Base Language - Feature #1590

implement PROPATH assignment

10/12/2012 04:34 PM - Greg Shah

Status:	Closed	**Start date:**	10/15/2012
Priority:	Normal	**Due date:**	
Assignee:	Costin Savin	**% Done:**	0%
Category:		**Estimated time:**	8.00 hours
Target version:	Runtime Support for Server Features		
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 10/12/2012 05:03 PM - Greg Shah

Progress has a PROPATH "function" (which looks like a well known or global variable) which reports on the current setting of the propath. It also has a PROPATH "statement" which looks like this:

PROPATH = <character_expression>.

It is essentially the way you dynamically assign/change the value of the propath. We already support the reading (PROPATH function), but we don't fully support the assignment (PROPATH statement) form. Your job is to:

1. Create some test cases to explore/demonstrate the behavior of PROPATH assignment.
2. Enhance the conversion to support PROPATH assignment.
3. Run the test cases and ensure that the support works, fixing anything in the runtime that is not right.

The runtime support for getting the PROPATH is in com/goldencode/p2j/util/EnvironmentOps.getSearchPath(). Various parts of the runtime use this value to calculate pathnames or find files. For example, look at FileSystemOps.searchPath() and FileSystemOps.ContextContainer.initialValue() where the search path (the propath) is sent down to be cached on the client.

An early version of the runtime support for setting the PROPATH is in com/goldencode/p2j/util/EnvironmentOps.setSearchPath(). HOWEVER, it only stores the change on the server side and I'm pretty sure it needs to also send it down to be cached on the client as is done in FileSystemOps.ContextContainer.initialValue(). Resolve this as needed.

The conversion support for querying the PROPATH can be found in rules/convert/variable_references.rules in the lines:

<rule>oldtype == prog.kw_propath
<action>methodTxt = "EnvironmentOps.getSearchPath"</action>
</rule>

There is no conversion support for setting the propath, yet. You should look to add it in this same variable_references.rules location. One effective way to do this is to change the "methodTxt" variable depending on whether the parent is an assignment operator:

<rule>oldtype == prog.kw_propath
<action>methodTxt = "EnvironmentOps.getSearchPath"</action>
<rule>parent.type == prog.assign
<action>methodTxt = "EnvironmentOps.setSearchPath"</action>
</rule>
</rule>

Until you get access to the 4GL development system, you will have to just implement this as best as you can determine. I would prefer that you can run your testcases on the 4GL dev system first, but it may take a few days to work that out.

#2 - 10/12/2012 05:04 PM - Greg Shah

- Estimated time changed from 4.00 to 8.00

#3 - 10/16/2012 04:49 AM - Costin Savin

- Status changed from New to WIP

#4 - 10/16/2012 01:29 PM - Costin Savin

<rule>oldtype == prog.kw_propath
<action>methodTxt = "EnvironmentOps.getSearchPath"</action>
<rule>parent.type == prog.assign
<action>methodTxt = "EnvironmentOps.setSearchPath"</action>
</rule>
</rule>

From what I understand to use this for creating the setter, the "assign" parent must be replaced to use instead the EnvironmentOps.setSearchPath methodName and the right side child of assign otherwise we end up with assign(EnvironmentOps.setSearchPath(), "<value>");? Is that correct? trying to find a way to accomplish this.

#5 - 10/16/2012 02:38 PM - Greg Shah

Yes, you're right. The assignment version of this must be put in convert/assignments.rules. Search on prog.kw_fr_val in both assignments.rules (for an example of how to emit the code) and in variable_references.rules (to see how to protect against the assignment case, since the get case should not emit if it has a parent that is the assign node).

#6 - 10/17/2012 08:17 AM - Costin Savin

Obtained the following rules settings that will convert the code according to the spec:

```
<rule>oldtype == prog.kw_propath
  <action>methodTxt = "EnvironmentOps.getSearchPath"</action>
  <rule>parent.type == prog.assign
    <action>methodTxt = null</action>
  </rule>
</rule>
```

inside **variable_references.rules**

and

```
<rule>oldtype == prog.kw_propath
  <action>methodtxt = "EnvironmentOps.setSearchPath"</action>
</rule>
```

inside **assignment.rules**
Will try additional tests to see if the code converts correctly in other situations than normal.

#7 - 10/17/2012 08:42 AM - Greg Shah

```
  <rule>compact
    <action>methodtxt = "EnvironmentOps.setSearchPath"</action>
    <action>ltimp = false</action>
  </rule>
```

This portion should be deleted for these reasons:

1. The entire section is only there if you were to want to emit a more compact form of the call. So instead of EnvironmentOps.setSearchPath() it might be just setSearchPath() with the addition of a static import for EnvironmentOps. But we don't need to do that in this case. PROPATH assignment is not done frequently enough to justify the additional static import.

2. The part regarding <action>ltimp = false</action> is actually a bug. The ltimp controls the addition of a static import for LogicalTerminal. That has nothing to do with the PROPATH case. It is being used in the setFrameValue() case to force a static import. The code here might turn off that needed import, causing a compile failure.

Otherwise the approach looks good on the conversion side.

#8 - 10/17/2012 11:35 AM - Costin Savin

- File cs_upd_20121017.zip added

Attached proposed update for the issue

#9 - 10/31/2012 01:52 PM - Greg Shah

- Target version set to Milestone 7

#10 - 11/02/2012 01:26 PM - Greg Shah

Feedback on the proposed changes:

1. Each file with a standard header needs the copyright notice updated to 2012 if it hasn't already been updated.

2. Each file with a standard header needs a history entry added to describe your change.

3. Don't import specific classes unless it is a requirement (this is only needed to resolve conflicts where the same class name appears in multiple packages).

4. EnvironmentOps changes:
- I don't want to have a reference to FileSystem here. That instance is already kept in FileSystemOps and it would be unexpected to have 2 instances of it.
- Even if it should go here, the approach you coded gets a new instance with every call to setSearchPath(String). We would only really want it to be done if work.obtain().fs == null, otherwise why are we caching the instance?
- Why are there any changes to setSearchPath(character)? The first line just unwraps the String value and calls the setSearchPath(String) method which already would handle all needed behavior. These changes cause this path to do the work twice. Trips down to the client are costly in terms of performance, so this MUST be avoided. Just remove all changes from this method.
- Instead of the current approach, I want you to add a package private static method to FileSystemOps called setSearchPath(String[] pathList, boolean caseSens). Then call this method from EnvironmentOps.setSearchPath(String).

- When you call the FileSystemOps.setSearchPath() to push the change down to the client, you must split it first. Your current code just sends the path down as new String[] { path }, which is not right. Please see getSearchPathList(). In fact, it might be best to create a private String[] pathToList(String) helper method in EnvironmentOps. Move the splitting code there from getSearchPathList() and call it from both getSearchPathList() and from setSearchPath().
- Please remove the check for !getLegacyCaseSensitive in setSearchPath(). The null check is enough. The caseSens check in FileSystemOps.ContextContainer.initialValue() is a minor performance optimization that should not be carried through to the EnvironmentOps code.
- FileSystemDaemon.setSearchPath() needs to be updated to set this.caseSens = caseSens! Otherwise we won't ever change the real value.

5. Please do not use hard tabs. Configure your editor to convert the tab key into 3 spaces so that there are no tab characters (0x09) in our source code.

#11 - 11/02/2012 01:27 PM - Greg Shah

When you post your updated code, do not delete the old update. I want those to stay around for reference purposes.

Also: make sure to use the proper file name convention (e.g. cs_upd20121017a.zip instead of cs_upd_20121017.zip).

#12 - 11/02/2012 01:53 PM - Greg Shah

Now that you have access to the 4GL development system, please make sure that you do more thorough testing of the PROPATH statement. The testcases do an assignment, but they don't do anything to actually test that the resulting assignment works.

There are 2 simple ways to test that the PROPATH is really different:

1. Use the PROPATH function to display it.

```
my-char-var = PROPATH.
if my-char-var <> "whatever_i_expected" then message "something went wrong".
```

2. Use the SEARCH built-in function to find specific files in the propath. You can search for something in the original propath (that deliberately can't be found), then assign the propath with something that adds a path that has what you are searching for, then search again (and hopefully find it this time).

Please enhance your testcases to use both these methods of checking. Also, the testcases don't have the paths setup with the proper separators. Windows uses ";" and Linux uses ":". Don't use spaces or other funky characters.

When using the database, wherever possible, use temp-tables. This will make it much easier on you and in most cases it is no different from permanent tables.

Make sure to also do some testing to ensure that the search order is well known. The search should start in the leftmost paths and move through the next path to the right in turns, until it finds what you are searching for or it can't find it. Setup a subdirectory tree with some bogus text files and multiple sub-paths. Include this in your update so that the testcases are self-contained and your work can be duplicated.

#13 - 11/05/2012 02:34 PM - Costin Savin

There seems to be a something wrong with the testcase when run on Progress or a Progress bug.
When assigning a new value to PROPATH it keeps some values and the new value is just added to that.
The search for a file whose containing directory or even complete file path, has been assigned to PROPATH doesn't return anything even is the path shows as being in PROPATH (as opposed to what p2j converted code does). However when searching for a file from the core PROPATH the correct result will be returned.
(Tried with both slash values when giving the paths)

#14 - 11/05/2012 02:36 PM - Greg Shah

Show the testcase code (that is behaving strangely) here and also show the output of running it on the 4GL. I will help you analyze the results.

#15 - 11/05/2012 02:51 PM - Costin Savin

Thanks I managed to figure out what the problem was :)!

#16 - 11/06/2012 10:01 AM - Costin Savin

- File cs_upd20121106a.zip added

Attached proposed update 2.

#17 - 11/06/2012 10:04 AM - Costin Savin

- File cs_upd20121106b.zip added

Attached equivalent tests used on progress windows os.

#18 - 11/06/2012 10:39 AM - Costin Savin

Tests gave the same results on both P2J and Progress
Given the following directory structure

```
propath-search
   |-dir_a
        |-dir_1
             |-lookup-1.file
        |-dir_2
             |-lookup.file
        |-lookup.file
        |-lookup-2.file

   |-dir_b
        |-dir_1
             |-lookup-1.file
        |-dir_2
             |-lookup.file
        |-lookup-2.file

   |-dir_c
        |-dir_1
             |-lookup-3.file
        |-lookup-2.file
```

and the following order of folders assigned to propath

```
PROPATH =
base-search-dir + "propath-search/dir_b/dir_1" + os-dir-separator +
base-search-dir + "propath-search/dir_c" + os-dir-separator +
base-search-dir + "propath-search/dir_a" + os-dir-separator +
```

```
base-search-dir + "propath-search/dir_b" + os-dir-separator +
base-search-dir + "propath-search/dir_b/dir_2" + os-dir-separator +
base-search-dir + "propath-search/dir_a/dir_2" + os-dir-separator +
base-search-dir + "propath-search/dir_b/dir_2" + os-dir-separator +
base-search-dir + "propath-search/dir_a/dir_1" + os-dir-separator.
```

Searching for **lookup.file** should return /propath-search/dir_a/lookup.file
Searching for **lookup-1.file** should return /propath-search/dir_b/dir_1/lookup-1.file
Searching for **lookup-2.file** should return /propath-search/dir_c/lookup-2.file
Searching for **lookup-3.file** should return unknown: ? because the path /propath-search/dir_c/dir_1 is not included in propath

The difference between P2J and Progress regarding PROPATH assignement is that Progress has a set of locations that will remain inside PROPATH when you change it's value (they are added at the end of PROPATH each time it's changed, and are probably only needed by Progress).

#19 - 11/06/2012 12:33 PM - Greg Shah

Please note that I am having to repeat the same feedback as provided previously, because some of the prior feedback was not resolved. Please take great care to follow and resolve EVERY feedback item before submitting the next proposed update. If anything is not clear, ask questions and we will discuss it.

Feedback:

1. Some files are still missing copyright changes (e.g. variable_references.rules, FileSystemOps...).

2. The history entries have a typo (assignement).

3. Some files still have hard TAB characters (e.g. variable_references.rules).

4. Always add a blank line between the end of one method and the beginning of the javadoc for the next method (e.g. FileSystemOps, EnvironmentOps).

5. Why is there an added work.obtain().fs.getSearchPath(); on line 878 of FileSystemOps? There isn't even such a method in FileSystem.java.

6. Line 905 of FileSystemOps should not extend past character 78. I would format it like this:

```
         work.obtain().fs = (FileSystem)
                            RemoteObject.obtainInstance(FileSystem.class, true);
```

7. More important than #6 is that this code should not be present in the file at all:

```
      if (work.obtain().fs == null)
      {
         work.obtain().fs = (FileSystem) RemoteObject.obtainInstance(FileSystem.class,
                                                                     true);
      }
```

The reason is that the work.obtain().fs is already set in initialValue() and so it will always be non-null. Look at the other locations in the file that use the fs member. It is intentional that they don't have any protective logic to set the fs if it is null, because it cannot be null.

8. Line 908 of FileSystemOps should have this removed: " || !caseSens". It is NOT valid to put that check in except ONLY in the initialValue() method.

9. Line 909 of FileSystemOps is not properly indented.

10. Don't put blank lines at the end of a method and before the closing }. We must put them after the closing } but NOT before (e.g. splitPath() in EnvironmentOps).

11. private methods should be grouped together at the bottom of the file. The coding standards define clear groupings by access control types. (e.g. splitPath() in EnvironmentOps).

12. Line 364 of EnvironmentOps has this code:

```
if (path != null )
```

Why have the extra space at the end? This is just distracting and makes the code look less regular. I know we are very picky about our code but it is for that reason that the code is very clean and regular. This makes a difference in the long run and so it is very important that you learn and follow the coding standards EXACTLY.

13. The javadoc for splitPath() needs to be formatted according to the coding standards.

14. The javadoc for splitPath() references pathSeparator. The reader can't really know what that means. What pathSeparator are you referring to? Be very clear in your documentation.

15. Line 616 of EnvironmentOps has this code which breaks from the coding standards:

```
} else
```

16. Where are the updates to FileSytemDaemon.java?

17. testcases/uast/propath-assignemnt.p has a mis-spelled file name.

18. Please don't send 2 copies of an almost same testcase. Just write one version that runs properly on both Windows and Linux. Make the paths relative and/or lookup paths dynamically at runtime. We definitely don't want to hard-code non-relative paths in these testcases.

19. This testcase is not extensive enough to show the behavior of propath assignment. For example, there are no searches before you assign the propath, so how do you know that the propath assignment is actually working? You need to have the same searches done before and after the assignment, where there are only supposed to be some changes. Design the testcase accordingly.

20. You have mentioned other findings about Progress propath having some fixed portion. The testcases should show this and you need to put more detail here about what this really means. If you have to display some output of the testcases here, that is fine.

#20 - 11/06/2012 03:16 PM - Costin Savin

3. I forgot that the indentation applies to all code not only to Java Files
For the formatting problems I've set the editor formatting rules according to the coding standard document
but there's clearly a problem with it, I will check it and check also manually for problems from now on.
16. I added that method (getSearchPath();) to FileSystem for debug reasons and didn't notice it was still there, this means there is no change to FileSystemDaemon
Will solve the other issues.
Sorry for the inconvenience.

#21 - 11/06/2012 03:35 PM - Greg Shah

FileSystemOps will still need the new setSearchPath() method exposed.

#22 - 11/07/2012 10:11 AM - Costin Savin

20.
The default propath on Progress will allways contain these values:

```
d:\oe102b\tty,d:\oe102b\tty\adecomm.pl,d:\oe102b\tty\adecomp.pl,d:\oe102b\tty\adeedit.pl,d:\oe102b\tty\adeshar
.pl,d:\oe102b\tty\prodict.pl,d:\oe102b\bin,
```

where d:\ is the location where oe102b folder is located

18.
In java the path used for search is relative to the folder from where p2j is run(../../uast/ will be the folder containing propath-search) , in Progress I'm not sure what are paths relative to.
Can the path and separator be asked as input from the user?

#23 - 11/07/2012 11:33 AM - Greg Shah

You should not need to prompt the user for pathing. Instead, use the filesystem features of the 4GL to make the program process identically. We want everything to be relative OR dynamically calculated from the current directory if it is necessary to make the paths absolute. In fact, your research needs to determine if there are any issues relating to relative or absolute paths in the 4GL propath support.

#24 - 11/07/2012 02:33 PM - Costin Savin

Researched and tried a couple of Progress methods to see if I can get to find the directory containing the test data dynamically, could you give me a hint of what I can use for this purpose to get the location of the source file and from there I could go to the test data directory ?(tried PROGRAM-NAME and OS-ENV and OS-DIR but none of them does what I need).

#25 - 11/07/2012 02:43 PM - Greg Shah

It seems like SEARCH) will return the fully qualified filename of the current program (so long as the current directory is included in the PROPATH).

You can check OPSYS and set the file and path separators based on the operating system. You only need to support Linux and Windows for now.

Then you can parse out the base name of the current program by finding the last path separator char. 4GL tools such as R-INDEX and SUBSTRING

will be helpful for this.

#26 - 11/07/2012 02:46 PM - Greg Shah

The other question: does Progress support relative paths in the PROPATH? I suspect it does, but you should determine this for sure.

In such a case, then the absolute path of the current directory doesn't matter, and you can use all relative paths for both Windows and Linux.

#27 - 11/08/2012 06:02 AM - Costin Savin

Progress seems to support relative paths in general so not sure why it wouldn't support it in PROPATH also
After running the following test file:

tst.p

```
DEFINE VARIABLE stat AS INTEGER.

DEFINE VARIABLE dir AS CHARACTER.

DEFINE VARIABLE dir1 AS CHARACTER.

/*File-name is supposed to hold the path to the compiled file
*however it holds the location of the temporary file created by
*Progress -> on my specific case the output is "D:\temp\oe102bworking\p57145_tst.ped"*/

message "{&FILE-NAME}".
dir = "test_root_directory".

dir1 = "../dir_path_relative".
/*try to create a dir in the root path*/
OS-CREATE-DIR VALUE (dir).
/* dir created in: D:\temp\oe102bworking -> this seems to be the reference location */

stat = OS-ERROR.

IF stat NE 0 THEN

   MESSAGE "FIRST DIRECTORY NOT CREATED. SYSTEM ERROR # " STAT.

/*try to create dir with relative path*/
OS-CREATE-DIR value (dir1).
/*dir successfully created in D:\temp */
stat = OS-ERROR.

IF stat NE 0 THEN

   MESSAGE "SECOND DIRECTORY NOT CREATED. SYSTEM ERROR # " STAT.
```

The problem is how to access my test folder when I can't get the location of where the source code is, and the relative path reference starts from Progress's temporary folder which we can't know what position it has compared to where the test source code might be.
The only solution that comes to my mind is to create the test folder and file structure to be used for test inside the Progress source code relative to the reference path whichever it might be (client run location in p2j and temporary work dir in Progress - permission to write to that location would also be required for it to work).

#28 - 11/08/2012 08:47 AM - Greg Shah

Your problems all stem from the fact that when you run the test program from the "Procedure Editor", the environment (and even the filename of the code) will all be defined by the Procedure Editor.

You can avoid all these problems by just running the .p file directly from your own working directory using the pro/mpro commands (for character mode) or the prowin32 (for GUI mode) programs. I think generally, you can pass -p my-program.p to run a specific program.

Please see the Progress "Startup Command and Parameter Reference" for details.

#29 - 11/08/2012 09:12 AM - Costin Savin

prowin32 and pro are not recognized as internal or external command, operable program or batch file on the Progress test system.
Maybe some system variable are not correctly set?

#30 - 11/08/2012 09:47 AM - Greg Shah

I don't know, it is possible that the PATH does not include the Progress directories. But you can do 2 simple things:

1. Look at how the OpenEdge icons/shortcuts are configured (their properties). It should show you the path and file name of the programs being run.

2. Search in the OpenEdge directories for those executibles. Normally they reside in a bin/ directory on UNIX.

#31 - 11/08/2012 11:01 AM - Costin Savin

Ran with the full path of prowin32:
When it is ran in this configuration the reference path "." is the directory where the source code is found and "{&FILE-NAME}" returns the source file relative to current directory like ".\source-file.p"
When ran in p2j the relative path starts from where the client is run (not from where the progress source code is) the test propath-serch location could be calculated depending on OS just like the separator (linux = p2j , windows=progress)

The alternative is to create the test directory configuration and access it from the relative path (in this case there will be the same path just relative to different directories).

Which of the 2 approaches would be preferable?

#32 - 11/08/2012 11:08 AM - Greg Shah

```
When it is ran in this configuration the reference path "." is the directory where the source code is found an
d "{&FILE-NAME}" returns the source file relative to current directory like ".\source-file.p"
```

I suspect that your current directory when you ran the prowin32 was just the same as where the source code was located. In other words, Progress leaves your current directory alone when you run it.

```
When ran in p2j the relative path starts from where the client is run (not from where the progress source code
 is )
```

As I noted above, I think it is the same behavior. It is just that in P2J, the code is gone. But you can run it from wherever you want.

```
the test propath-serch location could be calculated depending on OS just like the separator (linux = p2j , win
dows=progress)
```

I don't think this conclusion is correct. If it is correct, then please show me the conclusive evidence. Because that would mean that our propath behavior in P2J is wrong and must be fixed. But I think your interpretation of the situation may be incorrect.

```
The alternative is to create the test directory configuration and access it from the relative path (in this ca
se there will be the same path just relative to different directories).
```

This is exactly what I was originally suggesting. Go this way. But please prove or disprove the my points above.

#33 - 11/08/2012 11:55 AM - Costin Savin

I will implementing the 2nd choice, what I wanted to say about p2j starting from where the client is running referred to running using the client-terminal script from testcases/simple/client ,it can of course be modified to run from the location we want or run without using a script, but with the 2nd approach it will not be necessary to do any of that

#34 - 11/08/2012 02:47 PM - Costin Savin

Tried the following for creating a file

```
...
fileLoc = dir-a + "/lookup.file"
OUTPUT stream s1 to fileLoc.
PUT STREAM s1 UNFORMATTED SKIP.
OUTPUT stream s1 close
```

where dir-s is a variable containing a relative path. This creates a file called fileLoc.
tried copying the variable with OS-COPY

```
OS-COPY "fileLoc" fileLoc.
```

```
OS-COPY "fileLoc" /propath-search/dir-a/lookup.file.
```

```
OS-COPY "fileLoc" "./propath-search/dir-a/lookup.file".
```

none of these with the expected result.

#35 - 11/08/2012 03:26 PM - Greg Shah

This does not work in the 4GL for a simple reason: most language statements that deal with the file system do not DIRECTLY allow sub-expressions to be used for specifying filenames. The syntax LOOKS like it should be possible. But instead the filenames are treated like unquoted literal strings instead of a character expression.

```
fileLoc = dir-a + "/lookup.file"
OUTPUT stream s1 to fileLoc.
```

The above is the same as this:

```
OUTPUT stream s1 to "fileLoc".
```

In both cases these are essentially a literal (hard coded) filename in the source code and it does not dynamically vary at runtime because fileLoc is not a reference to a variable.

If you want to substitute a character expression, use VALUE like this:

```
myCharacterVar = dir-a + "/lookup.file"
OUTPUT stream s1 TO VALUE(myCharacterVar).
```

This is also valid:

```
OUTPUT stream s1 TO VALUE(dir-a + "/lookup.file").
```

In both cases, the VALUE construct tells the 4GL to evaluate the contained character expression and pass the result to the OUTPUT TO language statement.

You will have to look at the 4GL reference manual to determine where VALUE can be used. Not all language statements support it, but most do.

#36 - 11/09/2012 04:23 PM - Costin Savin

There seems to be a problem with slash and backslash when defining directories
When run on windows PROGRESS using "/" to delimit directories fails and "\" works, if "\" is used with p2j conversion fails with unexpected char: 0xFFFF error, will try to find a workaround to this

#37 - 11/09/2012 05:30 PM - Greg Shah

For now, just use the preprocessor conditional processing and the OPSYS built-in to detect the platform on which the code is running. Then set a local variable with the correct file separator character. I think something like this may work:

```
&IF "{&OPSYS}" = "WIN32" &THEN
def var file-sep as char initial "/".
&ELSE
def var file-sep as char initial "\".
&ENDIF
```

#38 - 11/10/2012 12:12 PM - Costin Savin

Already tried that with the same result - conversion fail, however double "\\" seems to work fine with p2j and progress.

#39 - 11/10/2012 12:23 PM - Costin Savin

- File cs_upd_20121110a.zip added

Added proposed update.

#40 - 11/12/2012 08:10 AM - Greg Shah

Please show a very small sample 4GL procedure here where Progress deals with "\" correctly and P2J fails. Make the code as small as you can make it to recreate the problem. Also post the error you see.

The problem is that we treat the \ as an escape character on Linux. But I think it also has to do with your p2j.cfg.xml not telling the preprocessor that it must act like it is running on "Windows".

#41 - 11/12/2012 08:27 AM - Greg Shah

Feedback on the code:

1. EnvironmentOps still needs the copyright date updated to 2012.

2. Please put the first line of EnvironmentOps.setSearchPath() inside the if (path != null) {} block. The idea is that a null input is invalid all the time and we should not save it, ever.

3. The EnvironmentOps.splitPath() javadoc formatting still needs minor changes to match the coding standards.

4. Always add a blank line between the end of one method and the beginning of the javadoc for the next method. This is still a problem for FileSystemOps.setSearchPath().

5. The FileSystemDaemon.java changes are still missing. If the propath is never saved on the client side, how can the propath assignment ever work?

6. The testcase should be enhanced to test some same searches BEFORE the propath is ever modified (including before it is RESET). In addition, there should be at least 1 case where something is FOUND before the changes and after the changes it is NOT found.

#42 - 11/12/2012 10:16 AM - Costin Savin

After looking for error replication discovered it fails when \ is at the end of the string operand.
Example:

```
define variable path-a as character.
path-a = "dir_a\" + "dir_b". /*This fails!*/
```

```
define variable path-a as character.
path-a = "dir_a" + "\dir_b". /*This does not fail*/
```

The error message is my case is :
Exception: test-backslash.p:2:27: unexpected char: 0xFFFF
ClearStream state: clearCount = 0; inComment = no; inString = yes; keepTildes = no

6. Choose to reset the Propath before searching, because if test is run more than once the value remained in propath from the last run a better aproach would be to assign the original **propath** value to a variable and then assign it back to **propath** once the test is finished.
For file found before the change it would be difficult to find the same thing on p2j and progress since the default value in p2j is the current dir "." from which the client is run and we can't know for sure what we can find there (maybe the client script?), while in Progress it has some default files and directories (which may also differ from OS to OS).

Should the next proposed update be added as attached archive or using Bazaar?

#43 - 11/12/2012 10:43 AM - Greg Shah

Please see the P2J Conversion Handbook, Project Setup chapter. In the "Global Configuration" section, look at the "unix-escapes" value. Try setting that to "false" in your p2j.cfg.xml and re-test your small failing test.

```
6. Choose to reset the Propath before searching, because if test is run more than once the value remained in p
ropath from the last run a better aproach would be to assign the original propath value to a variable and then
 assign it back to propath once the test is finished.
```

I think this is a problem with using the procedure editor. If you run the program using:

```
prowin32 -p test-program.p
```

then this will not happen.

```
For file found before the change it would be difficult to find the same thing on p2j and progress since the de
fault value in p2j is the current dir "." from which the client is run and we can't know for sure what we can
find there (maybe the client script?), while in Progress it has some default files and directories (which may
also differ from OS to OS).
```

Just move the directory and file creation to run before the section that does the first set of checks (which must be before the first propath change). That way you always have that directory structure and set of files to work with.

```
Should the next proposed update be added as attached archive or using Bazaar?
```

Still in Bazaar. Even after the code review passes, the changes must pass full regression testing before they can be approved for check in. And even the check in must be done with very specific timing. So once it passes, you still may need to hold the change until the time is right, because others may need to check in their changes first. We very carefully serialize our check ins.

#44 - 11/12/2012 01:02 PM - Costin Savin

- File cs_upd20121112a.zip added

Added revised proposed update

#45 - 11/13/2012 09:48 AM - Greg Shah

Feedback:

1. Please see the JavaDoc section of the doc.html section of the coding standards. Your EnvironmentOps.splitPath() and FileSystemOps.setSearchPath() javadoc is still not matching this standard:

```
@param entries should appear in a single group and must appear before any @return entry.  @return should appea
r before any @throws entries.  The sections for @param, @return and @throws should be separated by an empty li
ne. Note the format of the @param and @throws entries:  the name is on the first line and the description star
ts on the second line (and may be split onto multiple lines to meet the column width limitation).  Both the na
me and description are left justified starting in the same column.
```

```
The @param, @return and @throws entries all have their names and/or descriptions starting in the same column.
```

2. FileSystemDaemon has no history entry and its copyright date is not updated.

3. Why do you do the following in FileSystemDaemon.setSearchPath()?

this.caseSens = !caseSens;

This parameter is not meant to be a "toggle". It is just meant to be assigned, as far as I understand it. What is your thinking behind this change?

#46 - 11/13/2012 10:52 AM - Costin Savin

- File cs_upd20121113a.zip added

Added revised update,
1. Corrected the Javadoc , hopefully I done most of the code formatting mistakes to remember them in the future.
2-3. Had some changes made to this file for debug but when I reverted to add back only the code for this task I messed it up.
this.caseSens = !caseSens; should be this.caseSens = caseSens;

#47 - 11/13/2012 11:27 AM - Greg Shah

Looks good.

The next step is to get it fully regression tested. This change affects both conversion and runtime, so both kinds of regression testing are needed. Please work with Constantin to get both accomplished. Until we have the new development server setup, the runtime regression testing of Majic will have to occur on lightning staging. But the conversion regression testing can occur on your system.

If both of those pass, we will then move on to check-in and distribution of the change.

#48 - 11/14/2012 04:59 AM - Constantin Asofiei

MAJIC conversion fails because cases like this:

```
&IF DEFINED(EXCLUDE-test-function) = 0 &THEN

&ANALYZE-SUSPEND _UIB-CODE-BLOCK _FUNCTION-FORWARD test-function Procedure
FUNCTION test-function RETURNS CHARACTER()  FORWARD.
/* _UIB-CODE-BLOCK-END */
&ANALYZE-RESUME
&ENDIF
```

i.e. when there are preprocessor directives, and failing with this stacktrace:

```
java.lang.NullPointerException
        at com.goldencode.p2j.net.RemoteObject.obtainInstance(RemoteObject.java:1010)
        at com.goldencode.p2j.util.FileSystemOps$ContextContainer.initialValue(FileSystemOps.java:954)
        at com.goldencode.p2j.security.ContextLocal$Fallback.initialValue(ContextLocal.java:400)
        at java.lang.ThreadLocal.setInitialValue(ThreadLocal.java:141)
        at java.lang.ThreadLocal.get(ThreadLocal.java:131)
        at com.goldencode.p2j.security.ContextLocal.get(ContextLocal.java:156)
        at com.goldencode.p2j.util.FileSystemOps$ContextContainer.obtain(FileSystemOps.java:940)
        at com.goldencode.p2j.util.FileSystemOps.setSearchPath(FileSystemOps.java:905)
        at com.goldencode.p2j.util.EnvironmentOps.setSearchPath(EnvironmentOps.java:367)
        at com.goldencode.p2j.preproc.Preprocessor.evaluate(Preprocessor.java:543)
        at com.goldencode.p2j.preproc.TextParser.condtext(TextParser.java:905)
        at com.goldencode.p2j.preproc.TextParser.aif(TextParser.java:641)
        at com.goldencode.p2j.preproc.TextParser.ppstatement(TextParser.java:292)
        at com.goldencode.p2j.preproc.TextParser.textBlock(TextParser.java:209)
        at com.goldencode.p2j.preproc.TextParser.text(TextParser.java:164)
        at com.goldencode.p2j.preproc.Preprocessor.<init>(Preprocessor.java:704)
        at com.goldencode.p2j.uast.AstGenerator.preprocess(AstGenerator.java:1242)
        at com.goldencode.p2j.uast.AstGenerator.prepareDataStream(AstGenerator.java:1022)
        at com.goldencode.p2j.uast.AstGenerator.prepareLexer(AstGenerator.java:1477)
        at com.goldencode.p2j.uast.AstGenerator.parse(AstGenerator.java:1394)
        at com.goldencode.p2j.uast.AstGenerator.processFile(AstGenerator.java:942)
        at com.goldencode.p2j.uast.AstGenerator.processFile(AstGenerator.java:814)
        at com.goldencode.p2j.uast.ScanDriver.scan(ScanDriver.java:203)
        at com.goldencode.p2j.uast.ScanDriver.scan(ScanDriver.java:122)
        at com.goldencode.p2j.convert.ConversionDriver.runScanDriver(ConversionDriver.java:386)
```

```
        at com.goldencode.p2j.convert.ConversionDriver.front(ConversionDriver.java:283)
        at com.goldencode.p2j.convert.ConversionDriver.main(ConversionDriver.java:1649)
```

The problem is that the Preprocessor.evaluate calls EnvironmentOps.setSearchPath which in turn needs a FileSystem network instance, which is not available during conversion. I suggest to add a parameter to the EnvironmentOps.setSearchPath which will not call the FileSystem.setSearchPath, if in conversion mode:

```
   public static void setSearchPath(String path, boolean conversion)
   {
      if (path != null)
      {
         work.obtain().propath = path;
         if (!conversion)
         {
            String[] pathList = splitPath(path);
            FileSystemOps.setSearchPath(pathList, getLegacyCaseSensitive());
         }
      }
   }
```

and Preprocessor.evaluate will set this parameter to true.

#49 - 11/14/2012 09:09 AM - Greg Shah

This sounds like a good approach.

Costin: please make the changes to EnvironmentOps and to Preprocessor.evaluate(). You will be **adding** a new version of setSearchPath() to the file. The logic in the current setSearchPath() should be moved to the new 2 parm version. Then the old 1 parm version would call the 2 parm version (which would be the "worker" that backs both methods). Of course, when called internally, it would pass "false" for the 2nd parm.

Once the changes are made, upload them here for review.

#50 - 11/14/2012 02:15 PM - Costin Savin

I made the changes and ran the regression tests however I got a message at the end that Error occurred during initialization of VM.Could not reserve enough space for object heap

```
Elapsed job time:  01:17:12
```

```
Buildfile: build.xml

prepare:
     [copy] Copying 646 files to /home/costins/workspace/p2jconvert/build/classes
     [copy] Copying 110 files to /home/costins/workspace/p2jconvert/src

compile_jasper:
      [jrc] Compiling 1 report design files.
      [jrc] File : /home/costins/workspace/p2jconvert/srcnew/jrxml/SA-M122P_20110126_20110126.jrxml ... OK.

compile:
    [javac] Compiling 14607 source files to /home/costins/workspace/p2jconvert/build/classes
    [javac] Error occurred during initialization of VM
    [javac] Could not reserve enough space for object heap
    [javac] Could not create the Java virtual machine.

BUILD FAILED
/home/costins/workspace/p2jconvert/build.xml:431: Compile failed; see the compiler error output for details.
```

I'm guessing that doing other work while this runs might cause this.
Will run the test again then add the update after checking regression.

#51 - 11/14/2012 02:29 PM - Greg Shah

Only the build has failed. Check the build.xml to see the amount of memory needed for the compile target. Change that if needed. Then try the "ant jar" again.

Even without building, the conversion is done so you can compare the generated code.

#52 - 11/15/2012 05:17 AM - Costin Savin

- File cs_upd20121115a.zip added

Added proposed update. Done regression tests - resulting sources are identical.

#53 - 11/15/2012 11:09 AM - Greg Shah

The proposed update looks good. Please work with Constantin to get runtime regression testing done on lightning staging.

#54 - 11/15/2012 12:37 PM - Constantin Asofiei

Regression testing is running in a staging copy. If it passes, I'll apply the update to staging.

#55 - 11/16/2012 07:09 AM - Constantin Asofiei

Runtime regression testing has passed. The update was applied to staging (without the testcase), staging P2J was rebuilt.

Costin: you can commit the update to bzr and release it to the team. But before doing so, please split it into two separate updates:

- one with the P2J sources
- one with the 4GL testcases

Don't forget to commit the testcase to the new "testcases" bzr project.

#56 - 11/28/2012 09:00 AM - Costin Savin

- Status changed from WIP to Review

#57 - 11/28/2012 09:39 AM - Greg Shah

- Status changed from Review to Closed

#58 - 11/16/2016 11:43 AM - Greg Shah

- Target version changed from Milestone 7 to Runtime Support for Server Features

Files

cs_upd_20121017.zip	19.5 KB	10/17/2012	Costin Savin
cs_upd20121106a.zip	29.5 KB	11/06/2012	Costin Savin
cs_upd20121106b.zip	2.87 KB	11/06/2012	Costin Savin
cs_upd_20121110a.zip	25.8 KB	11/10/2012	Costin Savin
cs_upd20121112a.zip	32.3 KB	11/12/2012	Costin Savin
cs_upd20121113a.zip	32.4 KB	11/13/2012	Costin Savin
cs_upd20121115a.zip	42.5 KB	11/15/2012	Costin Savin