Specifications for PubLink Inc's new text editor

DATE
: Tues 12 June 2007

SPECS VERSION
: 4

AUTHOR
: Peter Butterfield. PubLink Inc.

PRODUCT NAME
: PLedit

(see README.v3.html file)

documentation

program base

Written from scratch in java. By agreement with PubLink, the developer will own rights to the core or base set of classes of the editor which provide general editing functionality not specific to PubLink's editor. He may use this base to develop and sell other editing products on his own, for applications that need features similar to those of PubLink's composition text file (as distinct from a programmer's program-editing application like jEdit). PubLink will own rights to the "outer layer" and to the full delivered package, including all functionality that is specific to our .tx2 files, DeskTopPro, XML Publisher, our new WebLinkPL interface, and PubLink's processes and procedures in general.

operating platform

Operating platform

Editor should operate on two platforms initially, with a third needed in the near future:

1. x86 platform, 128Mb memory min, running Windows XP SP2 and Java J2SE 5.0. The x86 CPU should be of a level and speed capable of handling well at least two browsers and/or java-based user-interface applications simultaneously.
2. Sun sparc workstation running Solaris 9 or higher, min Ultra10 with 256Kb memory.
3. Near future: x86 platform running Solaris 9 or higher.

Development is on a 1GB ram system purchased used for about $150. Specifically an emachine C1825 with 1GB ram and an AMD Athlon XP Processor 2800+ (2.083GHz). An additional 1GB of ram is about $90.

In the interest in getting PubLink something usable as soon as possible, I'm not optimising the code for a smaller amount of RAM and have no means of testing with less 1GB. However, I am designing the code to be as light as possible — RAM friendly.

Implementation with server:

PLedit must operate either locally on the Solaris server or remotely on a networked client platform, as follows.

- On the solaris workstation (server), whether Sun sparc or x86 (near future), PLedit runs locally, accessing files directly and performing external functions with direct "exec" system calls.
- On an x86 client workstation, PLedit runs remotely for production. Ideally it accesses files directly from the samba-mounted server's /Penta filesystem. If /Penta is not mounted, it accesses files via calls to an agent program on server. It performs all other external functions (flash H&J, file lock/unlock, server "exec" scripts) through

calls to the agent program. Communication method with the agent TBD.
- On an x86 client workstation, PLedit may also run in an alternate purely local mode. This mode does not offer production functionality and will be used for development, testing and demos. In this mode, documents are opened from anywhere in the Windows file system.

function of the app within publink's desktoppro publishing package

Will replace our current editor, EditMaster, which is an X-windows implementation of a hardware-specific editing station first developed in the 1970s. There will be a transation period with both editors on user's systems, when either one may be used.

PLedit (and EM before it) is the window into the text content, typesetting commands, and control or "pass-thru" coding underneath, of documents that are passed through the DeskTopPro typesetting/pagination system. It has two regular uses:

1. Many documents, formatted correctly in the originating word processor, XML editor, CMS or database extraction system, will be processed to PDF pages without need of edit. The others need examination and correction in the editor, before re-processing. Interface: Files are opened in the editor in three ways:
    - By browsing and selecting from the editor;
    - by clicking on the file's icon in our project-control window; or
    - in context by clicking on a line in the wysiwyg display of the associated made-up page.
2. PLedit is also used for initial entry of new documents for typesetting. All users do this occasionally for documents not sourced externally, or in-house publications, and some do it regularly as part of normal production flow, employing keyboard-entry people for the task. In this direct-entry case, the line numbers are meaningless and the file has none of the pass-thru information generated by H&J (hyphenation and justification).

overview: functions specific to an editor of typeset documents

1. Ability to open a textfile at a particular line number, by instruction from another process.
2. Ability to call our H&J engine on the textfile while it's open for editing ("flash" the file), and re-display the whole file as re-flowed by H&J.
3. Display lines of text and commands as decided by our H&J processor, showing line numbers, line break points, end-of-line "soft" hyphens, indication of left/ right/ center/ flush position within the text column, and numerical size of the line's word spaces, indents and depth.
4. Toggle display/suppression of underlying XML tags, tag-context paths, typeset coding mapped to the tags, and character entities.
5. Toggle display/suppression of command-error messages, editor's notes, author's notes.
6. Highlight the above string-types when displayed. Also highlight "[ab...]" typesetting commands.
7. Replace character-entity sequences with the glyph to be typeset, for display only (when defined).

8. Later, may add: Maintain two levels of edit trace (deletions and insertions), with ability to display/suppress the deletions, and highlight the inserts.

overview: functions common to an editor of any document type

1. Select text areas for cut/copy/paste using clipboard, and for saving in registers.
2. Fast interactive and global search/replace of visible text and commands.
3. Multi-level undo/redo.
4. Two or more files open at once.

textfile structure

Filename identified by '.tx2' suffix, and consists of up to 24 characters from this set: a-z, A-Z, 0-9, period, underscore, hyphen.

File is a stream of 7-bit bytes in the displayable-ascii range (0x20-0x7e and 0x0a), organized as a well-formed xml file. The stream consists of:

- xml elements providing hidden meta-data:
  - header-info element at start of file,
  - line-start elements containing line number and other line-start status info, at start of each line, and
  - "pass-thru" empty elements conveying certain H&J decisions.
- single characters (text and other glyphs) to be typeset,
- commands to control the text,
- xml empty elements and entities displayed as single characters and used either as glyphs to be typeset or as "single-key" commands, and
- xml elements fencing strings that may be toggled invisible/closed-up or displayed/ highlighted. The strings include:
  - program-inserted text & commands to be typeset,
  - program-inserted "pass-thru" informational strings, and
  - inserted and deleted text & commands (edit trace).

See file "tx2file.h" and "tx2filedoc.asc" for details on all xml tags and attributes that make up a tx2 file.

Each regular line as built by H&J begins with line-status element (or "line-start group"), which contains the line number to be displayed in the left gutter and other info, and ends with NL(0x0a). Line numbers may duplicate, as is the case with sub-lines within a table row. The editor or other applications may insert extra NL bytes for display convenience, creating sub-lines that begin without a line-start group. All NL bytes simply aid in display and are dropped upon the next processing by H&J.

basic editing operations

Opening file, general:

Documents to be edited all live on the Solaris server, which may be the

local machine or remote. On the server, all document paths obey this template:

/Penta/<treename>/desks/<projectname>.prj/<docname>.tx2

A document existing only in its previous-generation .txt form should be converted to .tx2 prior to opening, using PubLink-supplied converter. Document should be locked against multi-use.

Opening file using PLedit browser:

Program starts with a PLedit file-browser menu, which may be docked to an edge or corner. Browser's function is to guide user in selecting file to edit, and to keep that filename visible during the session AND after file is closed. This allows other menu pull-down functions to operate on the named file or process it, after its edits are complete.

For selecting a file to edit, browser restricts visible directories to the trees and projects in which all PubLink documents are stored (see above for template). The user should be able to browse among trees and the projects beneath them, while PLedit applies the template to complete the path prior to opening the file. User sees path of the form "treename/projectname/filename".

Prior to opening the selected file, PLedit should:

- Lock the file using our FACS (file access control system) facility. This is a library call for C apps; Java call method TBD.
- Convert .txt to .tx2 file if only the former exists, using converter program txcontx2. A temporary "converted from .txt" message in the status bar should appear.

After closing file, PLedit should unlock the file using FACS.

Opening file externally:

Ability to open a textfile at a particular line number, by instruction from another process. In this case, file should open in a new edit pane, and the PLedit browser should fill in with its name and path info. (Usage: User in DesignMaster clicks on a paginated line to open up the editor at that line.)

Opening file locally:

Need alternate file-open method, using normal local-file-system browser (File menu -> Open). This will open files on the local workstation, which may or may not be the Solaris server. Not meant for production documents, rather for development, testing and demo.

Multiple files:

Should be able to open two or more files at same time. Four is a reasonable maximum number, if a maximum is required. When editing multiples, the PLedit browser applies to the file "on top".

Saving file:

PLedit should periodically save file to disk, not depending on operator to do ctrl-S or other save procedure.

Sizing of display:

Ability to increase/decrease display size of characters in edit pane, overall (not per word or phrase), as aid in readability. Same for text in menu bar and its menus. The display size chosen should be remembered between editing sessions.

Display and handling of lines:

Each "line" of a file processed for typesetting contains the text, commands and underlying source code that H&J has determined make up one typeset line. Line may wrap to multiple display lines. Line's number contained within Line-Start Group should be displayed in left gutter. Edits made to lines should never affect succeeding numbered lines. Specifically, edits should not cause wrap into the next numbered line, nor pull data from the next line, nor cause succeeding line numbers to change. Keystrokes which normally cause a new line may cause insertion of NL bytes, but the new line should maintain the same line number. For instance, <Enter> should insert one of two forms of our End-Paragraph command and an NL byte: "[ep\n" or "<ep/>\n". Exception: Appending new text at end of file may freely generate new numbered lines at every wrap point.

Search/replace:

Ability to search through the open file for any string, and optionally replace. Should operate as jEdit's search/replace, although without multiple buffer/file search, bean-shell intelligence+, or HyperSearch+. Search options: Ignore case or not, backward or forward, use regular expressions or not. The search should ignore non-displaying "pass-thru" tags. Example: A search for ".[ep" (regexp off) should hit ".<cmd>[ep" (the <cmd> is the begin-command passthru tag normally found in front of any command, such as [ep). Similarly, a whole pass-thru string which is suppressed from view should be ignored in a search. This includes Line-Start groups, always ignored in a search. However, any pass-thru strings toggled for display by user, should become searchable along with the text.

Replacement should operate as expected; however if the found string spanned any hidden codes, the replacement string should be inserted in front of all such hidden codes.

Selected-text actions:

Known as BLOCK actions in EditMaster. Easy select of blocks (= user-selected areas of text). Actions once selected: Move, copy, delete and storage in a range of registers (or clipboard slots). Registers to be easily named by letter or one or two digits. Ability to copy a block from an external window into PLedit, and from PLedit to an external window. A deleted block should leave behind the Line-Start Groups that it previously spanned, but nothing else. Specifically, suppressed-from-display pass-thru strings that were within the deleted area should be deleted. An inserted block should add only to the numbered line containing the insertion point. (The next H&J will reflow all, generating new line #s.) A block as saved to a register or copied elsewhere into the file should keep included pass-thru bytes/strings as follows:

- Four types of strings, along with their fencing bytes, should be

dropped completely: Error Message (<errm>,</errm>), XML Path (<xpth>,</xpth>), XML Coding (<xcod>,</xcod>), and Line-Start Groups (<line .../>).
- [If they are supported by PLedit: Another two types of strings, along with their fencing bytes, should be dropped completely: Level 1 Delete (<bde1>,</bde1>), and Level 2 Delete (<bde2>,</bde2>).]
- Two types of strings should remain in the saved block, but their fencing bytes may be dropped: XML tag (<xtag>,</xtag>) and XML Entity (<xent>,</xent>). NOTE: It is also acceptable to keep the fencing bytes.
- [If they are supported by PLedit: Another two types of strings should remain in the saved block, but their fencing bytes may be dropped: Level 1 Insert (<bin1>,</bin1>) and Level 2 Insert (<bin2>,</bin2>). NOTE: It is also acceptable to keep the fencing bytes.]
- All other pass-thru bytes and strings should be kept, along with all displayable bytes.

Macro Keys:

Ability to capture sequence of keystrokes into a register, so that the sequence can be repeated elsewhere in document or other document. For example, EditMaster method is:

- <define>, <ctrl><PgDown>, digit, digit, <enter> to start defining macro identified by the two digits.
- <ctrl><PgDown>, digit, digit, <enter> , while defining, to stop defining.
- <ctrl><PgDown>, digit, digit, <enter>, at any other time, to perform the macro identified by the two digits.

Method in PLedit may be completely different.

additional editing features

Command highlighting:

Our compostion commands should be highlighted. Command syntax is either of two forms: "[ab" with no arguments or closing bracket (there is a discreet list of two-letter commands of this form), and "[ab212,34,56,word,-5]" with arguments specific to that command, and a closing bracket. Currently H&J (with the txcontx2 converter) brackets each command with <cmd> and </cmd> pass-thru tags, as flags for highlighting. PLedit should highlight either this way, or by defining the commands as keywords using regexps in the editing mode (jEdit model). (See detailed string actions, later in "Pass-thru" bytes & strings section.)

Flash:

Ability to call our H&J engine on the textfile while it's open for editing ("flash" the file). The file and its buffer are locked from any activity while flash is active. When flash completes, pop up a dialog announcing completion, with "OK" box to click. Then re-display the whole file as re-flowed by H&J. Ability to flash file while it's closed (name still displayed in PLedit browser). Other files open in PLedit are not affected by a flash.

Composition statistics:

Each Line-Start Group contains values for four different composition results for the line, and three optional keywords with other composition results. Any one of the four should be displayed in the line-number gutter when user toggles that display mode. Display of one stat replaces display of any previous stat. Specifics:

LINE-ELEM		TOGGLE METHOD	
ATTRIBUTE	CONTENTS	VIEW MENU	KEYBOARD
lind=	Left indent width.	"Left Indent"	<Display>,<Left>
rind=	Right indent width.	"Right Indent"	<Display>,<Right>
bands=	Space-band width.	"Space Bands"	<Display>,<Space>
lead=	Depth of line.	"Line Depth"	<Display>,<Down>

Keyboard shortcut to turn off display of composition stats is <Suppress>,<Space> (or <Left>, <Right> or <Down>).

When "Space Bands" are toggled on, then program should also display in the gutter:

- "+" if flags= value "lett" exists
- "-" if flags= value "kern" exists (lett & kern are mutually exclusive)
- the gutter highlighted in a brighter color for that line, if flags= value "alert" exists.

Measure copy depth:

Ability to display an enhanced depth-of-line composition statistic: Accumulated depth of a group of lines. Method in EditMaster:

- Position cursor on first line of type to measure.
- <Define>, <Word>
- Cursor down to last line of type to measure.
- <Display>, <Down> causes depth of those lines to display at bottom of screen (same key sequence as used above under "Composition statistics" to display depth of one line).
- <Enter> acknowledges/removes the depth message at bottom
- <Define>, <Block>, <Space> cancels pending definition

Method in PLedit may be different.

Edit trace:

This refers to tracking all edit insertions and deletions, for two purposes:

1. Allows undo and examination of original text state in the current edit pass. Edit-trace history is wiped out upon each H&J processing. Using an editor with reliable undo/redo with deep history (which EditMaster is not), this tracking overhead should not be necessary.
2. Some users can require that all edits be tracked to two levels (ie. current and previous edit passes), and that page proofs be printed with that information highlighted in color: Insertions underlined and deletions marked with a caret, each history level in its own color. However, our programs have never successfully combined the permanent edit-trace history strings in a textfile with the

XMLPublisher pass-thru strings. There were always unstable overlaps. Thus, the two are not supported together. PubLink has only one client who uses this, for financial publications where accuracy of corrections is critical, and they do not and cannot use XMLPublisher with its textfile overhead of pass-thru strings.

For these two reasons, Edit Trace will be one of the last phases in development.

displayable character set

SEE FILE PubLink.Character.Encoding.doc (a MS Word doc) for full encoding of the displayable characters.

Standard 7-bit displayable ascii is used, although some characters are re-mapped to display upon alternate character sequences. Re-mapping is done for different reasons and is not always consistent. For instance: Ascii code 0x23 ("#") is a single-character representation of our common typesetting command [cm (format copy merge). But users also need to typeset the symbol "#". Thus:

- When encountered in the file, it should be displayed as simple #. H&J will treat it as [cm command.
- When keyed by operator, it should be stored as the sequence ";ns" (our "autosort" sequence for the number sign glyph to typeset) in the file, and displayed as an alternate stylized #.
- When user keys <esc>-M, program should display and record "[cm"

Similar transalations are currently done for:

+ " % < > = [ ] \ | @ ^ _

There are also 21 codes in the ascii range 0x1 through 0x1f which each have their own display symbols and methods of keyboard access.

PubLink's own "character entities", pre-dating SGML, are three-character sequences of form ";ab" called autosorts. Specific sequence template: semicolon, lower-case letter, any displayable ascii (range 0x21 – 0x7e). When encountered in a textfile, PLedit should replace certain autosort sequences by a single glyph, for operator convenience. Database of replacements will be supplied by PubLink.

keyboard mapping

While editing, a few keys are mapped to composition commands rather than expected editor functions. For instance:

KEY	EXPECTED ACTION	PLedit ACTION
<Enter>	<0x0a> and start new line	"[ep<0x0a>" and start new line
<Tab>	<0x09> and move right to tab	"[nt"
<Shift>-<Tab>	Move left to prev tab	"[bt"

We also have an extensive existing template of control-key combinations using <Shift>, <Control> and <Alt>, as well as sequences beginning with

<Esc>, <Home>, <F1>, <F2>, <F3> and <F4>. Many of these should be kept where possible; others will be replaced by PLedit's built-in editing functions.

For complete list, see PubLink's keyboard map file Keyboardlayout.EditMaster.odt (heading: "PubLink's EditMaster Keyboard Layout") showing existing keyboard functions and how each should migrate to PLedit. (See here for HTML version.)

Desirable feature (not mandatory): Ability to supply an alternate keyboard layout in future.

detailed actions on "pass-thru" bytes & strings

File-start group string:

Coding
: <header rev="-3" etrace="0x0" edtime="100" mod="1" hiline="999" sgml="3" nohnj="0" flrev="2"/> (sample values; see tx2filedoc.asc for expl.)

Purpose
: Starts each file. Provides meta-data about the file.

Source
: Genned by any app that creates a tx2 file. Some values may be modified by any app that touches the file.

Editable
: No

This string may include
: None

This string may nest within
: None

Action
: Always invisible. Items edtime, mod and hiline should be updated as necessary by PLedit, as noted in following detail.

- Item 'rev=': Revision level of this textfile structure. Set by creating app to -3. Would change to a different value for new files in future if file structure changes.
- Item 'etrace=' (edit trace): Keep at value 0x0. In future, may activate edit-trace functionality, as there are users (just one now) that want to maintain edit-trace between editing sessions and/or for generation of red-lining marks on the printed output.
- Item 'edtime=': Accumulate editing time in seconds. PLedit should track time the file is open for editing, and add that amount to this item at each save and close.
- Item 'mod=': PLedit should set this flag 1 when file is modified (H&J resets to 0).
- Item 'hiline=': PLedit should set this item to highest line number in the file, if operator appends to end of file, increasing the last line number.
- Item 'sgml=': The XMLPublisher input module sets this.
- Item 'nohnj=' (file has not been H&Jed): Filter apps can set this to 1 when the Line-Start Groups are stripped or zeroed as part of processing. In this state, file may keep Line-Start Groups with "dumb" sequential line numbering and nulled line status, or may keep only Line-Start Group number one, with or without NLs for line breaks.
- Item 'flrev=': H&J sets this to its own software rev level.

Line-start group strings:

Coding
<line num="1234" lind="120" rind="40" bands="108" lead="120" flags="cerr,alert,"/> (sample values; see tx2filedoc.asc for expl.)
Purpose
Starts each line and provides its number, as well as other H&J-built line status info.
Source
Genned by H&J. May also be genned by any app that creates or extends a tx2 file, with basic values of line number and zeroes (<line num="1234" lind="0" rind="0" bands="0" lead="0" flags=""/>). Line number MUST be a positive integer.
Editable
No
This string may include
None
This string may nest within
XML coding string, command string
Action
Always invisible. PLedit should capture line number for display in gutter. Also capture other four values and some keywords for optional display in gutter (see COMPOSITION STATISTICS above for usage).

Error message strings:

Coding
<errm>string</errm>
Purpose
Guides editor in locating problems. Also used to provide status info messages such as page number or column vertical-justify results.
Source
Genned by H&J at commands with syntax error, and at certain pagination transition points.
Editable
No
This string may include
None
This string may nest within
XML coding string, command string
Action
Toggle display/suppress of all Error Messages with View Menu choice or kybd commands <Display>,<Cmd Error> and <Suppress>,<Cmd Error>. When displayed, highlight as reverse-video (FG white on BG black) (tentative).

- style: reverse-video
- TOGGLE
- NO EDIT

Secondary mode strings:

Coding
<secmb/>string<secme/> (NOTE: Two empty element tags, not an element with content)
Purpose
Highlights text in "super-shift" mode. Typical use: Text is to typeset

in caps & small caps rather than caps & lower-case.

Source
: Begin/end passthru tags genned by H&J or PLedit upon occurence of "[sm" and "[nm" commands, respectively.

Editable
: Yes

This string may include
: (Does not affect parsing (not an element with content)) Line-start group, error message, XML tag, XML path, XML entity, XML coding, XML float string

This string may nest within
: XML coding string, XML float string

Action
: Always displayed. Highlight as underscored (tentative).

- style: underline
- multi-line
- bounded by two special empty XML elements

XML tag strings:

Coding
: <xtag><tago/>tagname attrib="val"<tagc/></xtag>

Purpose
: XML/SGML tag pass-thru string. Always followed by XML Coding string, containing optional typeset commands & text mapped onto the tag by XMLPublisher.

Source
: XML Publisher's sgmlp process.

Editable
: No

This string may include
: None

This string may nest within
: XML float string, command string

Action
: Toggle display/suppress of all XML Tags and XML Entities (together called markup) with View Menu choice or kybd command <ctrl>-<F3>. When displayed, highlight as blue-purple text (tentative).

- style: highlight blue-purple text
- TOGGLE (with XML Entities)
- NO EDIT

XML path strings:

Coding
: <xpth>document/body/para<xpth/> (sample string)

Purpose
: XML/SGML context path to a tag pass-thru string. Precedes every XML Tag string except the top-level tag.

Source
: XML Publisher's sgmlp process.

Editable
: No

This string may include
: None

This string may nest within
: XML float string, command string

Action
Toggle display/suppress of all XML Paths with View Menu choice or kybd command <ctrl>-<F4>. When displayed, highlight as underscored black text on colored BG (tentative).

- style: highlight as underscored black text on colored BG
- TOGGLE
- NO EDIT

XML entity strings:

Coding
<xent>&greekmu;</xent> (sample string)
Purpose
XML/SGML entity pass-thru string. Always followed by the displayable character translation text mapped onto the entity by XMLPublisher.
Source
XML Publisher's sgmlp process.
Editable
No
This string may include
None
This string may nest within
XML float string, command string
Action
Toggle display/suppress of all XML Entities and XML Tags (together called markup) with View Menu choice or kybd command <ctrl>-<F3>. When displayed, highlight as blue-purple text (tentative).

- style: highlight blue-purple text
- TOGGLE (with XML Tags)
- NO EDIT

XML coding strings:

Coding
<xcod>[fy51,1,10,12][iw0][xh</xcod> (sample string)
Purpose
XML/SGML coding mapped by XMLPublisher onto a tag or entity; a pass-thru string. Always immediately follows an XML Tag or XML Entity pass-thru string.
Source
XML Publisher's sgmlp process.
Editable
No
This string may include
Command string, line-start group, error message
This string may nest within
XML float string, command string
Action
Toggle display/suppress of all XML Coding strings with View Menu choice or kybd command <ctrl>-<F5>. When displayed, highlight as black text on colored BG (tentative).

- style: highlight as black text on colored BG
- TOGGLE
- NO EDIT
- multi-line

- includes other toggle/styled elements

XML float strings:

Coding
<xflt><xpth>document/body/para</xpth><xtag><tago/>tagname attrib="val"<tagc/></xtag><xcod>[fs1]A floating text element here, such as a defined footnote.
</xcod><xtag><tago/>/tagname<tagc/></xtag><xcod>[ep[fx</xcod></xflt> (sample string)

Purpose
XML/SGML element forked by XMLPublisher into a parallel "float" .tx2 file. The string as duplicated in the "float" file is processed by H&J/Pagination, while its source string in the originating .tx2 is treated by H&J as a comment. A FLOAT string usually includes other XML pass-thru strings (fenced by tags of form <x—-> and </x—->) within. Also used for other informational strings to be ignored by H&J.

Source
XML Publisher's sgmlp process.

Editable
Yes (but not the fencing <xflt> and </xflt> tags)

This string may include
All other pass-thru bytes and strings, except file-start group

This string may nest within
None

Action
None; treat all text/codes within as normal. Ie. this is not a string for the edit app, rather its two fencing tags are just two pass-thru tags. (The tags tell H&J to pass over this string during processing.)

Isn't there a problem for the user if you can't see where a xflt element starts and ends but can edit the text that it bounds? How can the caret know if it is just inside the end tag or just beyond it?
Not a problem for the user, who may edit inside or outside this string. In practice, a float string, even if very long, is always built within one numbered line (I think there are no exceptions – PRB). It is always associated with a few pagination commands, so user recognizes the "float" text as such.

- multi-line
- includes other toggle/styled elements
- NO SPECIAL HANDLING BY PLEDIT

Command strings:

Coding
<cmd>[fy51,1,10,12]</cmd> (sample string)

Purpose
To highlight composition commands for operator convenience.

Source
HandJ process.

Editable
Yes

This string may include
line-start group, error message, XML tag, XML path, XML entity, XML coding

This string may nest within

XML coding, XML float

Action

Highlight the enclosed string.

- style: highlight
- multi-line

Level-1 insert and level-2 insert strings:

Coding

<bin1/>string<ein1/> and <bin2/>string<ein2/>. (NOTE: Empty element tags bracket each string, not an element with content)

Purpose

Flags text that has been inserted in editor. (See Edit Trace editing feature, above.)

Source

The editor

Editable

Yes

This string may include / nest within

In current edit pass (prior to H&J processing), may overlap and nest with other editable strings. For most users, the edit-trace info is dropped upon next H&J. If user wants to maintain edit-trace (few do), then after H&J, anything at all may nest within, and some strings may overlap.

Action

Always displayed. Toggle highlight of the two levels of inserted strings (and display/suppress of deleted strings) with View Menu choices or kybd commands <Display>,<Curr Edits> and <Suppress>,<Curr Edits> for level 1, and <Display>,<Prev Edits> and <Suppress>,<Prev Edits> for level 2.

Dazed and confused. When we get to Edit Trace, I need a phone call with Peter to walk through this carefully. Ditto the delete strings. If we maintain edit trace info through H&J, how stable is its position wrt <line.../>?

Level-1 delete and level-2 delete strings:

Coding

<bde1/>string<ede1/> and <bde2/>string<ede2/>. (NOTE: Empty element tags bracket each string, not an element with content)

Purpose

Contains text that has been deleted in editor. (See Edit Trace editing feature, above.)

Source

The editor

Editable

No

This string may include / nest within

In current edit pass (prior to H&J processing), may overlap and nest with other editable strings. For most users, the edit-trace info is dropped upon next H&J. If user wants to maintain edit-trace (few do), then after H&J, anything at all may nest within, and some strings may overlap.

Action

Normally suppressed. Toggle display/suppress of the two levels of deleted strings (and highlight of inserted strings) with View Menu

choices or kybd commands <Display>,<Curr Edits> and <Suppress>,<Curr Edits> for level 1, and <Display>,<Prev Edits> and <Suppress>,<Prev Edits> for level 2.

documentation

PLedit should be shipped with full on-line Help and with a usage manual. PubLink will prepare both based on these specs as much as possible. Both would grow along with the developing program, based on testing the program and on any program documentation provided by the developer.