

```

    ctl-opt dftactgrp(*no)          option(*nodebugio);

    // Note - Since this is using free form this is for use on a 7.1 system
    // normally when calling a local subprocedure the prototype is optional

    dcl-pr  Xmlhandler      int(10);
           MyCommArea      char(100);
           CityData        likes(citydata) dim(3) const;
           CitiesParsed    int(10)  value;

    dcl-ds  Cities          qualified;
           CityData        likes(CityData)  dim(3);
    end-ds;

    dcl-ds  CityData        qualified;
           CityName        char(20);
           Region          char(20);
           MonthlyData      likes(MonthlyDataDS) dim(
3);
    end-ds;

    dcl-ds  MonthlyDataDS;
           Month           char(9);
           Low             char(3);
           High            char(3);

    dcl-s  options1        char(100);
    dcl-s  filename        varchar(25)      inz('/xml/docs/citydata2.xml');

    // MyCommArea can be defined as any type/size you'd like.
    // It is used to communicate to the handler.
    dcl-s  MyCommArea      char(100);

    // Setup parsing options
    options1 = 'doc=file +
allowextra=yes allowmissing=yes case=any path=Cities/
CityData';

```

```

// Capture header data
xml-into %handler(XMLHandler:MyCommArea) %xml(filename:options1);

*inlr = *on;
return;

// This xml handler will be called each time the cities array fills up.
// Will return back to xml-into statement to continue until no data left to parse.
// Parsed data will be written to local file citydata2
// "STATIC" keyword not used on F spec, file will be opened and closed each time.
// Note - Local files require V6.1

dcl-proc Xmlhandler;

dcl-f citydata2 disk(*ext) usage(*output) qualified
rename(citydata2:citydatar);

dcl-ds CityDataRDS likerec(citydata2.citydatar:*output);

// When declaring a procedure interface, either specify the name
// of the procedure you are referencing or *n if you don't want to.
dcl-pi *n int(10);
MyCommArea char(100);
CityData likeds(citydata) dim(3) const;
CitiesParsed int(10) value;

dcl-s x packed(2:0);
dcl-s y packed(2:0);

// Each time this handler is executed, write parsed data to file

```

```
// There will be 3 months of data for each city processed.
for x = 1 to citiesparsed;
    citydatards.cityname = cities(x).cityname;
    citydatards.region = cities(x).region;

    for y = 1 to 3;
        citydatards.monthname = cities(x).monthlydata(y).month;
        citydatards.low = cities(x).monthlydata(y).low;
        citydatards.high = cities(x).monthlydata(y).high;
        write citydata2.citydatar citydatards;
    endfor;
endfor;

return 0;

end-proc;
```