

```

BEGIN

    VARIABLES ARE UNDEFINED

    %CURRENT      IS STRING LEN 120
    %IPADDR       IS STRING LEN 16
    %PS1          IS STRING LEN 255
    %PAIR         IS STRING LEN 255
    %TIME         IS STRING LEN 20

    %TAG_FOUND    IS FLOAT
    %LEVEL        IS FLOAT
    %LIST         IS FLOAT
    %LIST2        IS FLOAT
    %H            IS FLOAT
    %X            IS FLOAT
    %Y            IS FLOAT
    %NODELEVEL    IS FIXED
    %CONTENT      IS STRING LEN 60
    %XMLNODE      IS STRING LEN 30
    %XMLATTRIB    IS STRING LEN 40

    %IPADDR = fWEB_IPADDR
    %TIME   = fWEB_DATENS
    %X      = fSETG('NEXTPROC', 'STOP')

*... create a list to save the parsed XML tag=value pairs
    %LIST2 = fLISTNEW

*... retrieve the incoming XML into a fLIST.
    %LIST = fLISTNEW
    %LIST2 = fLISTNEW
    %X     = fWEB_LIST_RECV(%LIST, , , 'TEXT')

    %LEVEL = 0

*... loop over the entire xml document.
    FOR %Y FROM 1 TO fLISTCNT(%LIST)
        %PS1 = fDEBLANK(fLISTINF(%LIST,%Y),1)

        REPEAT WHILE fLEN(%PS1) GT 0
            *... is this a tag?
            IF fINDEX(%PS1,'<') EQ 1 THEN
                %CURRENT = fPARSE(%PS1,'>')
                %CURRENT = fPARSEX(%CURRENT,'<')
                %PS1      = fPARSEX(%PS1,'>')

                *... we can ignore comment tags.
                IF fONEOF(fSUBSTR(%CURRENT,1,1),'?!','/') THEN
                    JUMP TO KEEP_PARSING
                END IF

                *... notice when we've hit a close tag.
                IF fINDEX(%CURRENT,'/') EQ 1 THEN
                    %LEVEL = %LEVEL - 1

                    IF %LEVEL EQ 0 THEN
                        *... We could do some special processing when we
                        *    hit end-of-xml-stream, but not sure what yet.
                        END IF
                        JUMP TO KEEP_PARSING
                    END IF
                END IF
            END IF
        END IF
    END IF

```

```

    *... okay, it's a valid tag.
    %LEVEL          = %LEVEL + 1
    %XMLNODE        = fUPCASE(fPARSE(%CURRENT,' '))
    %XMLATTRIB      = fPARSEX(%CURRENT,' ')
    %NODELEVEL      = %LEVEL
    %TAG_FOUND      = 1

    *... If it's the root node we use the element name
    *      as the name of the procedure to call.
    IF %LEVEL EQ 1 THEN
        %X = fSETG('NEXTPROC','PDP.BS.' WITH %XMLNODE)
    END IF

    *... else we're parsing content, which we stash in
    *      a flist as tag=element content.
ELSE

    %CONTENT = fPARSE(%PS1,'<')

    *... save the tag=value pairs in the list
    IF NOT fDEBLANK(%CONTENT) = '' -
        AND %TAG_FOUND THEN
        %PAIR = %XMLNODE WITH '=' WITH %CONTENT
        %X = fLISTADD(%LIST2,%PAIR)
        %TAG_FOUND = 0
    END IF

    %PS1 = fSUBSTR(%PS1,fLEN(%CONTENT)+1)
END IF
KEEP_PARSING:
    END REPEAT
END FOR

END_OF_DOC:

*      This subroutine can be turned on for diagnostic display.
*      CALL DIAGNOSTICS

    %X = fLISTSAVE(%LIST2,'PARMLIST')

*****
DIAGNOSTICS: SUBROUTINE
*****
    FOR %X FROM 1 TO fLISTCNT(%LIST2)
    AUDIT 'LIST' AND fLISTINF(%LIST2,%X) WITH '*'
    END FOR
END SUBROUTINE DIAGNOSTICS

END

```