarchy of help information can be built for a given command by setting up the help indexes for the command and all its modifiers.

To become familiar with the help index numbers, look at those for existing Personal Designer commands. You can determine any command's help index numbers for VNP, modifier processor, and Getdata by entering the Personal Designer command SEL MESSAGE HELP. Then, whenever a command is invoked, or the system enters the modifier processor or Getdata, the help index numbers associated with the command will be displayed.

Other support files related to the help system are the keyboard macro file and the messages file (PDMAC.DEF and PDMSG.DEF). The keyboard macro file determines what values are passed to Personal Designer when a key is pressed. When the help system is invoked, keyboard macro set four is activated. Look in PDMAC.DEF for information on what codes will be returned while in the help system. The messages file determines the text of messages output by Personal Designer concerning the syntax of commands. In PDMSG.DEF, message numbers 908 through 912 give the prompts that appear at the bottom of the help screens. These files are supplied on the Personal Designer distribution diskettes in the directory \PDMISC, or use the file PDDUMP.DEF to decompile and output them.

Writing Personal Designer Commands

The BLDF commands that are unique to the help system are:

File Modifiers <modifiers file name>

This command gives the file name of a modifier processor file to get words from when validating the modifier words given in the Modifiers help command.

Help <help index> [<answer help indexes>]
<help text>

@

This command gives help text to be associated with <help index>. The <help index> can be any integer in the range 1 through 32,767. The <help index> = 1 has a special purpose in that it is the help index that is given for the Personal Designer HELP command.
The optional <answer help indexes> are <help index>s from other Help commands. If <answer help indexes> are given, when the user responds with answer one, the first <help index> in <answer help indexes> is used as the <help index> for the next screen to appear. If the user responds answer two, then the second <help index> is used, etc. (Any key can be used for user responses, currently PDMAC.DEF is arranged so that F I sends the answer one response code, F2 sends answer two response code, etc.) The <help index>s in <answer help indexes> do NOT have to be defined before they can be used.

VNP <vnp index> [<help index>]
<help text>

@

This command assigns help text to a particular VNP index. The VNP index is the number assigned to the command or UPL program in the Verb/Noun table. If the UPL program is not added to the VNP a one may be used. The optional <help index> gives the user the chance to get more help if they choose the "More" response.

ModIndex <modifier index> [<help index>]

This command sets the modifier index number to use in subsequent Modifiers commands. The modifier index number is the number used in the modifier table to reference a set of command modifiers. If a UPL program's modifiers are added to the modifier table, a unique modifier index should be used. If not, a one may be used.

The optional <help index> gives the user the chance to get more help if they choose the "More" response. This is a good place to put the help index which points to a special <help text>. This special text explains all of the modifiers for the given modifier index. There are many ways you can organize the help index numbers. Hint: To help you keep track of which help screens go with which groups of modifiers, you could use the same number of the <modifier index> for the <help index>. Or you could use the modifier index*10 then use the units for other help associated with this set of modifiers.

Note that no help text is associated with this command. It only sets up the current modifier index and eurrent modifier more help index for future Modifiers commands.

Modifiers Default | Exact | Any | All | None [<modifier words>]
<help text>
@

This command allows you to link the <help text> with the combinations of [<modifier words>] for a UPL program or a Personal Designer command. When the set of modifier words entered by the user matches the specified <modifier Words>, the <help text> is displayed.

The matching is controlled by the keyword preceding the modifier list. The first "Modifiers" command that matches the selection criteria will be activated. Therefore, it is important how you order the Modifiers commands under the VNP command.

If you have the first Modifiers command with the Default keyword, there is no chance that any of the Modifiers commands below it will ever be selected for the current VNP command. Also note that if only one modifier word is -iven the All and Any keywords act the same way.

The match keywords have the following meanings:

Default -	You can group "Modifiers" statements into sections that use different modifier combinations. If a command is activated that does not trigger any previous modifier section, the default section is activated. InLIN: goes directly from VNP to Getdata), this is the Modifiers group that will be used. No modifier words are given with this keyword.
Exact -	the modifiers entered by the user must exactly match the specified <modifier words>
Any -	activates the <help text> if any of the modifiers entered by the user match the specified <modifier words>.
All -	activates the <help text> if all of the modifiers entered by the user match the specified <modifier words>. This is similar to Exact but more modifiers may be select in Personal Designer than what is given.
None -	activates the <help text> if none of the modifiers entered by the user match the specified <modifier words>.

GetData <getdata index> [<help index>]
<help text>
@

This command links <help text> to the given <getdata index>. This command is activated when the user asks for help by typing ! in getdata. The Getdata help given will be associated with the previous Modifiers command.

> The Getdata indexes are set up in the Personal Designer software or with the UPL intrinsic **SetHelp.** The help documenter can find out what these indexes are for each command by using the Personal Designer command SELect MESSage HELP.

The optional <help index> gives the user the chance to get more help if they choose the "More" response.

In general the order of the commands should be as follows. First, use the VNP command for the command or program. Next come the Modifiers commands. Under each modifier, any appropriate Getdata commands should be given for combinations of modifiers which have special effects on the Getdata help. A more general Getdata command may be used if no special ones are needed. It should be given after the last MOdifiers command. The Help <index> commands can be used anywhere in the file in any order. They should be inserted wherever they apply to a modifier or whatever. This makes the help file easier to read.

If each UPL program's help file is made by a different user and later merged into one help file, you must coordinate index number assignments. That is, if only one help file is used, the help indexes of all the commands or programs used by it must be unique to those at commands or programs. If you have a separate help file for your UPL program, the indexes are easier to create. See the INSBCIR.DEF file and the INSBCIR.UPL programs for an example of a separate help file for a UPL program.

NOTE: The form feed character ASCII 12 or ^L can be used anywhere in a <help text>. It will cause the rest of the help screen to be blank. The next part of the <help text> can then be obtained by pressing a space bar or return key.

Writing Personal Designer Commands

Outline of UPL Program for a Personal Designer Command

The following is pseudo-code for setting up a UPL program to emulate a Personal Designer command.

```
First, set up modifiers using DefineModifier

Loop to get modifier values
    We are all done if LastChar = ^C or a <CR>

    If we just started the UPL command and LastChar is
    not a ':' Or
    If this isn't the first time through this loop then
        Set up on-line help with SetHelp
        Get the modifier values from the user
        with AskModifiers

        We are all done if LastChar = ^C or a <CR>

        Get the modifier values the user input
        with GetModifier
    Else
        Just keep going
    End If

    Loop (to get Getdata type information)
        Set up on-line help for Getdata with SetHelp
        Loop
            Now we enter Getdata by using GetEnt, GetEnd
            and/or GetDig

            We are all done if LastChar = ^C

            If we got the entities and/or digitizes we needed Then
                Exit this loop
            End If
        End Loop

        Do whatever to carry out the function of the command

        If LastChar = <CR> Then
            Exit command (we are all done)
        Else If LastChar = ':' Then
            Exit the loop we are in to go back to
            the modifier processor and get new modifier values
        Else
            (LastChar must have been ";" )
            Stay in the getdata section and get some
            more Getdata type information.
            Use the current values of the modifiers
        End If
    End of loop for get data section
End of loop for modifiers section
```

Also you can look at the INSBCIR.UPL program for a simple example.