

XBASIC – Extended BASIC Compiler

Email: Info@PrecisOnline.com

Introduction

Despite the best efforts of the SMA, et al., there are a number of constructs on each Multivalue platform that are incompatible with other platforms. Often times, there are functional alternatives, though the syntax for these alternatives is generally quite different from system to system. For example, to open a sequential file, write an item to it, and close the file, the following lines could be used...

... on an AP/SCO system

```
STREAM = %FOPEN(file,'w')
STATUS = %FPUTS(record,(char *) STREAM)
STATUS = %FCLOSE((char *) STREAM)
```

...on Universe/Unidata

```
OPENSEQ file TO F.SEQ THEN
  WRITESEQ record APPEND TO F.SEQ ELSE NULL
  CLOSESEQ F.SEQ
END
```

The end result of each of these blocks is the same; a file will be opened, written to, and closed at the Unix level. However, the Universe/Unidata syntax is incompatible with AP, and the AP syntax is incompatible with Universe/Unidata. This is where XBASIC can help.

Using XBASIC as the compiler, the following code can be included in a source item, and the various blocks of code will be included in the source item depending on the destination system.

```
#IF AP
  STREAM = %FOPEN(file,'w')
  STATUS = %FPUTS(record,(char *) STREAM)
  STATUS = %FCLOSE((char *) STREAM)
#END

#IF UNIVERSE
  OPENSEQ file TO F.SEQ THEN
    WRITESEQ record APPEND TO F.SEQ ELSE NULL
    CLOSESEQ F.SEQ
  END
#END
```

Note the #IF and #END directives. At the beginning of a block of code for a specific platform, you can include the #IF directive as follows:

```
#IF platform
```

In this syntax, *platform* is a user-defined name of the platform that supports the following code. Any number of platform codes may be provided here, as long as:

- ❑ All platforms are delimited by a comma
- ❑ There are no spaces between the first character of the first platform name and the last character of the last platform name.

Therefore, if a block of code is specific to AP only, the following #IF directive can be used:

```
#IF AP
```

If a block of code is specific to Universe and Unidata, the following #IF directive can be used:

```
#IF UNIVERSE, UNIDATA
```

At the end of the platform-specific code, the #END directive is added. Comments may follow the #END directive, as follows:

```
#END Universe specific code
```

Note: XBASIC converts unused lines to comments, as well as the #IF and #END directives, so the line numbers of the source will agree with the line numbers in the debugger.

Using XBASIC

XBASIC can be used as a functional replacement for the BASIC verb at TCL. When used, XBASIC will prompt for the operating environment, as follows: (Bold-faced items are user entries.)

```
> XBASIC XXXPROGS TEST
Operating Environment > UNIDATA
Precompile for UNIDATA? (Y/<cr>=N) : Y
```

Like BASIC, XBASIC can be run from a select list, supports multiple item names on the command line, and supports all compiler options. (Note: Compiler options are recognized as any words on the command line that are prefaced by a dash (for Universe/Unidata) or open parenthesis (for AP/Pick).)

Failure Recovery

Immediately before compilation, the precompiled item (with all of the current system-specific code included, and other system-specific code removed) is written ON TOP OF the original item. If the system should go down during the compilation, the original code can be restored from an item named *item.Orig* in the current file. Therefore, using the above example, if the system were to fail during compilation, you can simply copy XXXPROGS TEST.Orig over to XXXPROGS TEST to restore the original item.

Installation

To install XBASIC, compile (but do not catalog) the source code provided. In the MD/VOC of the account where XBASIC is to be used, create a proc as follows:

```
001 PQ
002 HRUN filename XBASIC
003 P
```

(Fill in *filename* with the name of the file where XBASIC was saved and compiled.)

Once the proc has been created, XBASIC should be ready to use!