PUG CHALLENGE EXCHANGE AMERICAS

296: Everyday OOABL

Oct 7, 2019

Chad R. Thomson
Senior Principal Consultant
Progress Software

Speaker BIO

- Over 20 years of industry experience, favoring reality over formality
- Specializing in vendor-neutral, cross-platform application and service integration
- Passionate technology advocate
- Session code examples and slide-updates available in GitHub repo:
  - https://github.com/ChadThomsonPSC/pugna-2019-everyday-oop

Only those who have the patience to do simple things perfectly will acquire the skill to do difficult things easily

Agenda

- Why Bother with OOABL?
- Setting up for Success
- OOABL Primer
- Procedural vs OOABL Comparison
- Every-day Use Examples
- Next Steps

Why Bother with OOABL?

- OOABL will enforce method and type checking during **compilation**
  - Doesn't fix code/logic, but mitigates opportunity for failure
  - Even the most experienced developers still encounter run-time errors when procedure signatures don't match
- You want to **modernize**, right?
  - PASOE web handlers, (REST) BusinessEntity – Class-based
- Object caching and memory management, superior to procedural counter-parts
  - Garbage collection assists in *limiting* the amount of clean-up concerns
  - Please, pick-up after yourself, though
- Inheritance and static classes can be simple scaffolding for application architecture
  - reduces the amount of traditional "framework" management code
    - eg. super-procedure-based: run, persistent, make super, handle locate, repeat

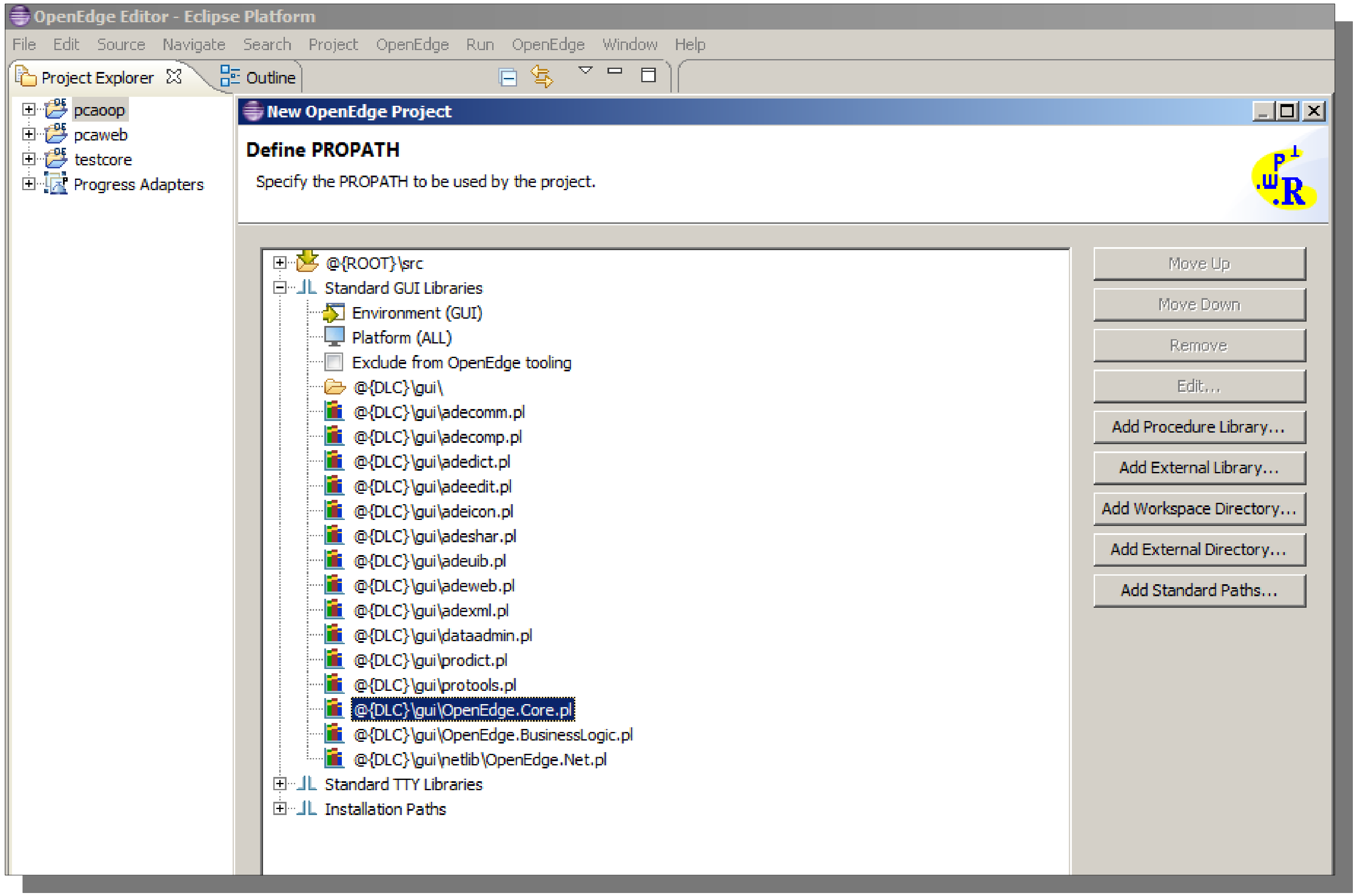
Why Bother: ...it’s too difficult

- It doesn't have to be
  - It can be as simple or complex as you want or need it to be
- Don't get caught-up in the **academics** of it all
  - Design Patterns
    - Static Singleton Factory Provider Pattern
    - Inversion of Control, Dependency Injection
    - Polymorphism
    - Wait, what?
- Start with small, simple tasks
  - Global functions
  - Logging
  - Messaging
- Slowly **mix-in** OOABL to your procedural code

PUGCHALLENGE AMERICAS EXCHANGE

Setting Up for Success

Setting Up for Success: the IDE



Define PROPATH

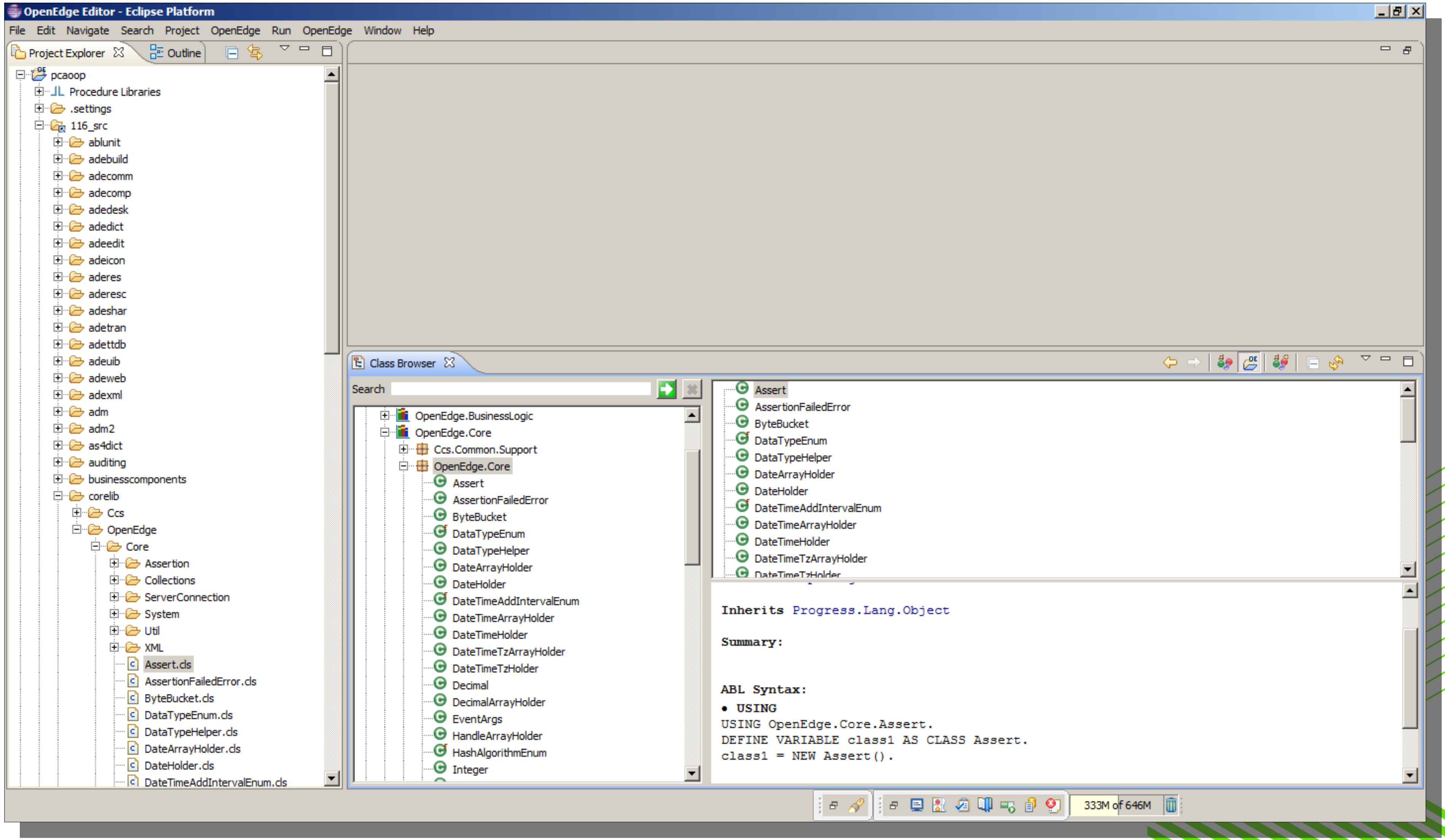
Specify the PROPATH to be used by the project.

```
@{ROOT}\src
Standard GUI Libraries
    Environment (GUI)
    Platform (ALL)
    Exclude from OpenEdge tooling
    @{DLC}\gui\
    @{DLC}\gui\adecomm.pl
    @{DLC}\gui\adecomp.pl
    @{DLC}\gui\adedict.pl
    @{DLC}\gui\adeedit.pl
    @{DLC}\gui\adeicon.pl
    @{DLC}\gui\adeshar.pl
    @{DLC}\gui\adeuib.pl
    @{DLC}\gui\adeweb.pl
    @{DLC}\gui\adexml.pl
    @{DLC}\gui\dataadmin.pl
    @{DLC}\gui\prodict.pl
    @{DLC}\gui\protools.pl
    @{DLC}\gui\OpenEdge.Core.pl
    @{DLC}\gui\OpenEdge.BusinessLogic.pl
    @{DLC}\gui\netlib\OpenEdge.Net.pl
Standard TTY Libraries
Installation Paths
```

- Use **PDSOE** for projects
  - Convenience libraries usually already included in PROPATH
  - OpenEdge.Core.pl
  - OpenEdge.BusinessLogic.pl
  - OpenEdge.Net.pl

Setting Up for Success: Get the Source, View the Source

- Get the **ADE Dev Tools** source
- https://community.progress.com
- Link in ADE Dev Tools source
- Use the Class Browser



Inherits Progress.Lang.Object

Summary:

ABL Syntax:

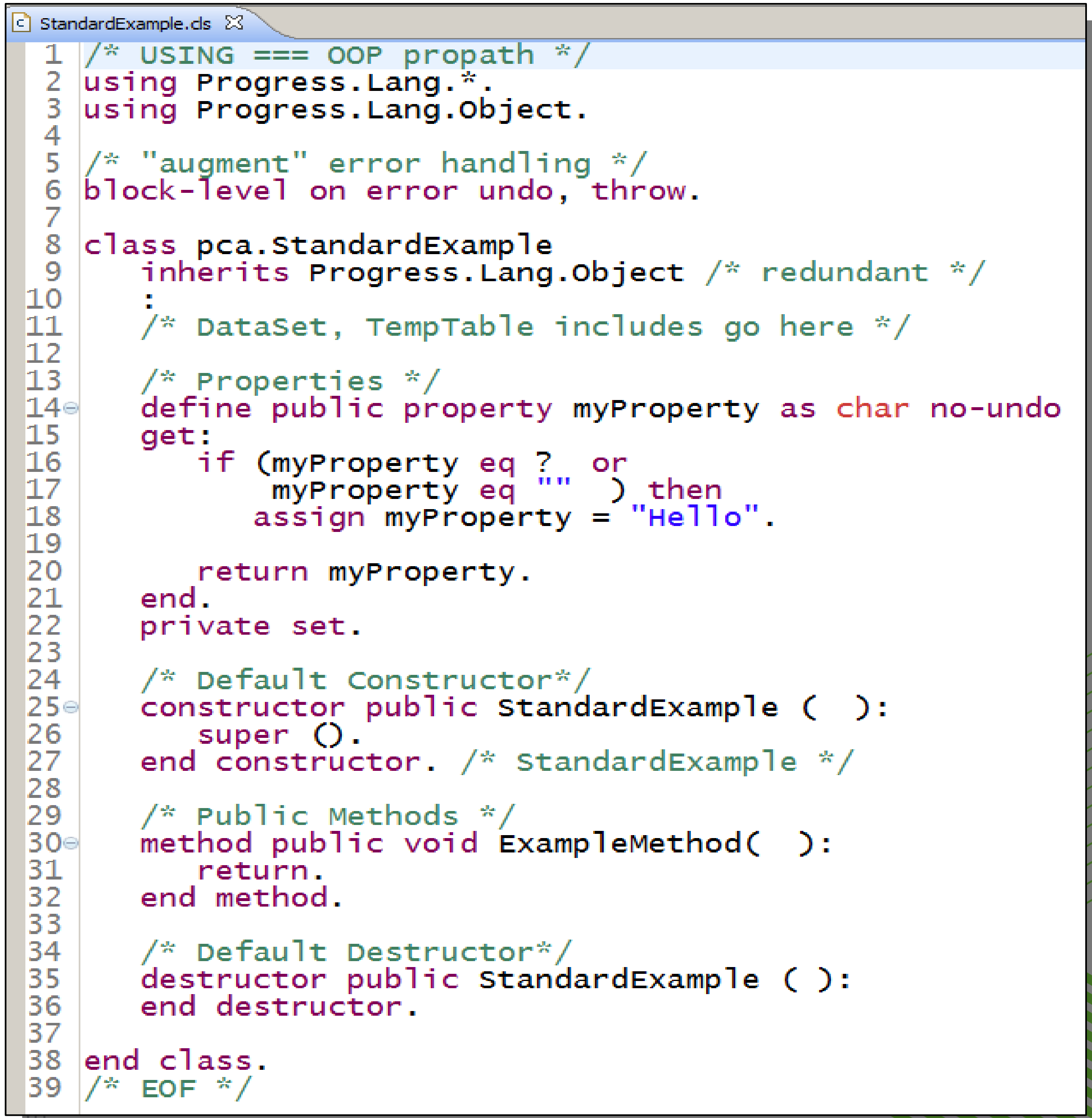
- **USING**

```
USING OpenEdge.Core.Assert.
DEFINE VARIABLE class1 AS CLASS Assert.
class1 = NEW Assert().
```

OOABL Primer

OOABL Primer: Anatomy of a Class

- **using** === OOP Propath
  - @ line 2,3
  - Replace "." with "/" to find Class r-code
  - Exception "Progress.*"
- Class "**Package**" or "**Namespace**"
  - @ line 8
  - Means: path to folder containing class files
- **Inherits** "Progress.Lang.Object" by default
  - @ line 9

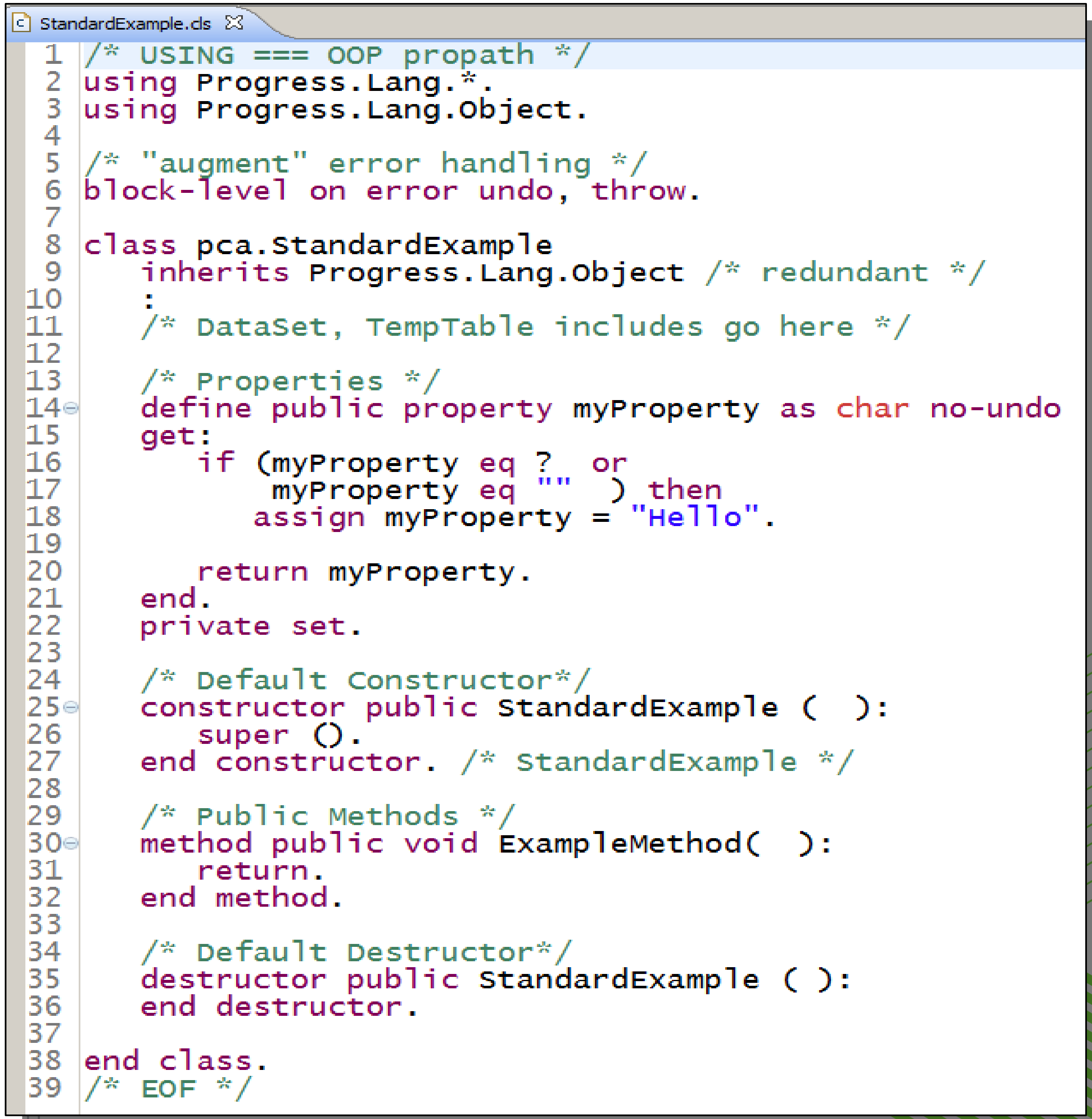


```
 1 /* USING === OOP propath */
 2 using Progress.Lang.*.
 3 using Progress.Lang.Object.
 4
 5 /* "augment" error handling */
 6 block-level on error undo, throw.
 7
 8 class pca.StandardExample
 9    inherits Progress.Lang.Object /* redundant */
10    :
11    /* DataSet, TempTable includes go here */
12
13    /* Properties */
14    define public property myProperty as char no-undo
15    get:
16       if (myProperty eq ?  or
17           myProperty eq ""   ) then
18          assign myProperty = "Hello".
19
20       return myProperty.
21    end.
22    private set.
23
24    /* Default Constructor*/
25    constructor public StandardExample (   ):
26       super ().
27    end constructor. /* StandardExample */
28
29    /* Public Methods */
30    method public void ExampleMethod(  ):
31       return.
32    end method.
33
34    /* Default Destructor*/
35    destructor public StandardExample ( ):
36    end destructor.
37
38 end class.
39 /* EOF */
```

OOABL Primer: Anatomy of a Class

- Variables, Datasets, etc
  - @ line 11
  - Called: [**Data-]Members**
- **Properties**
  - @ line 14
  - Variables with built-in logic
- **Constructor**, **Destructor**
  - @ line 25
  - Optional*
  - * *based-on super-class*
- Member Accessibility modes
  - @ line 30
  - **Public**, **Private**, **Protected**

PUGCHALLENGE EXCHANGE AMERICAS



StandardExample.cls

```
 1 /* USING === OOP propath */
 2 using Progress.Lang.*.
 3 using Progress.Lang.Object.
 4
 5 /* "augment" error handling */
 6 block-level on error undo, throw.
 7
 8 class pca.StandardExample
 9    inherits Progress.Lang.Object /* redundant */
10    :
11    /* DataSet, TempTable includes go here */
12
13    /* Properties */
14    define public property myProperty as char no-undo
15    get:
16       if (myProperty eq ?  or
17           myProperty eq ""  ) then
18          assign myProperty = "Hello".
19
20       return myProperty.
21    end.
22    private set.
23
24    /* Default Constructor*/
25    constructor public StandardExample (  ):
26       super ().
27    end constructor. /* StandardExample */
28
29    /* Public Methods */
30    method public void ExampleMethod(  ):
31       return.
32    end method.
33
34    /* Default Destructor*/
35    destructor public StandardExample ( ):
36    end destructor.
37
38 end class.
39 /* EOF */
```

OOABL Primer: Error Handling – Catch, Handle, and Throw

- Very convenient to start using **THROW**, **CATCH**
  - There is no Try, only blocks that THROW vs raise errors
  - Fewer lines of 'if .. then … else'-style error processing logic

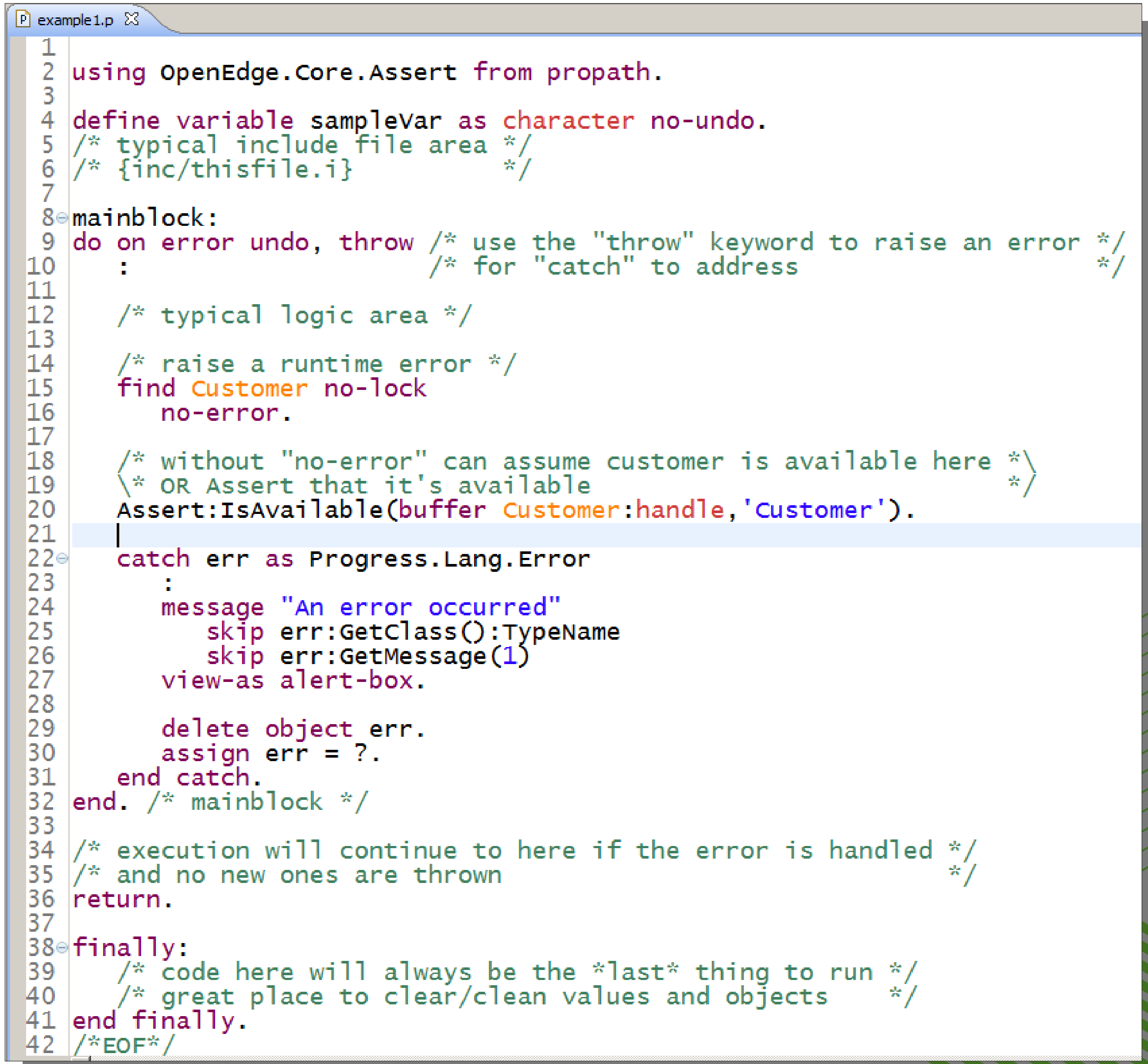
- Change your *mindset*
  - let errors happen, and handle vs test for them

- **FINALLY**
  - Runs as the *very last line* of executable code prior to procedure or block exiting completely
  - A good place for 'clean-up' logic

- Use of these error handling methods is a **requirement** when using Assertions (later)

OOABL Primer: Error Handling Mix-in

- **Throw**
  - Added to blocks
  - Creates error object
- **Catch**
  - Inside blocks that Throw
  - Passed error object for handling
- **Finally**
  - <u>LAST</u> block to execute
    - Even after 'return' statement
    - Can be inside other blocks
  - Convenient place to clean-up objects, handles, etc

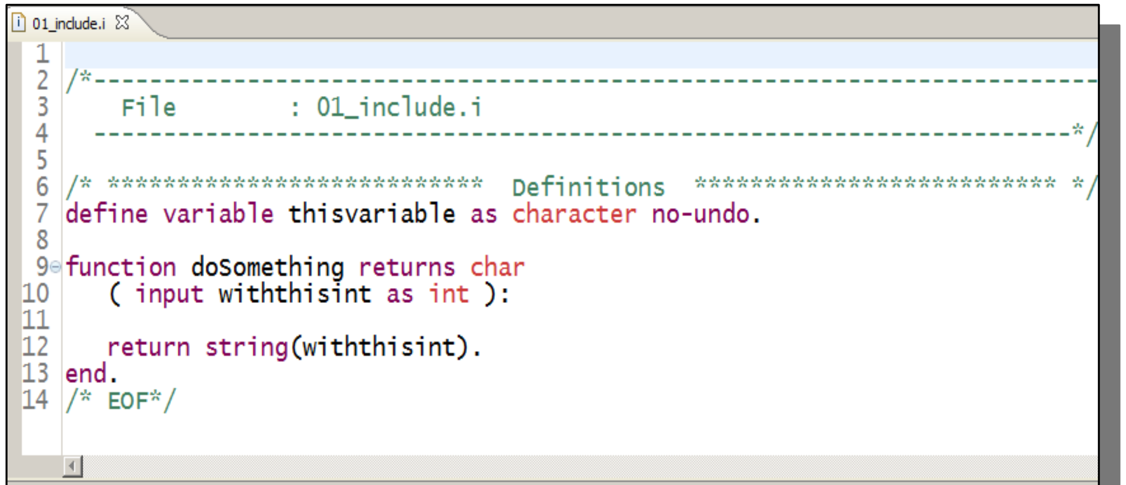

example1.p

```
using OpenEdge.Core.Assert from propath.

define variable sampleVar as character no-undo.
/* typical include file area */
/* {inc/thisfile.i}          */

mainblock:
do on error undo, throw /* use the "throw" keyword to raise an error */
   :                    /* for "catch" to address                      */

   /* typical logic area */

   /* raise a runtime error */
   find Customer no-lock
      no-error.

   /* without "no-error" can assume customer is available here *\
   \* OR Assert that it's available                            */
   Assert:IsAvailable(buffer Customer:handle,'Customer').

   catch err as Progress.Lang.Error
      :
      message "An error occurred"
          skip err:GetClass():TypeName
          skip err:GetMessage(1)
      view-as alert-box.

      delete object err.
      assign err = ?.
   end catch.
end. /* mainblock */

/* execution will continue to here if the error is handled */
/* and no new ones are thrown                              */
return.

finally:
   /* code here will always be the *last* thing to run */
   /* great place to clear/clean values and objects    */
end finally.
/*EOF*/
```

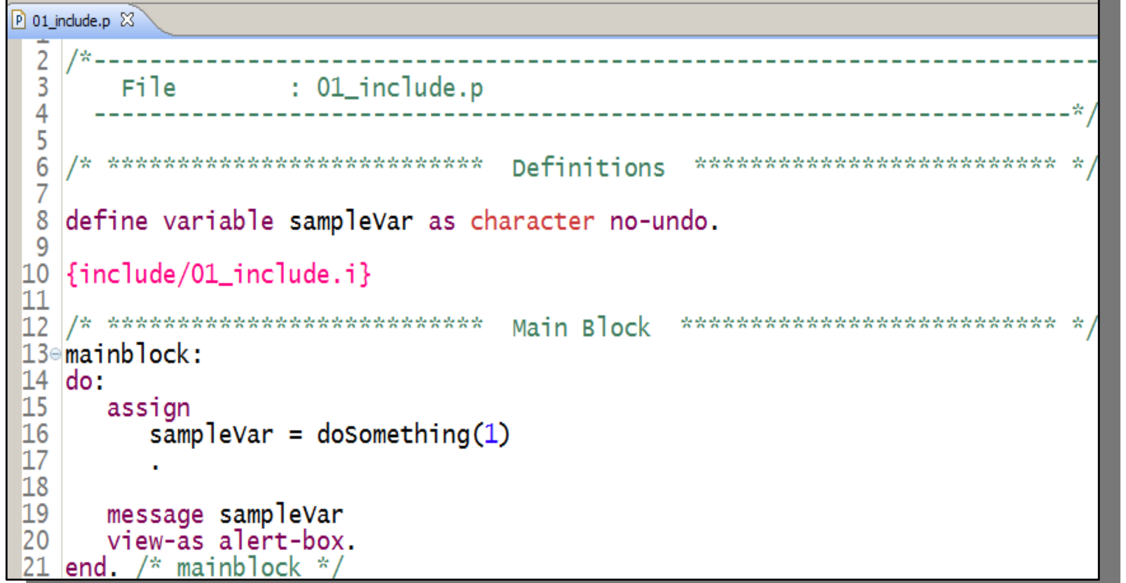
Procedural vs OOABL Comparison

Procedural vs OOABL Comparison: Include vs Inherits

Procedural Include

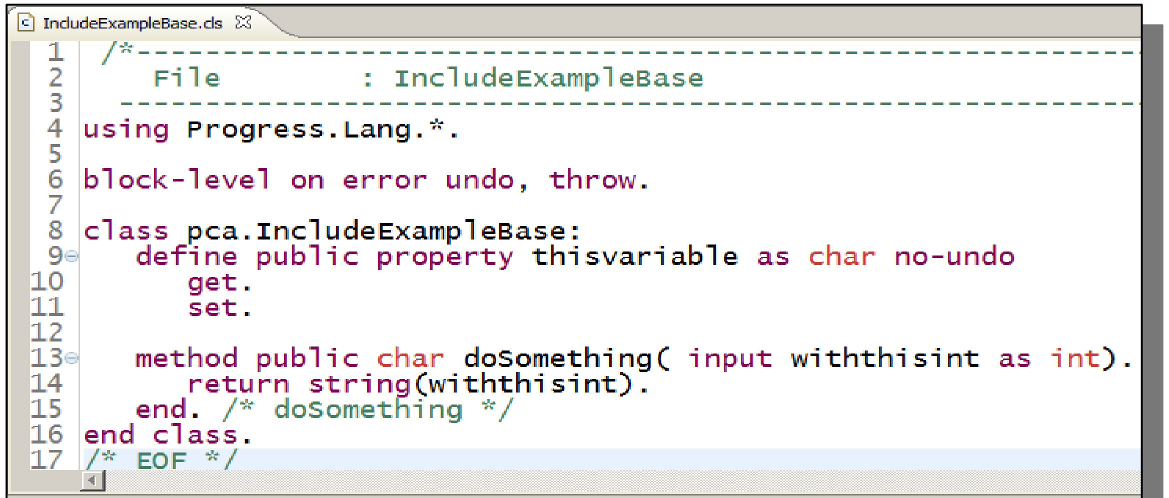


```
/*------------------------------------------------------------------------
    File        : 01_include.i
  ----------------------------------------------------------------------*/

/* ***************************  Definitions  ************************** */
define variable thisvariable as character no-undo.

function doSomething returns char
   ( input withthisint as int ):

   return string(withthisint).
end.
/* EOF*/
```

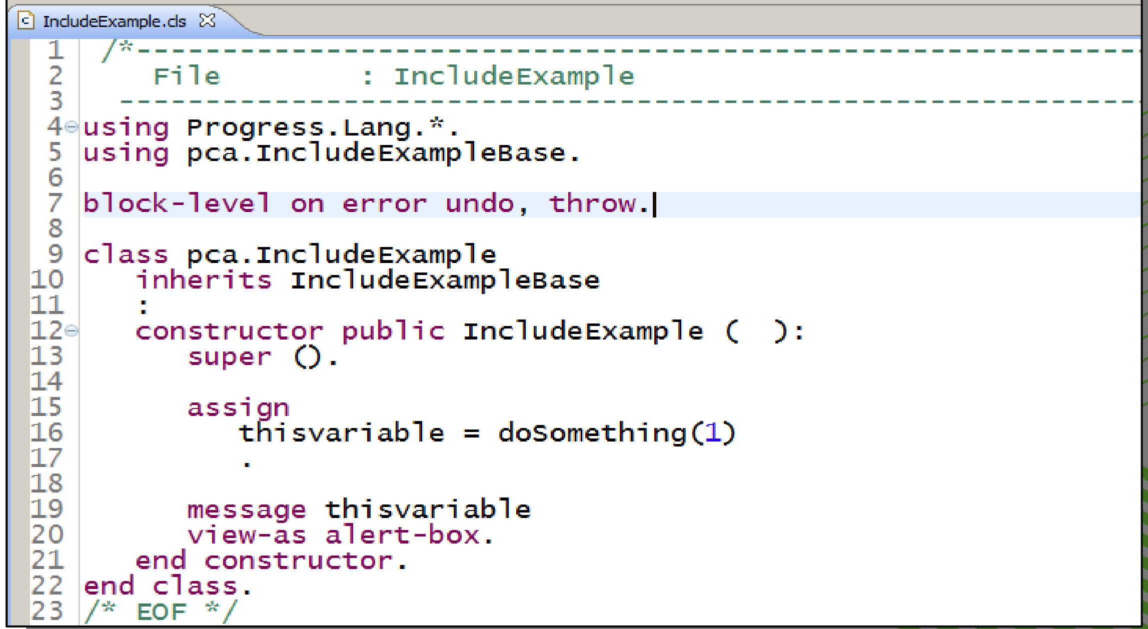


```
/*------------------------------------------------------------------------
    File        : 01_include.p
  ----------------------------------------------------------------------*/

/* ***************************  Definitions  ************************** */

define variable sampleVar as character no-undo.

{include/01_include.i}

/* ***************************  Main Block  *************************** */
mainblock:
do:
   assign
      sampleVar = doSomething(1)
      .

   message sampleVar
   view-as alert-box.
end. /* mainblock */
```

OOABL Inherits



```
/*------------------------------------------------------------------------
    File        : IncludeExampleBase
  ------------------------------------------------------------------------
using Progress.Lang.*.

block-level on error undo, throw.

class pca.IncludeExampleBase:
    define public property thisvariable as char no-undo
        get.
        set.

    method public char doSomething( input withthisint as int).
        return string(withthisint).
    end. /* doSomething */
end class.
/* EOF */
```

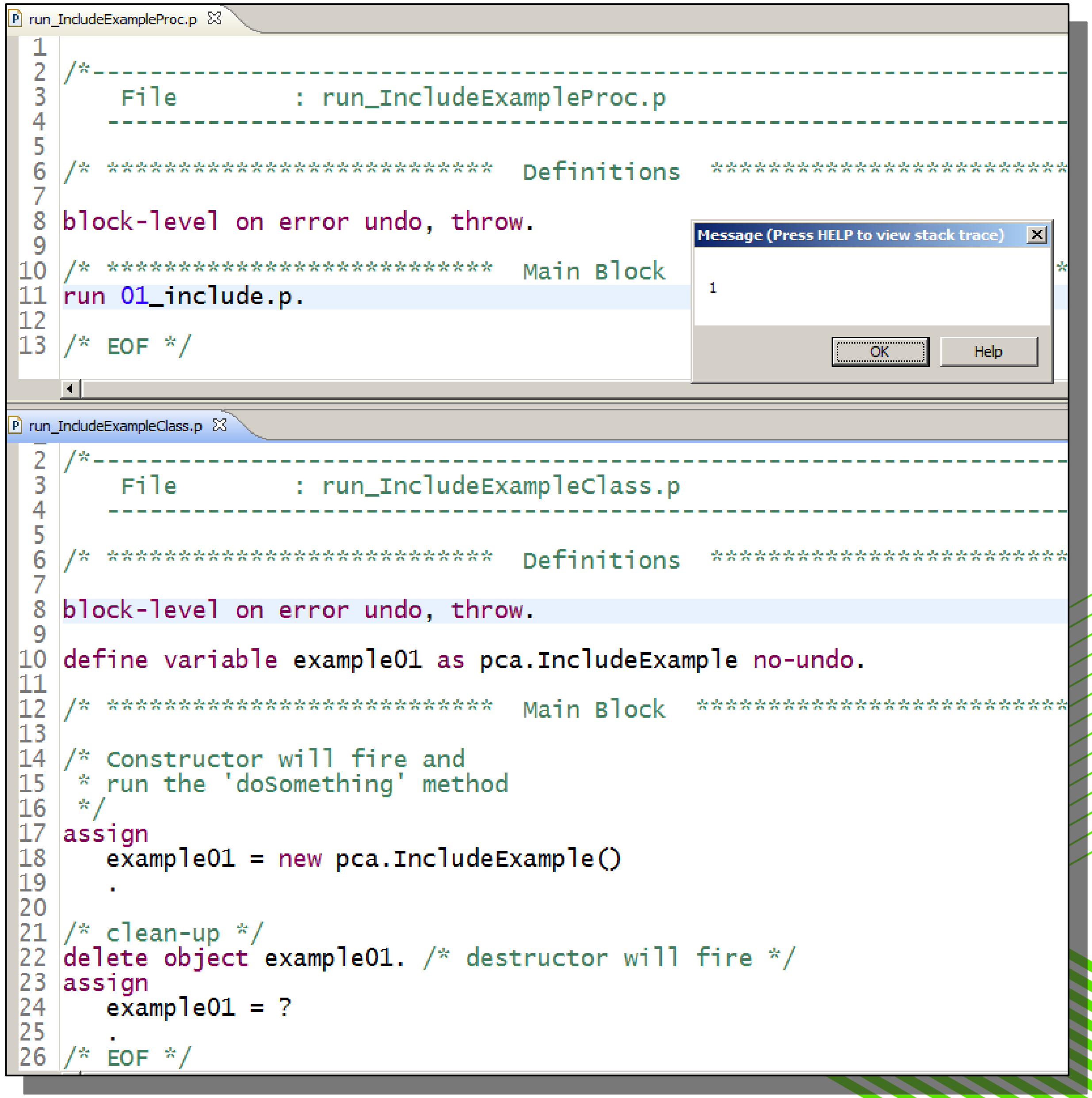


```
/*------------------------------------------------------------------------
    File        : IncludeExample
  ------------------------------------------------------------------------
using Progress.Lang.*.
using pca.IncludeExampleBase.

block-level on error undo, throw.

class pca.IncludeExample
    inherits IncludeExampleBase
    :
    constructor public IncludeExample (  ):
        super ().

        assign
            thisvariable = doSomething(1)
            .

        message thisvariable
        view-as alert-box.
    end constructor.
end class.
/* EOF */
```

Procedural vs OOABL Comparison: Run it!

- Procedural
  - Use "run" statement

- OOABL
  - create a **NEW** "instance"
  - Be sure to delete your object
  - A more Apples-to-apples example
    - Better to call "logic" in a method
    - eg: "<charvar> = example01:doSomething(1)"



run_IncludeExampleProc.p

```

/*------------------------------------------------------------------
    File        : run_IncludeExampleProc.p
   ----------------------------------------------------------------

/* ***************************  Definitions  **************************

block-level on error undo, throw.

/* ***************************  Main Block  ***************************
run 01_include.p.

/* EOF */
```

Message (Press HELP to view stack trace)

```
1
```

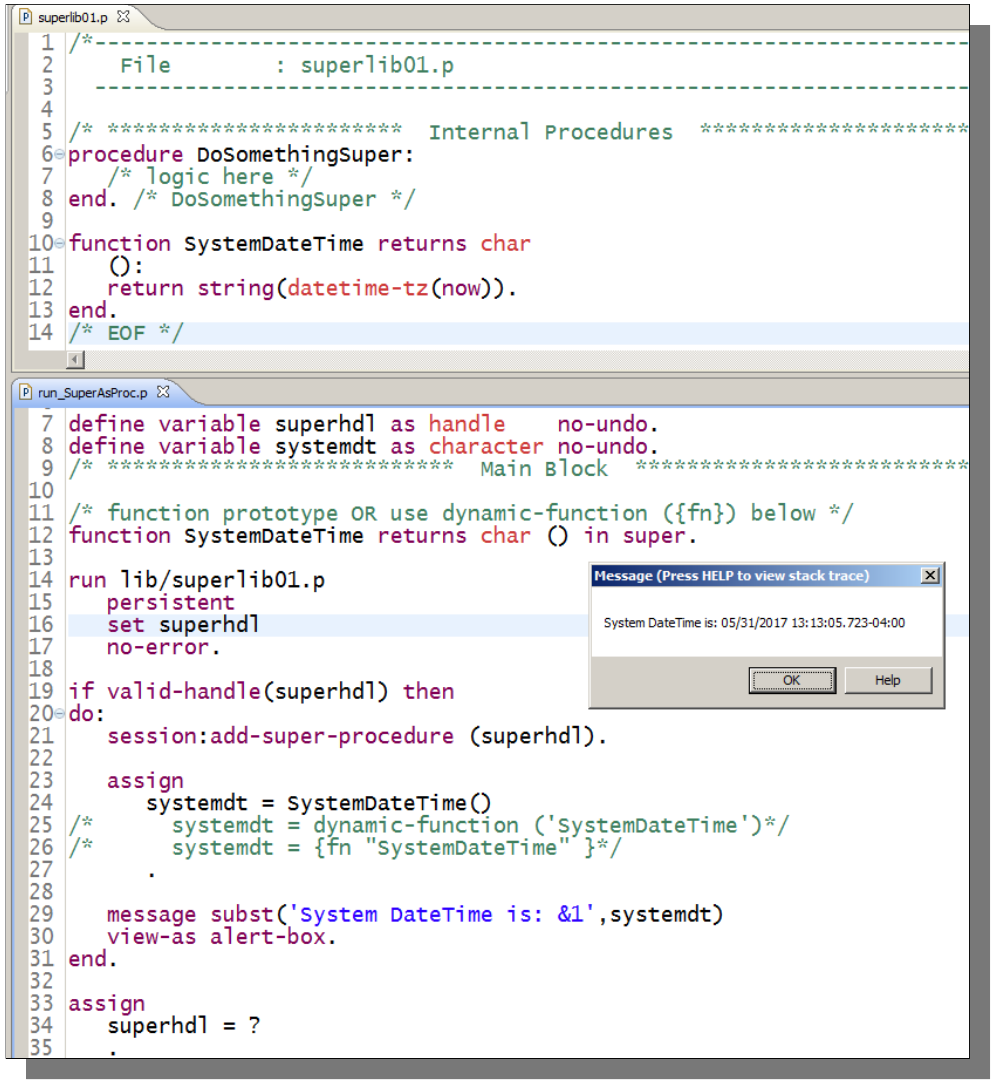
OK Help

run_IncludeExampleClass.p

```
/*------------------------------------------------------------------
    File        : run_IncludeExampleClass.p
   ----------------------------------------------------------------

/* ***************************  Definitions  **************************

block-level on error undo, throw.

define variable example01 as pca.IncludeExample no-undo.

/* ***************************  Main Block  *************************

/* Constructor will fire and
 * run the 'doSomething' method
 */
assign
   example01 = new pca.IncludeExample()
   .

/* clean-up */
delete object example01. /* destructor will fire */
assign
   example01 = ?
   .
/* EOF */
```

Procedural vs OOABL Comparison: Super vs Static

Procedural Super

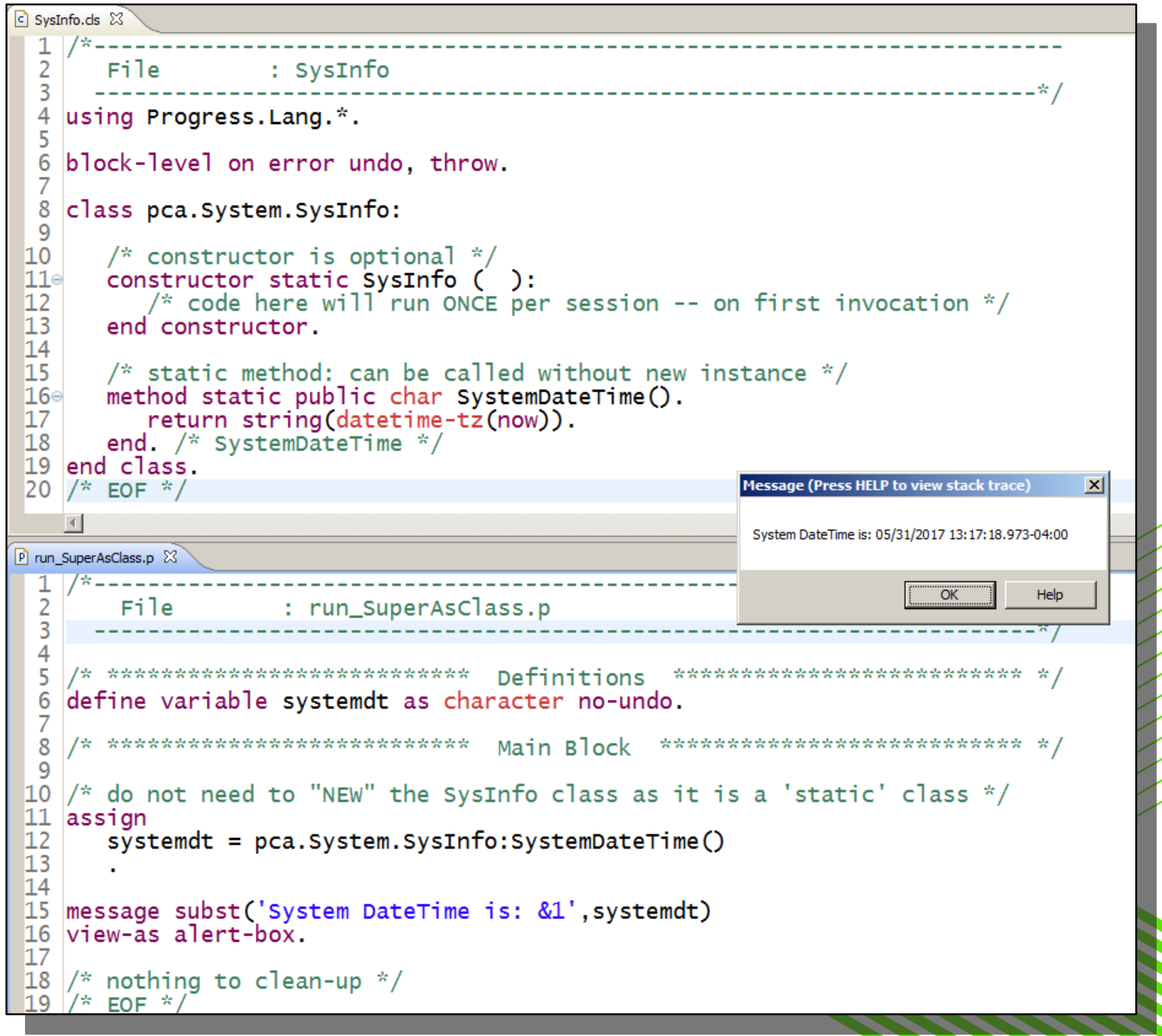


```
/*-----------------------------------------------------------------------
   File        : superlib01.p
  -----------------------------------------------------------------------

/* ***********************  Internal Procedures  ********************
procedure DoSomethingSuper:
   /* logic here */
end. /* DoSomethingSuper */

function SystemDateTime returns char
   ():
   return string(datetime-tz(now)).
end.
/* EOF */
```

```
define variable superhdl as handle    no-undo.
define variable systemdt as character no-undo.
/* ***************************  Main Block  ***************************

/* function prototype OR use dynamic-function ({fn}) below */
function SystemDateTime returns char () in super.

run lib/superlib01.p
   persistent
   set superhdl
   no-error.

if valid-handle(superhdl) then
do:
   session:add-super-procedure (superhdl).

   assign
      systemdt = SystemDateTime()
/*      systemdt = dynamic-function ('SystemDateTime')*/
/*      systemdt = {fn "SystemDateTime" }*/
      .

   message subst('System DateTime is: &1',systemdt)
   view-as alert-box.
end.

assign
   superhdl = ?
   .
```

```
System DateTime is: 05/31/2017 13:13:05.723-04:00
```

OOABL Static



```
/*------------------------------------------------------------------------
   File        : SysInfo
  ----------------------------------------------------------------------*/
using Progress.Lang.*.

block-level on error undo, throw.

class pca.System.SysInfo:

   /* constructor is optional */
   constructor static SysInfo (  ):
      /* code here will run ONCE per session -- on first invocation */
   end constructor.

   /* static method: can be called without new instance */
   method static public char SystemDateTime().
      return string(datetime-tz(now)).
   end. /* SystemDateTime */
end class.
/* EOF */
```

```
System DateTime is: 05/31/2017 13:17:18.973-04:00
```

```
/*------------------------------------------------------------------------
    File        : run_SuperAsClass.p
  ----------------------------------------------------------------------*/

/* ***************************  Definitions  ************************** */
define variable systemdt as character no-undo.

/* ***************************  Main Block  *************************** */

/* do not need to "NEW" the SysInfo class as it is a 'static' class */
assign
   systemdt = pca.System.SysInfo:SystemDateTime()
   .

message subst('System DateTime is: &1',systemdt)
view-as alert-box.

/* nothing to clean-up */
/* EOF */
```

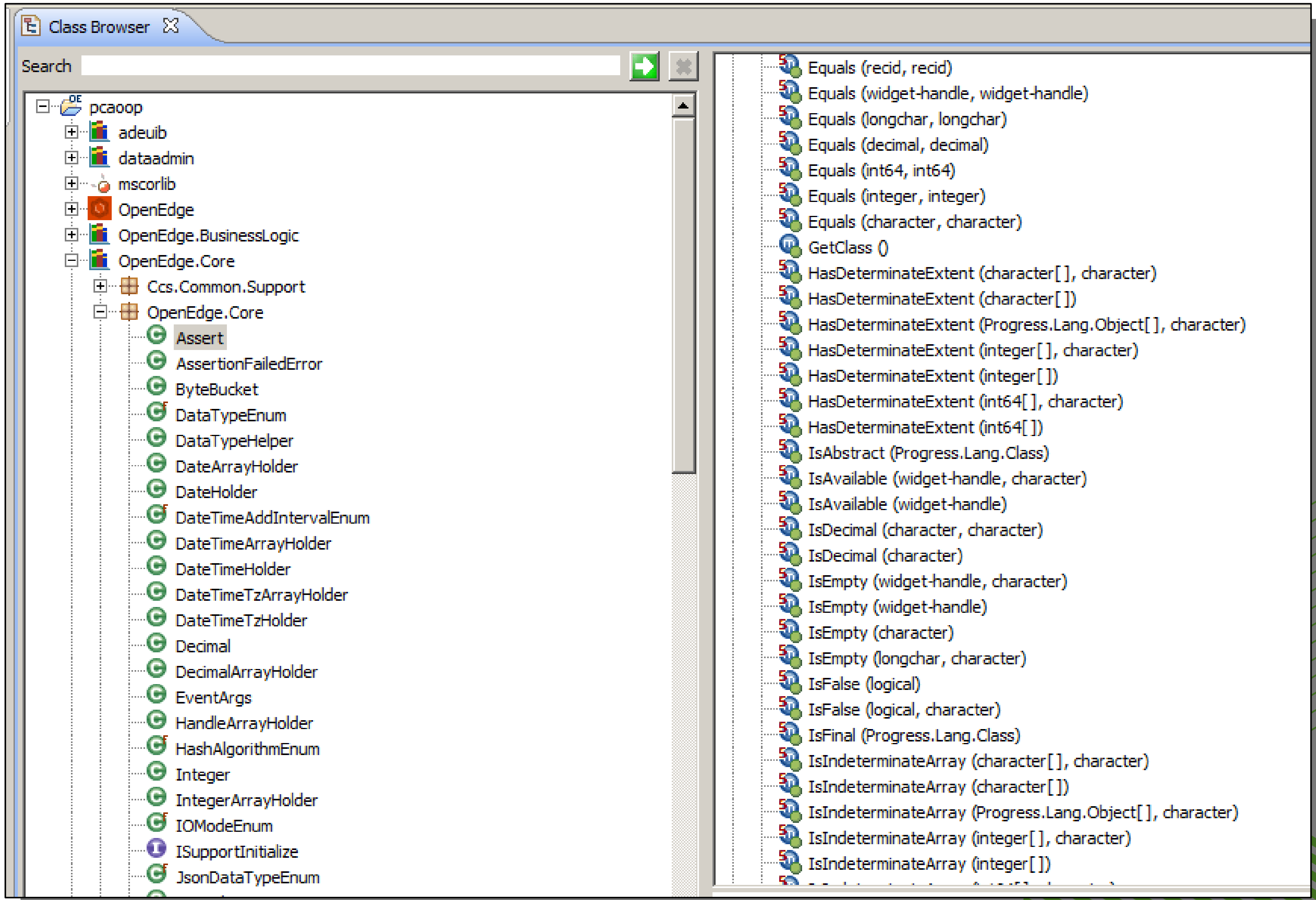
PUGCHALLENGE EXCHANGE AMERICAS

PUGCHALLENGE AMERICAS EXCHANGE

Every-day Use Examples

Every-day Use: OpenEdge.Core.Assert

- Easily the most versatile class
- Static methods
  - Covers many every-day use-cases



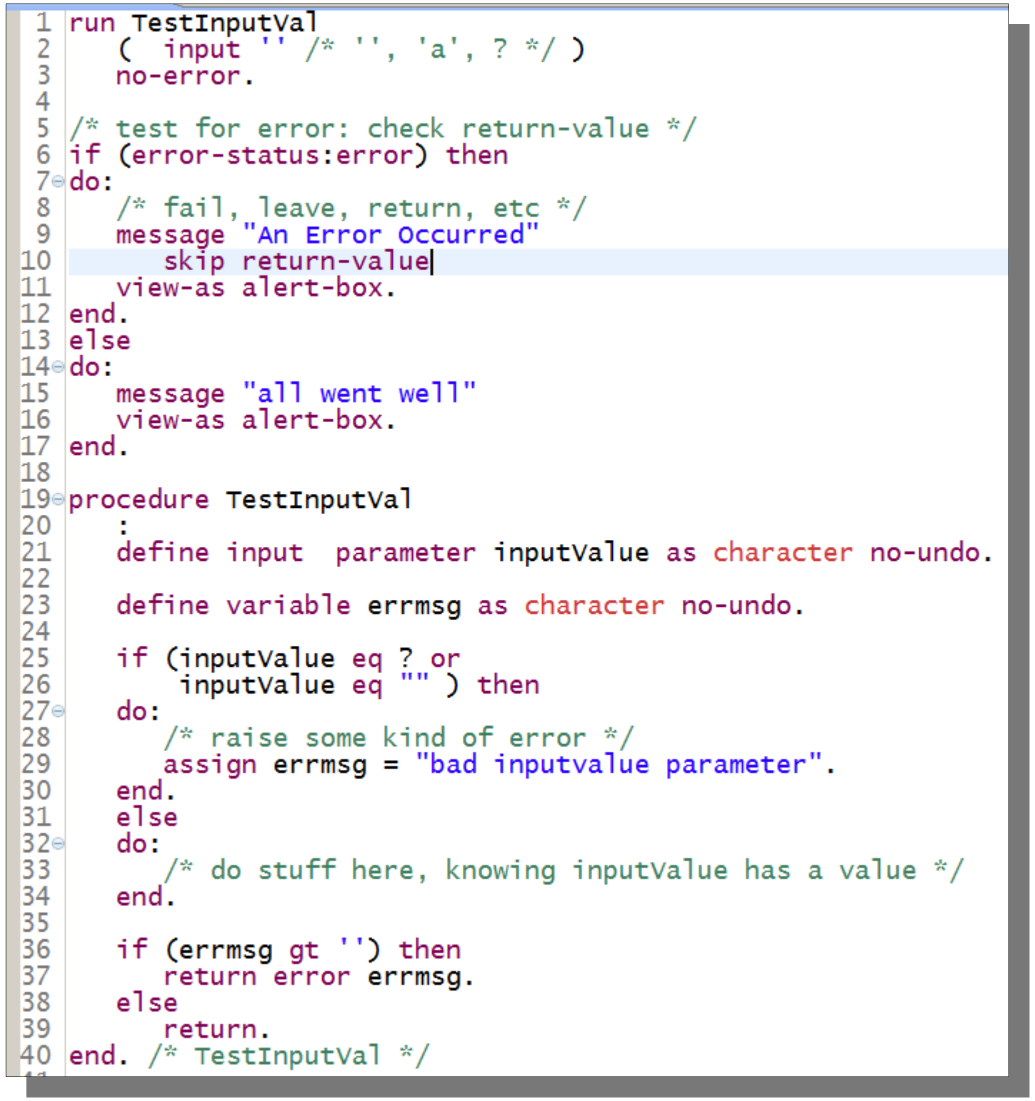
```
pcaoop
  adeuib
  dataadmin
  mscorlib
  OpenEdge
  OpenEdge.BusinessLogic
  OpenEdge.Core
    Ccs.Common.Support
    OpenEdge.Core
      Assert
      AssertionFailedError
      ByteBucket
      DataTypeEnum
      DataTypeHelper
      DateArrayHolder
      DateHolder
      DateTimeAddIntervalEnum
      DateTimeArrayHolder
      DateTimeHolder
      DateTimeTzArrayHolder
      DateTimeTzHolder
      Decimal
      DecimalArrayHolder
      EventArgs
      HandleArrayHolder
      HashAlgorithmEnum
      Integer
      IntegerArrayHolder
      IOModeEnum
      ISupportInitialize
      JsonDataTypeEnum
```

```
Equals (recid, recid)
Equals (widget-handle, widget-handle)
Equals (longchar, longchar)
Equals (decimal, decimal)
Equals (int64, int64)
Equals (integer, integer)
Equals (character, character)
GetClass ()
HasDeterminateExtent (character[ ], character)
HasDeterminateExtent (character[ ])
HasDeterminateExtent (Progress.Lang.Object[ ], character)
HasDeterminateExtent (integer[ ], character)
HasDeterminateExtent (integer[ ])
HasDeterminateExtent (int64[ ], character)
HasDeterminateExtent (int64[ ])
IsAbstract (Progress.Lang.Class)
IsAvailable (widget-handle, character)
IsAvailable (widget-handle)
IsDecimal (character, character)
IsDecimal (character)
IsEmpty (widget-handle, character)
IsEmpty (widget-handle)
IsEmpty (character)
IsEmpty (longchar, character)
IsFalse (logical)
IsFalse (logical, character)
IsFinal (Progress.Lang.Class)
IsIndeterminateArray (character[ ], character)
IsIndeterminateArray (character[ ])
IsIndeterminateArray (Progress.Lang.Object[ ], character)
IsIndeterminateArray (integer[ ], character)
IsIndeterminateArray (integer[ ])
```

PUGCHALLENGE EXCHANGE AMERICAS

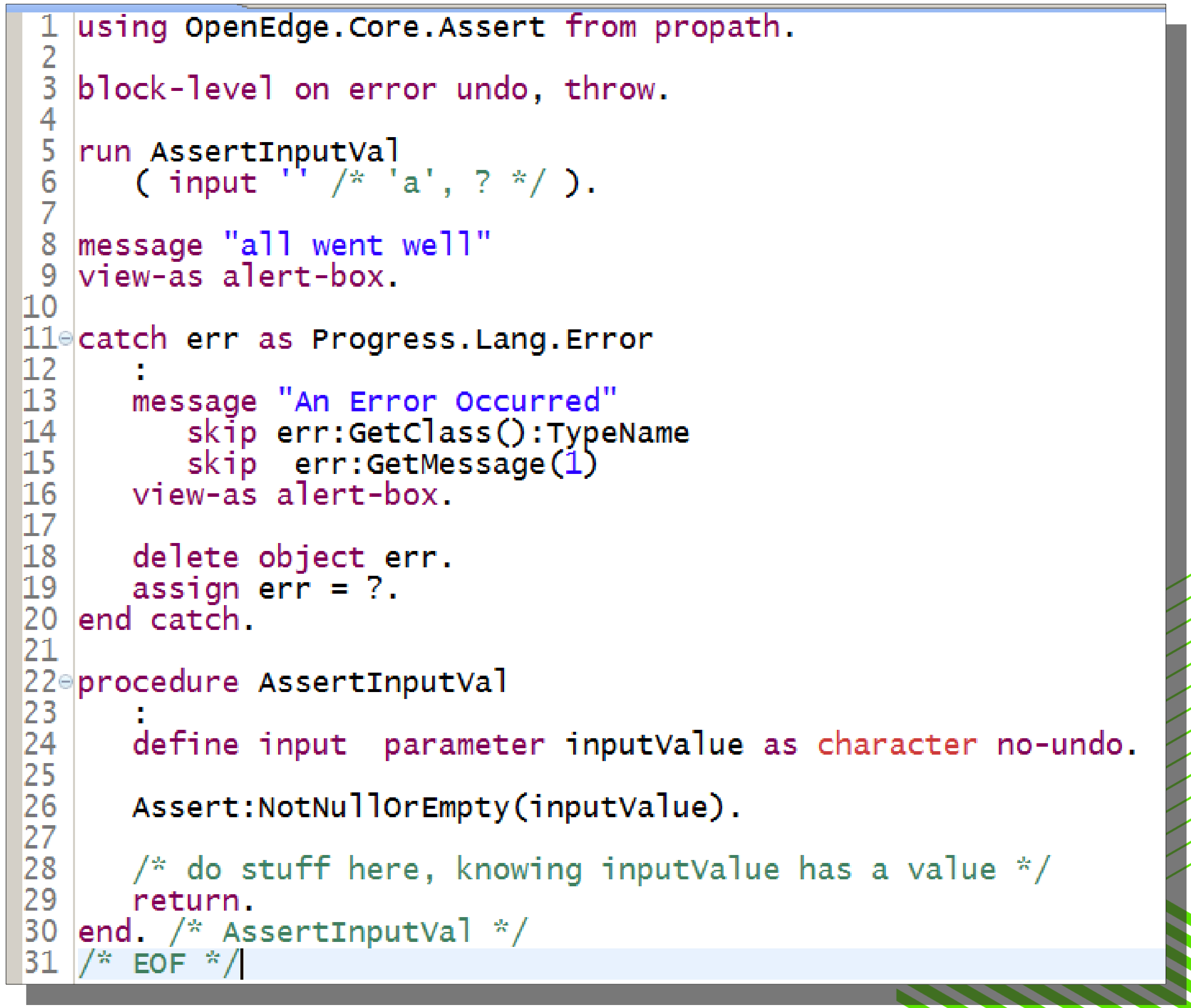
Every-day Use: OpenEdge.Core.Assert

Procedural Validation



```
run TestInputVal
   (  input '' /* '', 'a', ? */ )
   no-error.

/* test for error: check return-value */
if (error-status:error) then
do:
   /* fail, leave, return, etc */
   message "An Error Occurred"
      skip return-value
   view-as alert-box.
end.
else
do:
   message "all went well"
   view-as alert-box.
end.

procedure TestInputVal
   :
   define input  parameter inputValue as character no-undo.

   define variable errmsg as character no-undo.

   if (inputValue eq ? or
       inputValue eq "" ) then
   do:
      /* raise some kind of error */
      assign errmsg = "bad inputvalue parameter".
   end.
   else
   do:
      /* do stuff here, knowing inputValue has a value */
   end.

   if (errmsg gt '') then
      return error errmsg.
   else
      return.
end. /* TestInputVal */
```

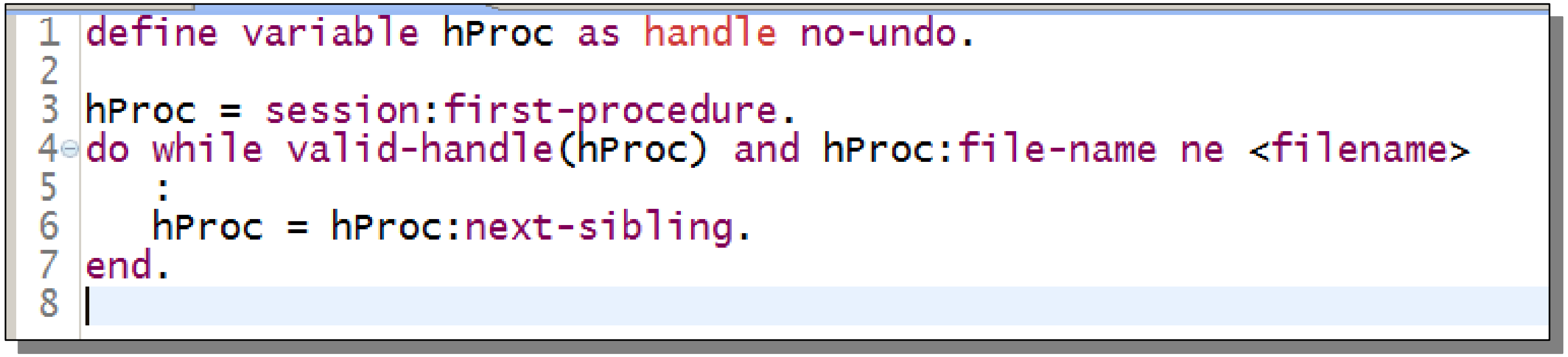
OOABL Assertions



```
using OpenEdge.Core.Assert from propath.

block-level on error undo, throw.

run AssertInputVal
   ( input '' /* 'a', ? */ ).

message "all went well"
view-as alert-box.

catch err as Progress.Lang.Error
   :
   message "An Error Occurred"
      skip err:GetClass():TypeName
      skip  err:GetMessage(1)
   view-as alert-box.

   delete object err.
   assign err = ?.
end catch.

procedure AssertInputVal
   :
   define input  parameter inputValue as character no-undo.

   Assert:NotNullOrEmpty(inputValue).

   /* do stuff here, knowing inputValue has a value */
   return.
end. /* AssertInputVal */
/* EOF */
```

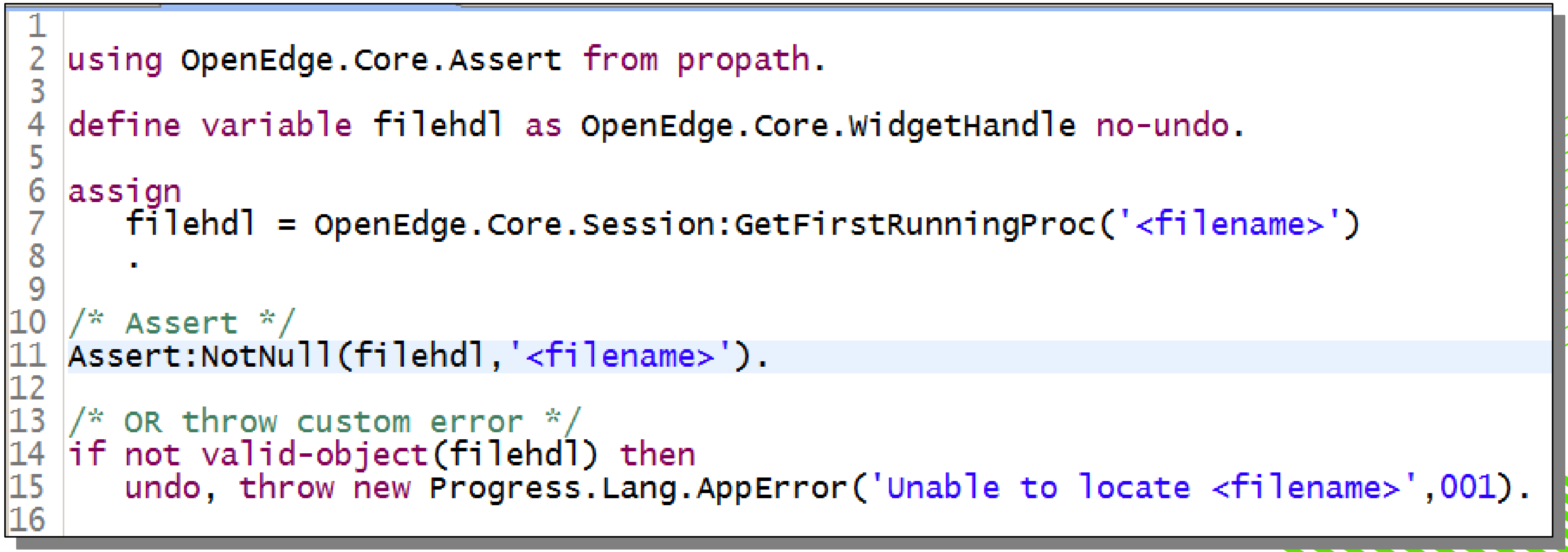
Every-day Use: OpenEdge.Core.Session

Procedural Session



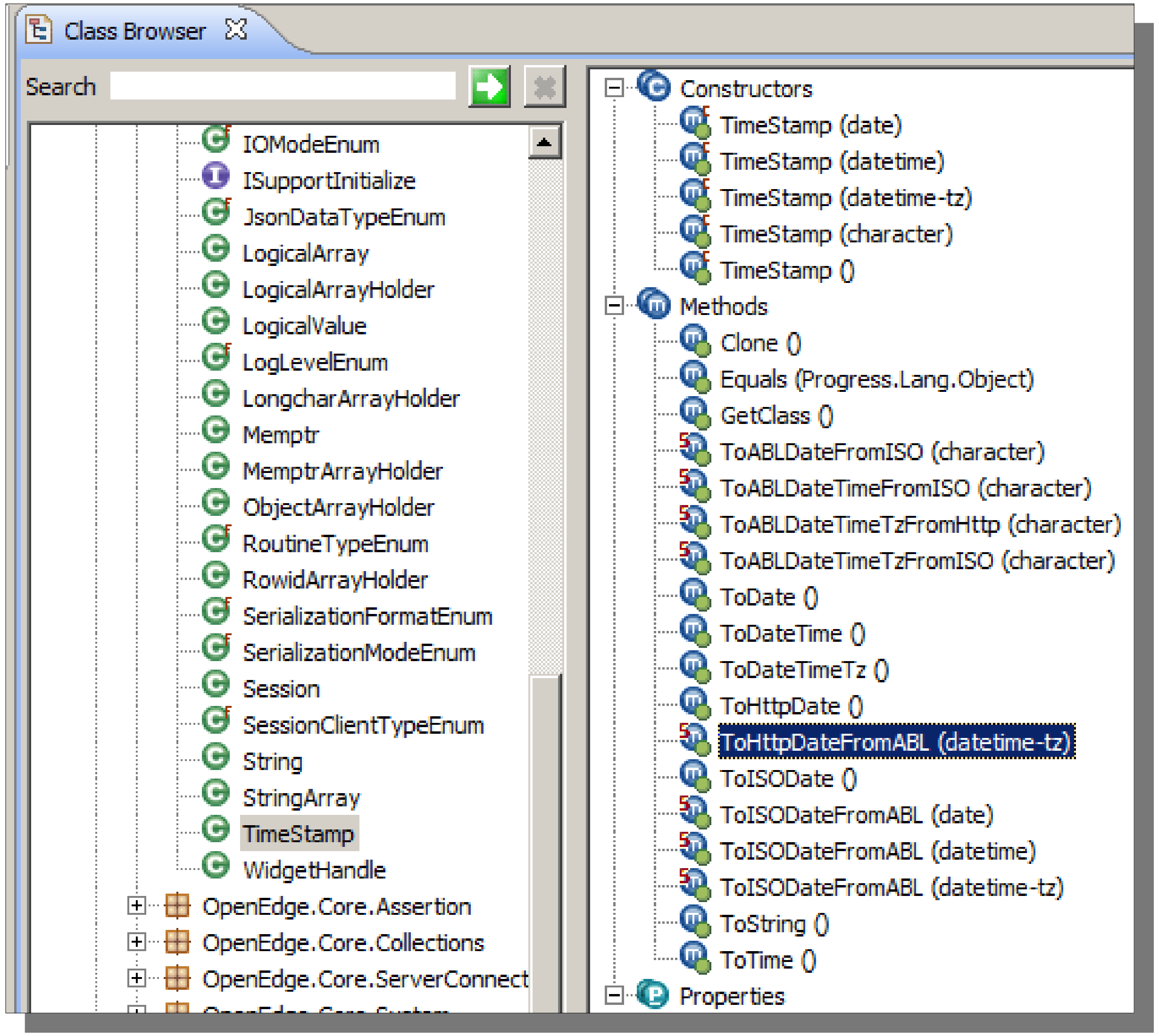
```
define variable hProc as handle no-undo.

hProc = session:first-procedure.
do while valid-handle(hProc) and hProc:file-name ne <filename>
   :
   hProc = hProc:next-sibling.
end.

```

OOABL Session



```

using OpenEdge.Core.Assert from propath.

define variable filehdl as OpenEdge.Core.WidgetHandle no-undo.

assign
   filehdl = OpenEdge.Core.Session:GetFirstRunningProc('<filename>')
   .

/* Assert */
Assert:NotNull(filehdl,'<filename>').

/* OR throw custom error */
if not valid-object(filehdl) then
   undo, throw new Progress.Lang.AppError('Unable to locate <filename>',001).

```

Every-day Use: OpenEdge.Core.TimeStamp


Class Browser

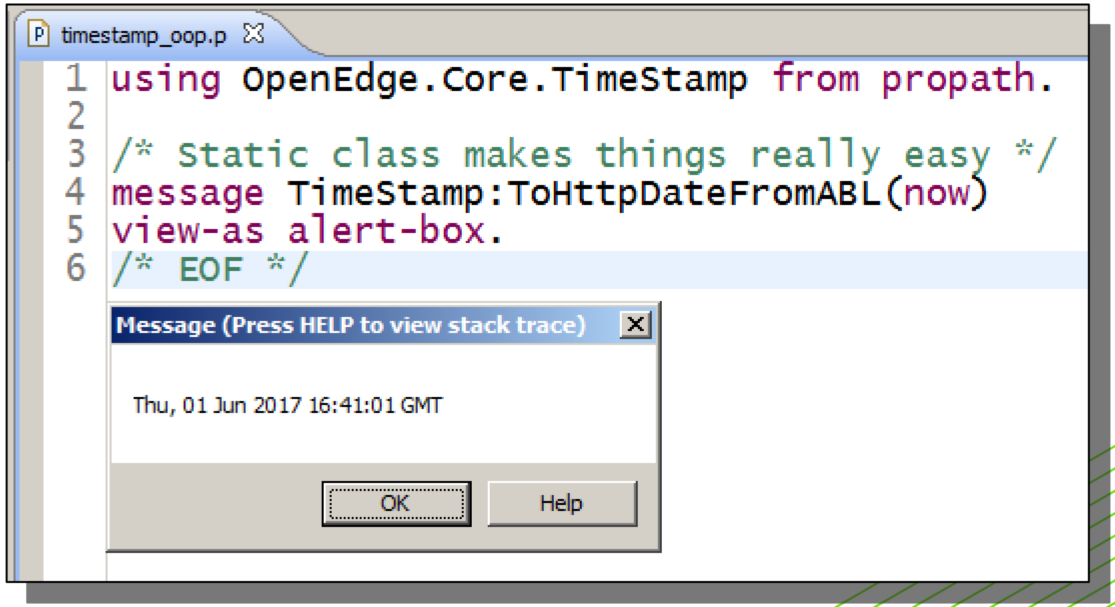
Search

- IOModeEnum
- ISupportInitialize
- JsonDataTypeEnum
- LogicalArray
- LogicalArrayHolder
- LogicalValue
- LogLevelEnum
- LongcharArrayHolder
- Memptr
- MemptrArrayHolder
- ObjectArrayHolder
- RoutineTypeEnum
- RowidArrayHolder
- SerializationFormatEnum
- SerializationModeEnum
- Session
- SessionClientTypeEnum
- String
- StringArray
- TimeStamp
- WidgetHandle
- OpenEdge.Core.Assertion
- OpenEdge.Core.Collections
- OpenEdge.Core.ServerConnect

Constructors
- TimeStamp (date)
- TimeStamp (datetime)
- TimeStamp (datetime-tz)
- TimeStamp (character)
- TimeStamp ()

Methods
- Clone ()
- Equals (Progress.Lang.Object)
- GetClass ()
- ToABLDateFromISO (character)
- ToABLDateTimeFromISO (character)
- ToABLDateTimeTzFromHttp (character)
- ToABLDateTimeTzFromISO (character)
- ToDate ()
- ToDateTime ()
- ToDateTimeTz ()
- ToHttpDate ()
- ToHttpDateFromABL (datetime-tz)
- ToISODate ()
- ToISODateFromABL (date)
- ToISODateFromABL (datetime)
- ToISODateFromABL (datetime-tz)
- ToString ()
- ToTime ()

Properties


timestamp_oop.p

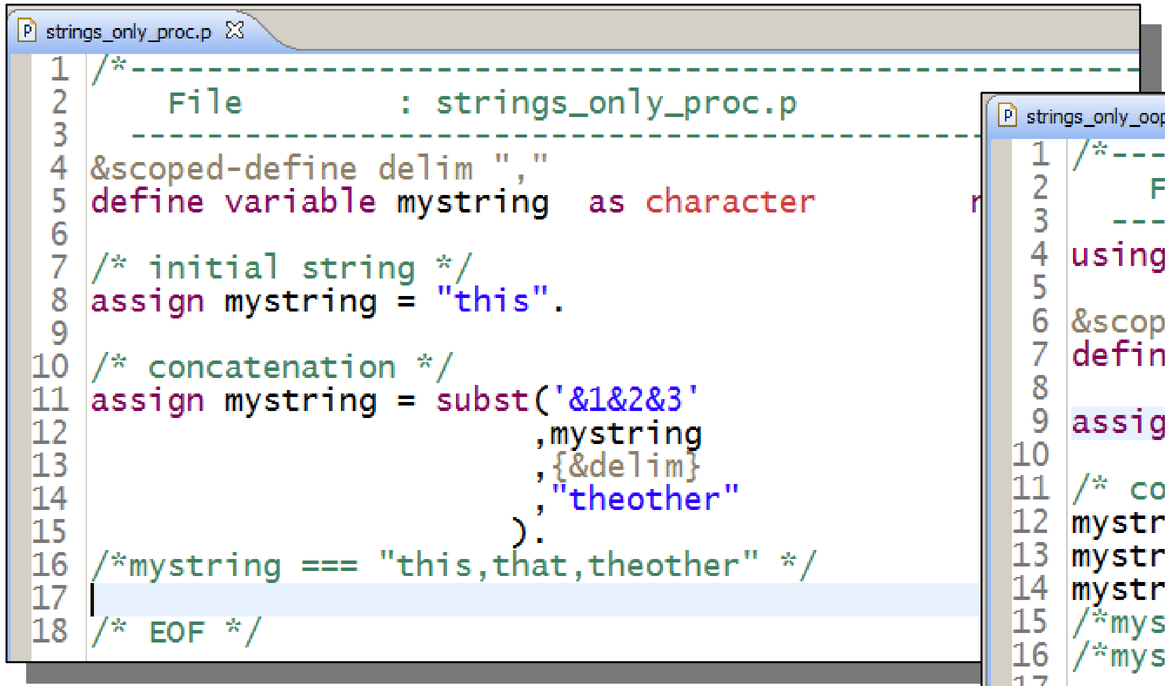
```
using OpenEdge.Core.TimeStamp from propath.

/* Static class makes things really easy */
message TimeStamp:ToHttpDateFromABL(now)
view-as alert-box.
/* EOF */
```

Message (Press HELP to view stack trace)

```
Thu, 01 Jun 2017 16:41:01 GMT
```

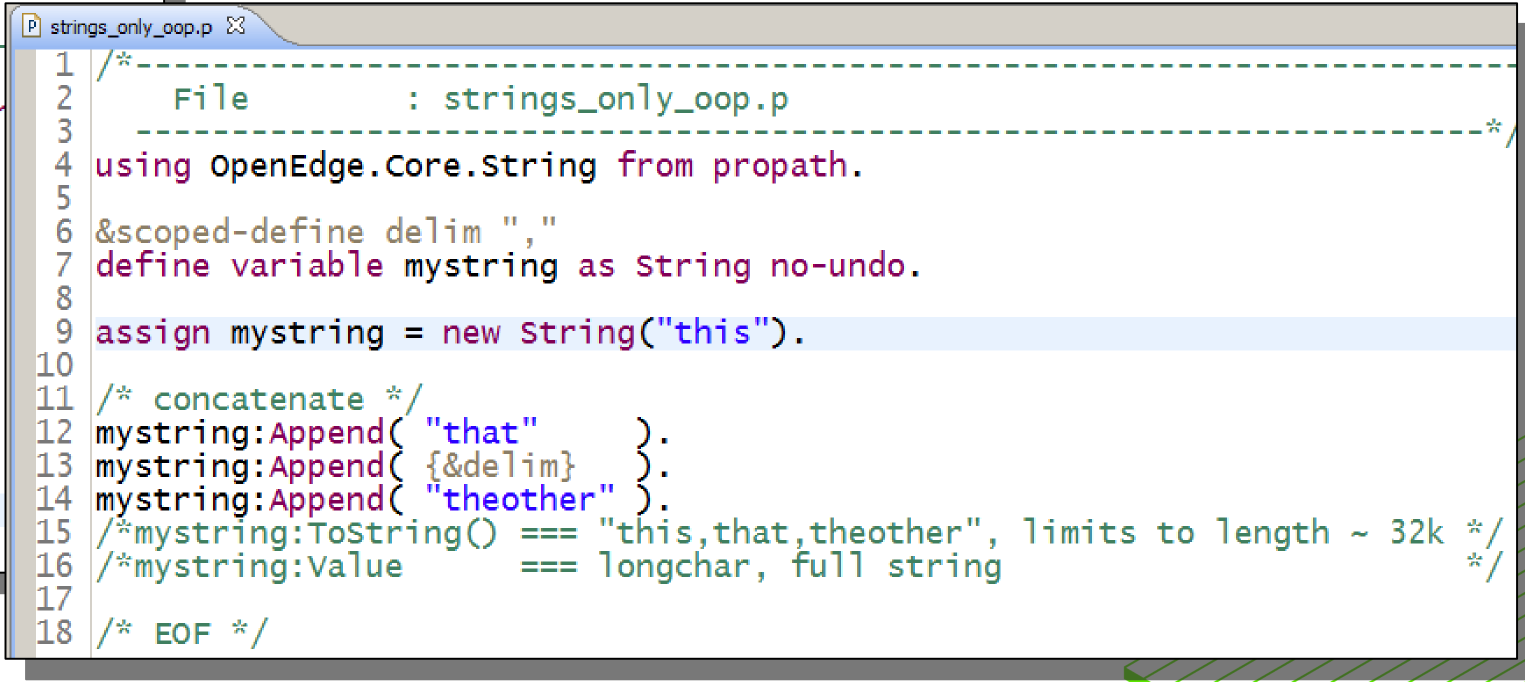
Every-day Use: OpenEdge.Core.String

Procedural Strings



```
/*------------------------------------------------------------------
    File        : strings_only_proc.p
  ----------------------------------------------------------------
&scoped-define delim ","
define variable mystring  as character          r

/* initial string */
assign mystring = "this".

/* concatenation */
assign mystring = subst('&1&2&3'
                       ,mystring
                       ,{&delim}
                       ,"theother"
                      ).
/*mystring === "this,that,theother" */

/* EOF */
```

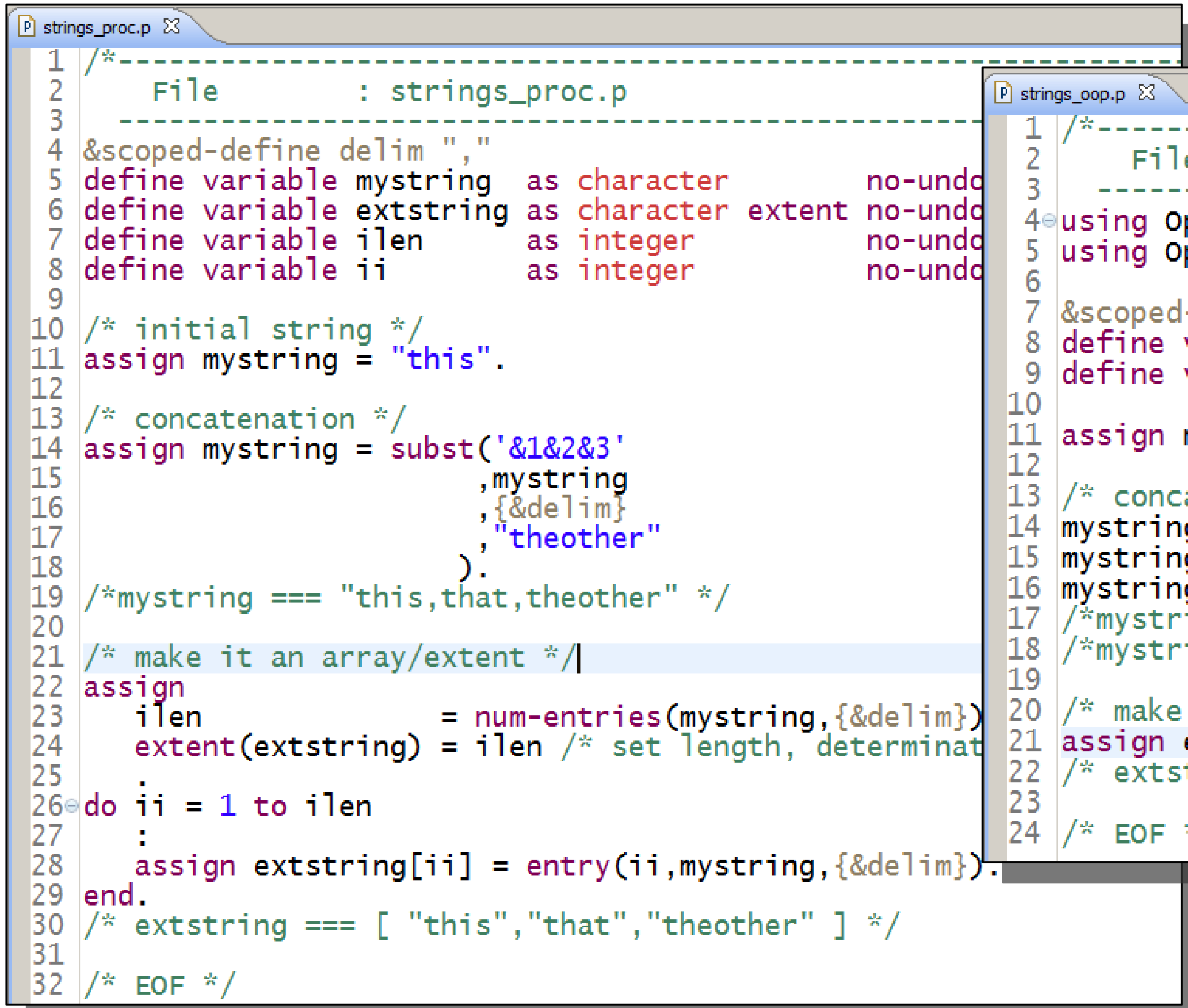
OOABL Strings



```
/*----------------------------------------------------------------------
    File        : strings_only_oop.p
  ---------------------------------------------------------------------*/
using OpenEdge.Core.String from propath.

&scoped-define delim ","
define variable mystring as String no-undo.

assign mystring = new String("this").

/* concatenate */
mystring:Append( "that"      ).
mystring:Append( {&delim}    ).
mystring:Append( "theother" ).
/*mystring:ToString() === "this,that,theother", limits to length ~ 32k */
/*mystring:Value      === longchar, full string                        */

/* EOF */
```

Every-day Use: OpenEdge.Core.Collections.Array

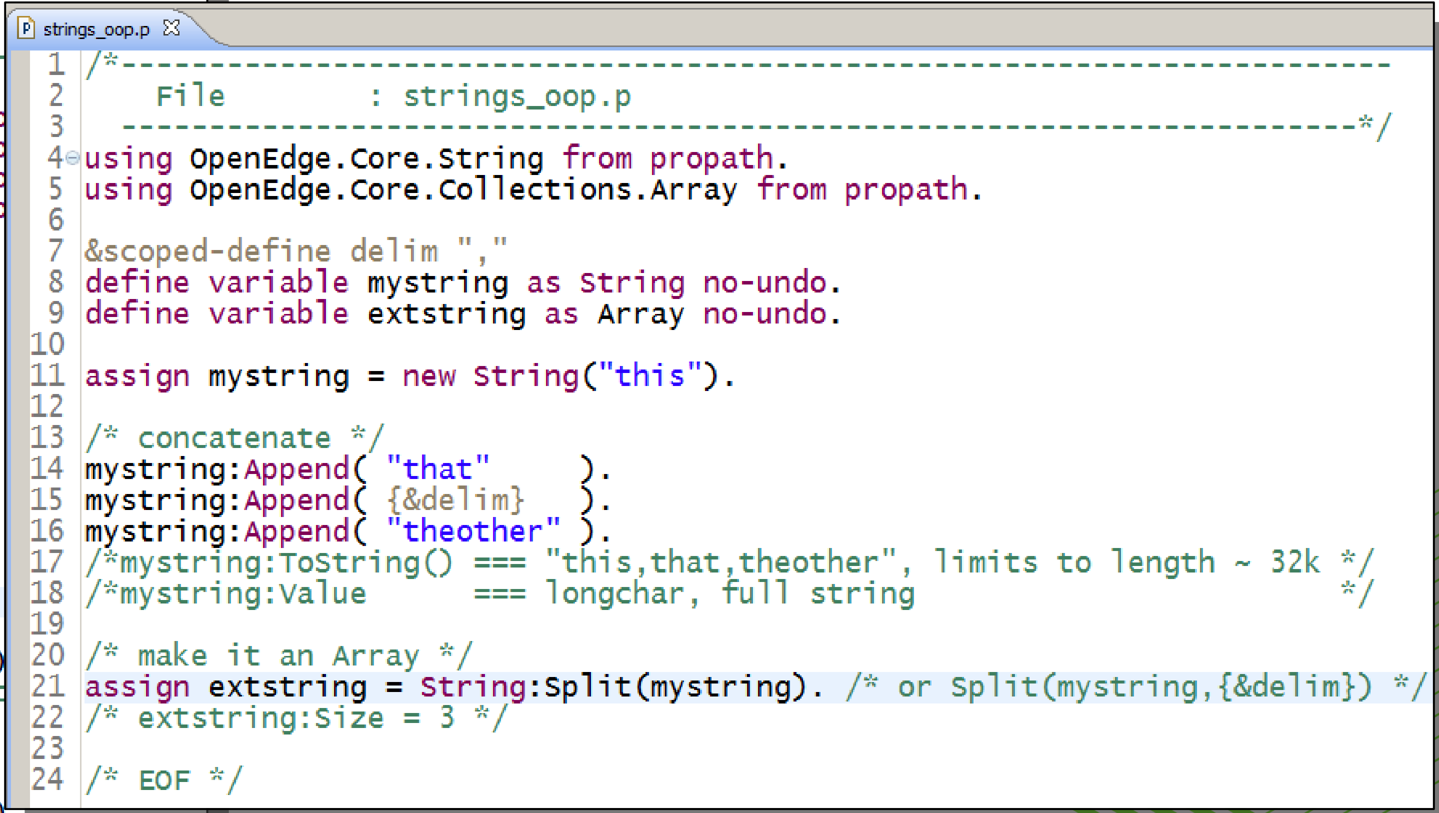
Procedural Array/Extent (variable len)



strings_proc.p

```
/*-----------------------------------------------------------------
    File        : strings_proc.p
  ----------------------------------------------------
&scoped-define delim ","
define variable mystring  as character        no-und
define variable extstring as character extent no-und
define variable ilen      as integer          no-und
define variable ii        as integer          no-und

/* initial string */
assign mystring = "this".

/* concatenation */
assign mystring = subst('&1&2&3'
                        ,mystring
                        ,{&delim}
                        ,"theother"
                       ).
/*mystring === "this,that,theother" */

/* make it an array/extent */
assign
   ilen              = num-entries(mystring,{&delim})
   extent(extstring) = ilen /* set length, determinat
   .
do ii = 1 to ilen
   :
   assign extstring[ii] = entry(ii,mystring,{&delim}).
end.
/* extstring === [ "this","that","theother" ] */

/* EOF */
```

OOABL Array

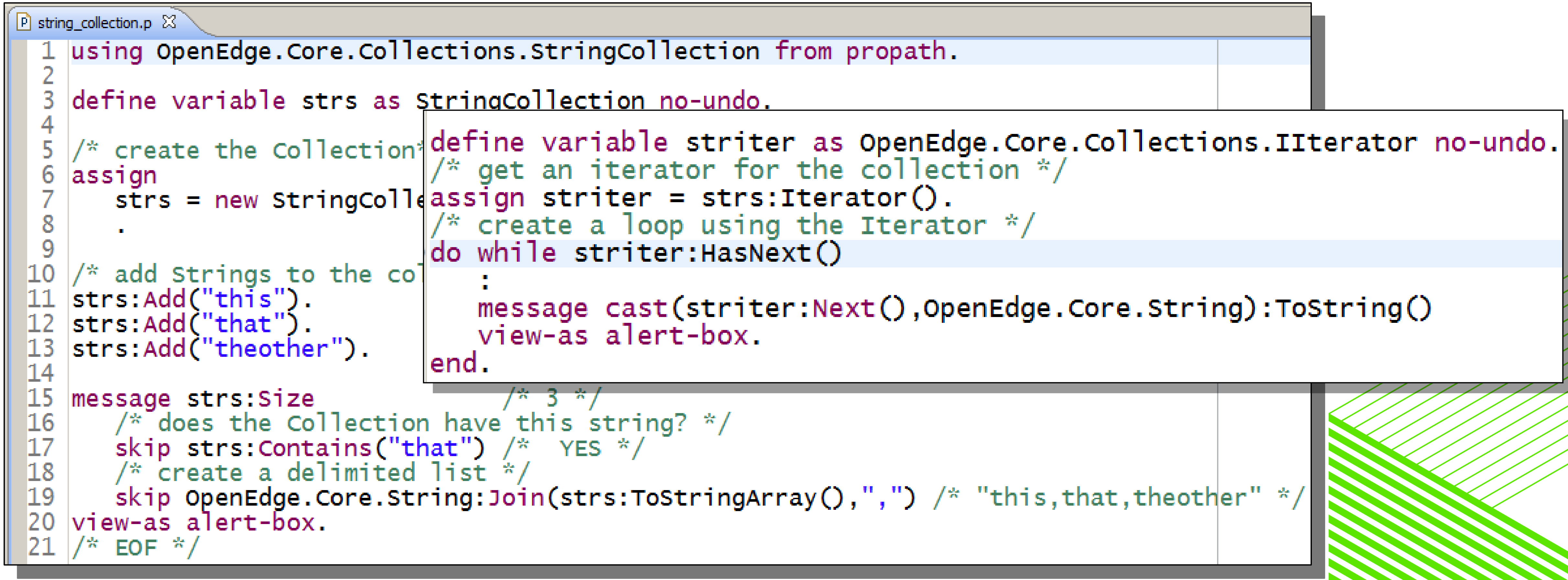


strings_oop.p

```
/*--------------------------------------------------------------------------
    File        : strings_oop.p
  ------------------------------------------------------------------------*/
using OpenEdge.Core.String from propath.
using OpenEdge.Core.Collections.Array from propath.

&scoped-define delim ","
define variable mystring as String no-undo.
define variable extstring as Array no-undo.

assign mystring = new String("this").

/* concatenate */
mystring:Append( "that"      ).
mystring:Append( {&delim}    ).
mystring:Append( "theother" ).
/*mystring:ToString() === "this,that,theother", limits to length ~ 32k */
/*mystring:Value      === longchar, full string                       */

/* make it an Array */
assign extstring = String:Split(mystring). /* or Split(mystring,{&delim}) */
/* extstring:Size = 3 */

/* EOF */
```

Every-day Use: OpenEdge.Core.Collections.StringCollection

- A Better Way to build a list of Strings
- Consider it a temp-table of typed-objects
- Use Iterator to loop entries



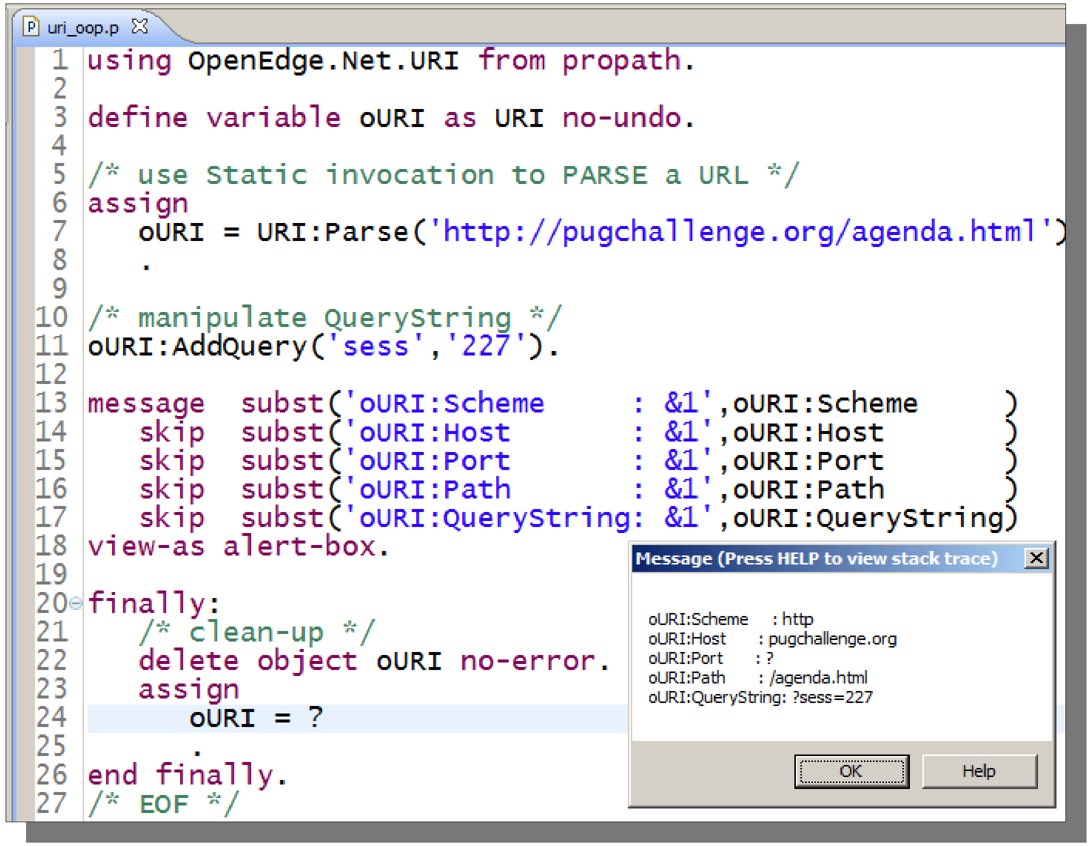
```
using OpenEdge.Core.Collections.StringCollection from propath.

define variable strs as StringCollection no-undo.

/* create the Collection
assign
   strs = new StringColle
   .

/* add Strings to the co
strs:Add("this").
strs:Add("that").
strs:Add("theother").

message strs:Size          /* 3 */
   /* does the Collection have this string? */
   skip strs:Contains("that") /*  YES */
   /* create a delimited list */
   skip OpenEdge.Core.String:Join(strs:ToStringArray(),",") /* "this,that,theother" */
view-as alert-box.
/* EOF */
```

```
define variable striter as OpenEdge.Core.Collections.IIterator no-undo.
/* get an iterator for the collection */
assign striter = strs:Iterator().
/* create a loop using the Iterator */
do while striter:HasNext()
   :
   message cast(striter:Next(),OpenEdge.Core.String):ToString()
   view-as alert-box.
end.
```

PUGCHALLENGE EXCHANGE AMERICAS

Every-day Use: OpenEdge.Net.URI

- String-parsing of URIs can be hit-or-miss
- URI has both Static and Instance methods



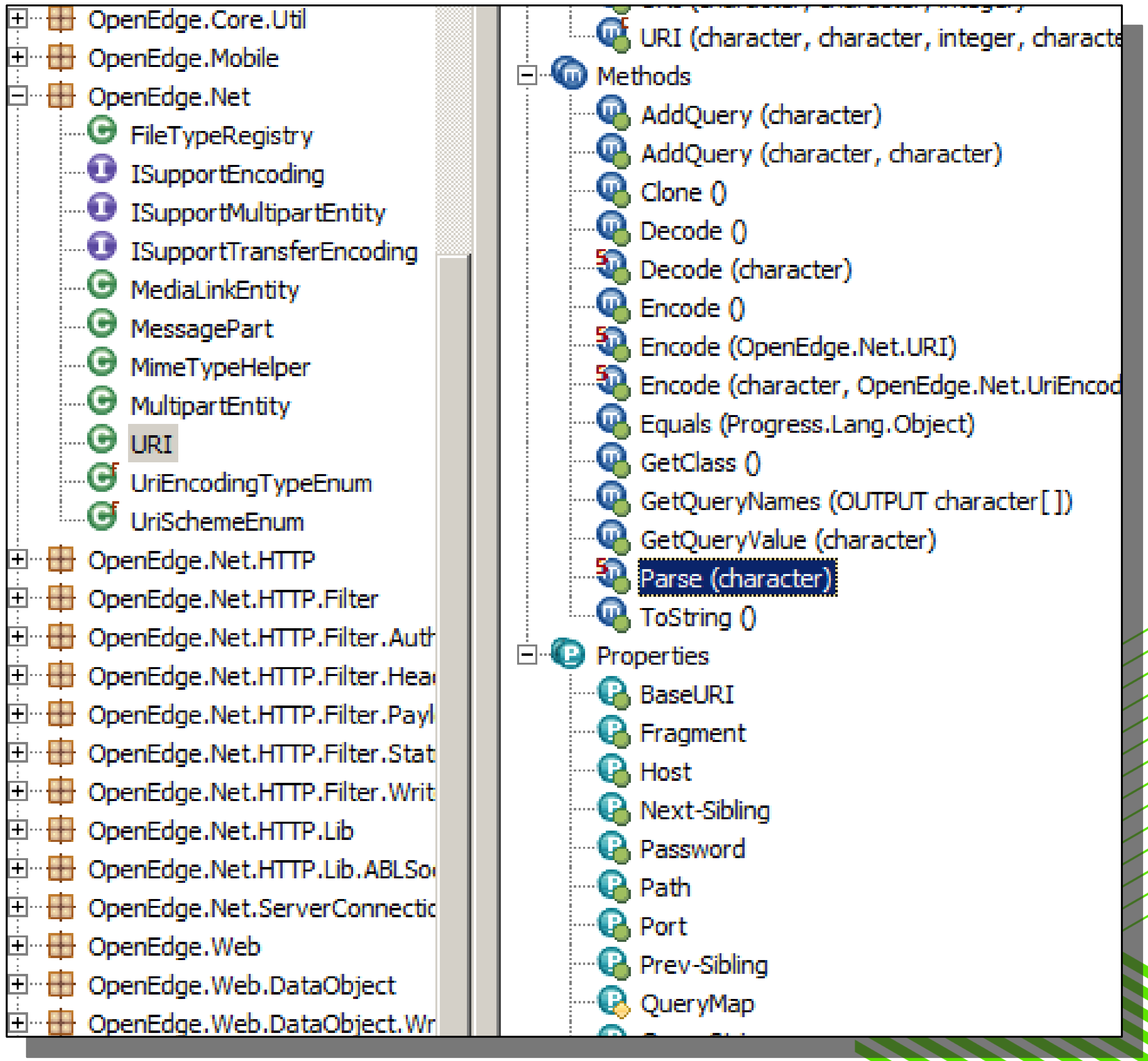
uri_oop.p

```
using OpenEdge.Net.URI from propath.

define variable oURI as URI no-undo.

/* use Static invocation to PARSE a URL */
assign
   oURI = URI:Parse('http://pugchallenge.org/agenda.html')
   .

/* manipulate QueryString */
oURI:AddQuery('sess','227').

message  subst('oURI:Scheme      : &1',oURI:Scheme      )
   skip  subst('oURI:Host        : &1',oURI:Host        )
   skip  subst('oURI:Port        : &1',oURI:Port        )
   skip  subst('oURI:Path        : &1',oURI:Path        )
   skip  subst('oURI:QueryString: &1',oURI:QueryString)
view-as alert-box.

finally:
   /* clean-up */
   delete object oURI no-error.
   assign
      oURI = ?
      .
end finally.
/* EOF */
```

Message (Press HELP to view stack trace)

```
oURI:Scheme     : http
oURI:Host       : pugchallenge.org
oURI:Port       : ?
oURI:Path       : /agenda.html
oURI:QueryString: ?sess=227
```

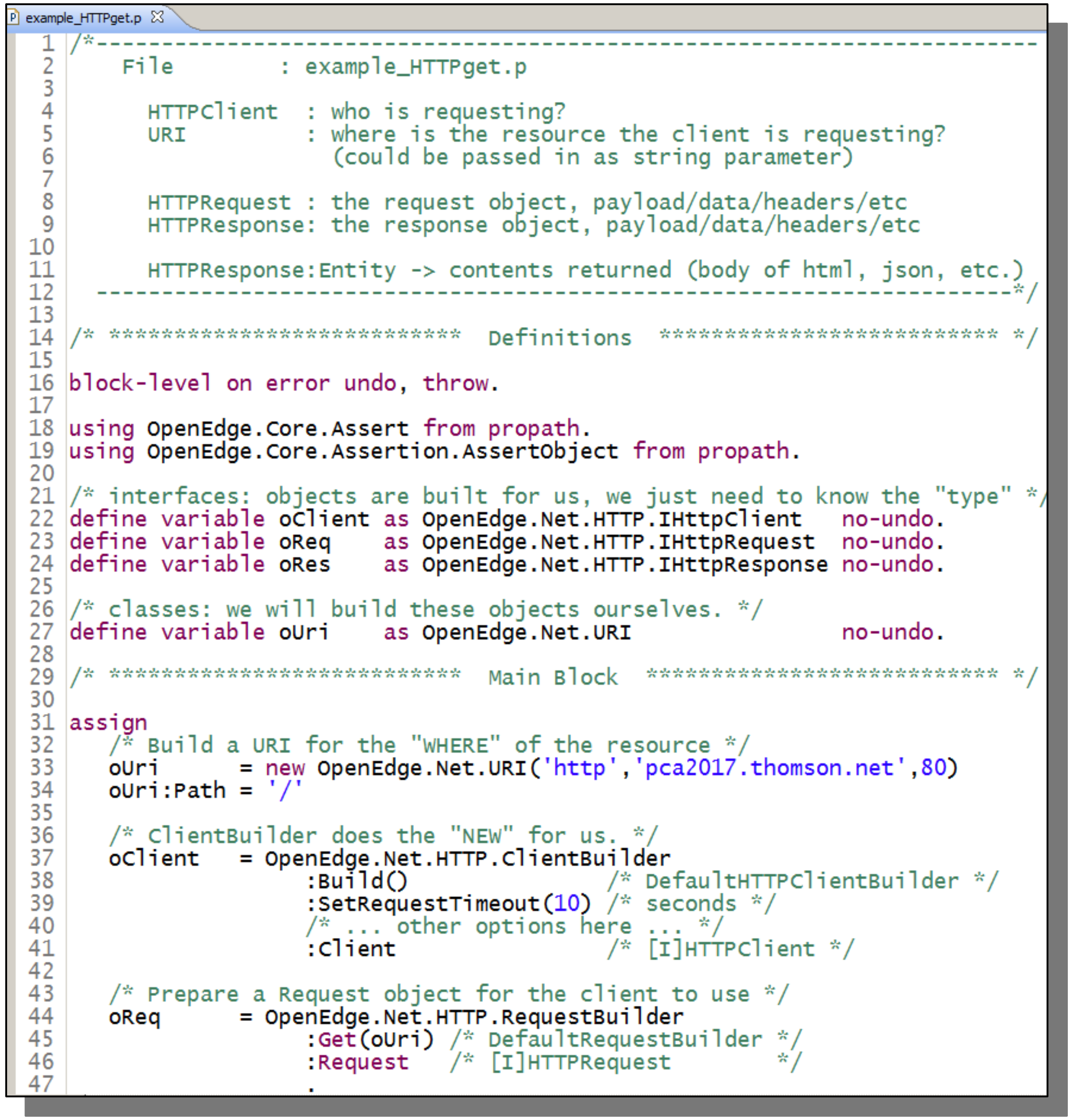
OK   Help



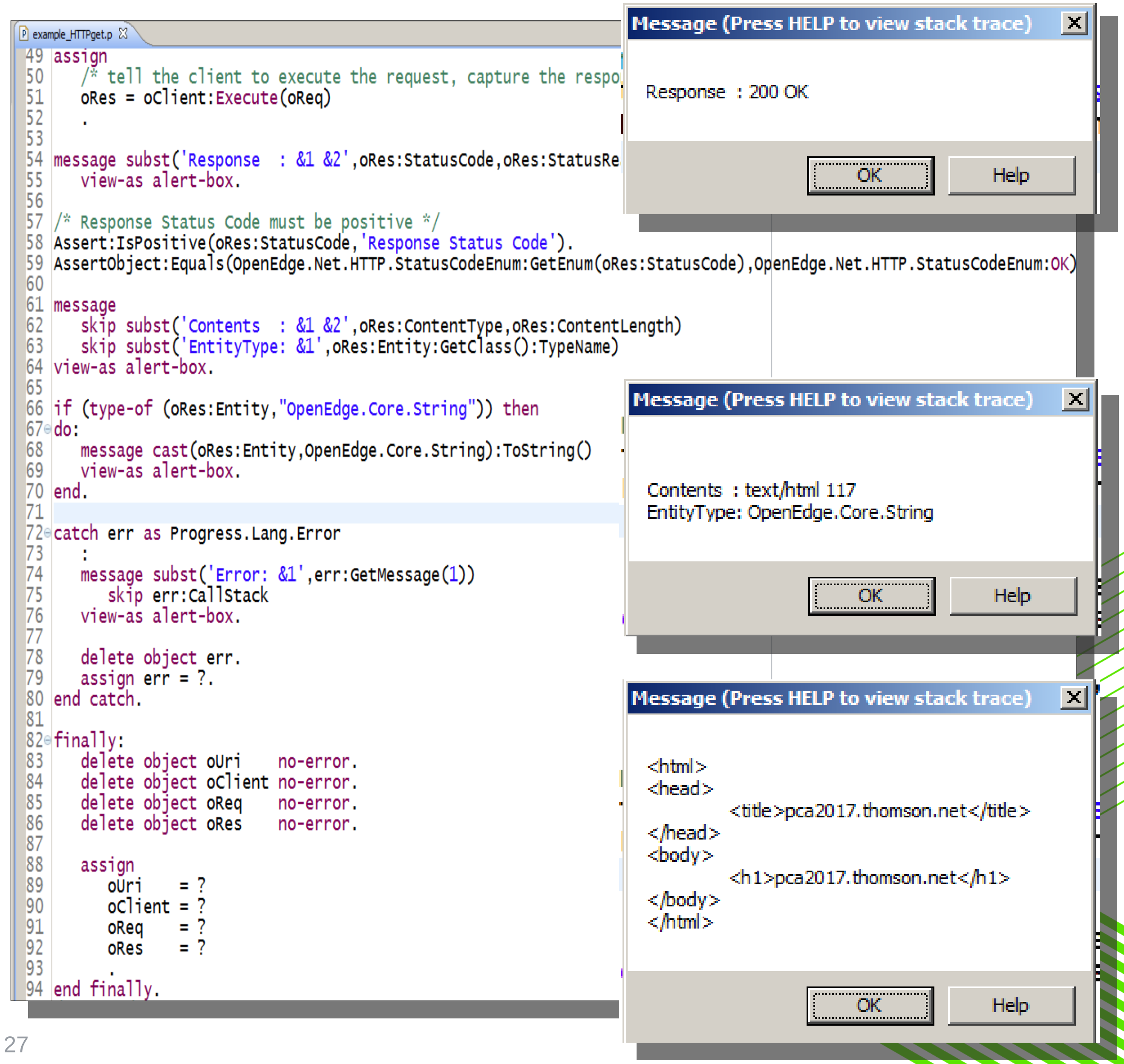
```
+ OpenEdge.Core.Util
+ OpenEdge.Mobile
- OpenEdge.Net
    FileTypeRegistry
    ISupportEncoding
    ISupportMultipartEntity
    ISupportTransferEncoding
    MediaLinkEntity
    MessagePart
    MimeTypeHelper
    MultipartEntity
    URI
    UriEncodingTypeEnum
    UriSchemeEnum
+ OpenEdge.Net.HTTP
+ OpenEdge.Net.HTTP.Filter
+ OpenEdge.Net.HTTP.Filter.Auth
+ OpenEdge.Net.HTTP.Filter.Head
+ OpenEdge.Net.HTTP.Filter.Payl
+ OpenEdge.Net.HTTP.Filter.Stat
+ OpenEdge.Net.HTTP.Filter.Writ
+ OpenEdge.Net.HTTP.Lib
+ OpenEdge.Net.HTTP.Lib.ABLSo
+ OpenEdge.Net.ServerConnectic
+ OpenEdge.Web
+ OpenEdge.Web.DataObject
+ OpenEdge.Web.DataObject.Wr
```

```
    URI (character, character, integer, characte
- Methods
    AddQuery (character)
    AddQuery (character, character)
    Clone ()
    Decode ()
    Decode (character)
    Encode ()
    Encode (OpenEdge.Net.URI)
    Encode (character, OpenEdge.Net.UriEncod
    Equals (Progress.Lang.Object)
    GetClass ()
    GetQueryNames (OUTPUT character[])
    GetQueryValue (character)
    Parse (character)
    ToString ()
- Properties
    BaseURI
    Fragment
    Host
    Next-Sibling
    Password
    Path
    Port
    Prev-Sibling
    QueryMap
```

Every-day Use: OpenEdge.Net.HTTP.*


example_HTTPget.p

```
/*--------------------------------------------------------------------------------
   File         : example_HTTPget.p

     HTTPClient  : who is requesting?
     URI         : where is the resource the client is requesting?
                   (could be passed in as string parameter)

     HTTPRequest : the request object, payload/data/headers/etc
     HTTPResponse: the response object, payload/data/headers/etc

     HTTPResponse:Entity -> contents returned (body of html, json, etc.)
  ------------------------------------------------------------------------------*/

/* ***************************  Definitions  ************************** */

block-level on error undo, throw.

using OpenEdge.Core.Assert from propath.
using OpenEdge.Core.Assertion.AssertObject from propath.

/* interfaces: objects are built for us, we just need to know the "type" */
define variable oClient as OpenEdge.Net.HTTP.IHttpClient   no-undo.
define variable oReq    as OpenEdge.Net.HTTP.IHttpRequest  no-undo.
define variable oRes    as OpenEdge.Net.HTTP.IHttpResponse no-undo.

/* classes: we will build these objects ourselves. */
define variable oUri    as OpenEdge.Net.URI                no-undo.

/* ***************************  Main Block  *************************** */

assign
   /* Build a URI for the "WHERE" of the resource */
   oUri      = new OpenEdge.Net.URI('http','pca2017.thomson.net',80)
   oUri:Path = '/'

   /* ClientBuilder does the "NEW" for us. */
   oClient   = OpenEdge.Net.HTTP.ClientBuilder
                  :Build()                 /* DefaultHTTPClientBuilder */
                  :SetRequestTimeout(10) /* seconds */
                  /* ... other options here ... */
                  :Client                  /* [I]HTTPClient */

   /* Prepare a Request object for the client to use */
   oReq      = OpenEdge.Net.HTTP.RequestBuilder
                  :Get(oUri) /* DefaultRequestBuilder */
                  :Request   /* [I]HTTPRequest        */
                  .
```


example_HTTPget.p

```
assign
   /* tell the client to execute the request, capture the respo
   oResp = oClient:Execute(oReq)
   .

message subst('Response  : &1 &2',oRes:StatusCode,oRes:StatusRe
   view-as alert-box.

/* Response Status Code must be positive */
Assert:IsPositive(oRes:StatusCode,'Response Status Code').
AssertObject:Equals(OpenEdge.Net.HTTP.StatusCodeEnum:GetEnum(oRes:StatusCode),OpenEdge.Net.HTTP.StatusCodeEnum:OK)

message
   skip subst('Contents  : &1 &2',oRes:ContentType,oRes:ContentLength)
   skip subst('EntityType: &1',oRes:Entity:GetClass():TypeName)
view-as alert-box.

if (type-of (oRes:Entity,"OpenEdge.Core.String")) then
do:
   message cast(oRes:Entity,OpenEdge.Core.String):ToString()
   view-as alert-box.
end.

catch err as Progress.Lang.Error
   :
   message subst('Error: &1',err:GetMessage(1))
      skip err:CallStack
   view-as alert-box.

   delete object err.
   assign err = ?.
end catch.

finally:
   delete object oUri    no-error.
   delete object oClient no-error.
   delete object oReq    no-error.
   delete object oRes    no-error.

   assign
      oUri    = ?
      oClient = ?
      oReq    = ?
      oRes    = ?
      .
end finally.
```

Message (Press HELP to view stack trace)

```
Response  : 200 OK
```

Message (Press HELP to view stack trace)

```
Contents  : text/html 117
EntityType: OpenEdge.Core.String
```

Message (Press HELP to view stack trace)

```
<html>
<head>
        <title>pca2017.thomson.net</title>
</head>
<body>
        <h1>pca2017.thomson.net</h1>
</body>
</html>
```

PUGCHALLENGE AMERICAS EXCHANGE

What to Expect

What to Expect

- Adopt **PDSOE** for your IDE
  - The auto-complete and class-viewer are invaluable
  - At minimum, use an IDE that performs color-coding and syntax checks
- Confusion – Move at a comfortable pace for you
  - Start simple
  - Move deeper into OOP as your understanding increases
  - Learn using good examples
- Reference libraries **will change** between versions
  - Acquire the source and actually look at it
  - Compare the source between versions
    - Compilations will let you know if things are *broken*, you will want to confirm the behavior hasn't changed beyond your expectations

Next Steps

- Create your own custom override classes
  - Explore use of overloads to simplify call signature
- Interfaces, Abstract classes
- JSON
  - Progress.JSON.*
- Enumerators
  - Progress.Lang.Enum
  - OpenEdge.Core.*Enum
- Runtime Application Framework
  - CCS-style

Next Steps

- Next-level OOP design recommendations
  - CCS Samples: https://github.com/consultingwerk/CCS_Samples
  - IOC, Dependency Injection: https://github.com/PeterJudge-PSC/InjectABL
  - AutoEdge, The Factory: https://github.com/PeterJudge-PSC/autoedgethefactory
- (2016) 806: OO-Oh, Mike Fechner, Consultingwerk, Ltd.
  - http://pugchallenge.org/downloads2016/806 - OO-Oh.pdf
- (2018) 326: Building Great Interfaces with OOABL, Mike Fechner, Consultingwerk, Ltd.
  - http://pugchallenge.org/downloads2018/Fechner_OOABL.PDF
- Sessions by Peter Judge
  - excellent resources – couldn't find links :(
- OE Dev Tools Source
  - Progress Community Website
  - https://community.progress.com/community_groups/openedge_general/w/openedgegeneral/tags/development_5F00_tools

Questions?

PUGCHALLENGE AMERICAS EXCHANGE