Package 20001 - Email Tools
QPrint (*web/qprint)
Details

Calling Sequence:(must be made to a specific Entry Point)

Entry Points Available for converting Strings:

```
CALL "*WEB/QPRINT;ENCODE_STR",instring$,outstring$,mimetype$
CALL "*WEB/QPRINT; DECODE_STR",instring$,outstring$,mimetype$,EndOfLine$
```

Entry Points Available for converting entire files:

```
CALL "*WEB/QPRINT;ENCODE_FILE", infile$,outfile$, mimetype$, EndOfLine$
CALL "*WEB/QPRINT;DECODE_FILE",infile$,outfile$,mimetype$, EndOfLine$
```

Purpose of Program:
This utility will encode and decode quoted-printable data. You may encode both text and binary data in quoted-printable format, however quoted printable is not an efficient encoding format for binary data.Each character outside of a specific range of ASCII characters are translated from 1 character to 3 characters.

The maximum length of a single line when encoded is 76 characters (not including the CRLF line terminator).

Encoded Characters are represented as =XX within the output. Where XX is the ASCII representation of the hex code of the character.

Usage Details:
InString$ and OutString$ may not exceed 32000 characters. If an encoded string attempts to exceed this length, then an error will be generated.

MimeType$ is the mime content-type for the type of data being passed or received from this utility. Typically ASCII Text data is represented as beginning with "text/". This utility will use the mime type to determine the end of line conversion characteristics.

The EndOfLine$ may be given or may be null. If it is null, then the applicable end of line terminator will be chosen based on which OS this program is running on. $0A$ for UNIX and $0D0A$ for MS Windows.

The InFile$ and OutFile$ may be given as either specific file names, or as channel numbers (Infile$="23"). If the Outfile$ does not exist then it will be created. The InFile$ is always read from its beginning, while the OutFile$ is always appended to. If you wish to pass channel numbers, then the file must already be opened on that channel. This utility will only close channels that it specifically opened. By using channel numbers you may encode or decode directly to a *memory* device.

If the string has a CRLF and its MIMETYPE$ begins with "text/" then the CRLF's become CRLF's in the outgoing stream of data. However, all other content types while have the CRLF turned into =0D=0A

Notes on dealing with *memory* files during Encode/Decode:

During Encode:
If the incoming data has a binary mime type then no end of line terminators are assumed. If the data is a text mime type, then end of line terminators are assumed and will be added to each record read if the record does not end in an end of line terminator. If the outgoing channel is a *memory* file, then each line of encoded data is written as an individual indexed record, without any EndOfLine$

During Decode:
If the incoming data has a binary mime type then no end of line terminators are assumed, if it is a text mime type then end of line terminators are assumed and will be added to the end of each record read if the record does not end in an end of line terminator. If the outgoing channel is a memory file, and the data is of type text, then each record will be written without an end of line terminator. If you wish to save such data then an end of line terminator must be added to each record read, to obtain the original file. If binary, then each record will be written with no end of line terminator, to save such binary data, read each record and append without changes or additions to a file.

Text encoding either to a String or to a File:
Due to the nature of Quoted Printable encoding, the encoded data may end up with an extra end-of-line terminator. Data that finishes with an end of line terminator will encode and decode exactly the same. However data that does not end with an end of line terminator, will encode and decode, and the decode version will be a byte or two longer.

memory files as either the incoming or outgoing side of encode or decode operation:
The end-of-line terminator may be either $0d0a$ $0a$ or no terminator.
When output to a *memory* file, the records will be written without a terminator. It is then the responsibility of the task that called this program to add the appropriate terminator for the job they wish done.

If this program encounters an error during processing then it will EXIT ERR.

Errors Returned:
Typical File open errors, for either infile$ or outfile$.
0-Busy/perms, 12-cant find, etc

Handling Errors:

46 - string when encoded, will exceed maximum string length.
48 - Invalid Data, Unknown Character encountered during decode, indicates corrupted data, or data not in quoted-printable format.