Conversion Tools - Feature #3225

windows vs linux in p2j.cfg.xml

01/12/2017 02:41 PM - Greg Shah

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Ovidiu Maxiniuc	**% Done:**	70%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**version:**	
vendor_id:	GCD		

Description

Related issues:

Related to Conversion Tools - Feature #3247: need for different configuration...	**New**
Related to Conversion Tools - Bug #5021: improve/fix file-ignore-list.txt ('X...	**Closed**
Related to Base Language - Feature #3396: shift searchpath and other pathing ...	**WIP**

History

#1 - 01/12/2017 02:58 PM - Greg Shah

When we run conversion, you have to have a different p2j.cfg.xml file that depends on whether you are running the conversion on Windows or on Linux. This is true, even for the same project.

Here is an example:

```
<       <parameter name="p2j_rules"        value="${P2J_HOME}/p2j/rules" />
<       <parameter name="patpath"          value=".:${P2J_HOME}/pattern:${p2j_rules}/include:${p2j_rules}:" />
<       <parameter name="registry"         value="${P2J_HOME}/cfg/registry.xml" />
---
>       <parameter name="p2j_rules"        value="${P2J_HOME}\p2j\rules" />
>       <parameter name="patpath"          value=".;${P2J_HOME}\pattern;${p2j_rules}\include;${p2j_rules};" />
>       <parameter name="registry"         value="${P2J_HOME}\cfg\registry.xml" />
12c12
<       <parameter name="rootlist"         value="${P2J_HOME}/cfg/rootlist.xml" />
---
>       <parameter name="rootlist"         value="${P2J_HOME}\cfg\rootlist.xml" />
15,24c15,22
<       <parameter name="path-separator"      value=":" />
<       <parameter name="file-separator"      value="/" />
<       <parameter name="case-sensitive"      value="true" />
<       <parameter name="basepath"            value="./src/some/app" />
<       <parameter name="opsys"               value="UNIX" />
---
>       <parameter name="path-separator"   value=";" />
>       <parameter name="file-separator"   value="\" />
>       <parameter name="case-sensitive"   value="false" />
>       <parameter name="basepath"         value=".\src\some\app" />
>       <parameter name="opsys"            value="WIN32" />
27c25
<       <parameter name="propath"          value="${P2J_HOME}:${P2J_HOME}/src/some/app:${P2J_HOME}/src/some/ap
p/more:" />
---
>       <parameter name="propath"          value="${P2J_HOME};${P2J_HOME}\src\some\app;${P2J_HOME}\src\some\ap
p\more;" />
30,31c28,29
<       <parameter name="output-root"      value="${P2J_HOME}/src" />
---
>       <parameter name="output-root"      value="${P2J_HOME}\src" />
```

This is for the same exact source code. Please eliminate the need to edit the p2j.cfg.xml depending on the platform on which we are running the

conversion.

I prefer to encode everything in a standard way (using / and :). The opsys value in the example above is probably not an issue (I think it could have been set either way and not made a difference).

#2 - 01/13/2017 12:54 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> I prefer to encode everything in a standard way (using / and :).

Committed 1st revision. As documented in code, I used OS independent internal separators for testing. In the branch I switched to unix values for these. When the configuration file is loaded, they are replaced with current OS values.

> The opsys value in the example above is probably not an issue (I think it could have been set either way and not made a difference).

It is used to select the proper _user metadata import file. (on Unix systems, default dump name is _User.d, on Windows, _user.d). So this value depends on the OS where the db dump was executed.

The same stands for case-sensitive, path-separator and file-separator. They are used by NameMappingWorker and SourceNameMapper to emulate (if possible) the OS where the 4GL program was exported from. Having different values for these parameters seems to be incorrect.

What it seems incorrect to me is having the unix-escapes parameter. Shouldn't this value be extracted from opsys?

I have some concerns about the propath. This should be using the separators from the OS where the program was imported (the 4GL programmer can access/parse and even modify this at his own will). We are trying to create a similar environment but using the internal P2J_HOME variable, which uses separators from the current OS. Maybe we should reverse the switch operation so that the 4GL procedures to receive the value using the file separators as its programmer expected.

Task branch 3225a created from 11135 trunk. Only the Configuration was altered in the first commit. Not yet tested ion Windows.

#3 - 01/13/2017 03:05 PM - Constantin Asofiei

Ovidiu Maxiniuc wrote:

> The same stands for case-sensitive, path-separator and file-separator. They are used by NameMappingWorker and SourceNameMapper to emulate (if possible) the OS where the 4GL program was exported from. Having different values for these parameters seems to be incorrect.

The problem here is that when converting on Windows, I had to change this to be Windows-specific, even if the 4GL project was linux-compatible.

#4 - 01/19/2017 01:32 PM - Ovidiu Maxiniuc

- % Done changed from 0 to 70

Status of: 3225a/11139:

- passed full regression testing on devsrv01;
- conversion successful on both Linux/Windows (without altering the cfg file);
- reimported the whole database on windows (there was some path issue but this update has fixed it).

I am going now to run it through ETF on Linux and probably for Windows too, at least to know where we stand.

#5 - 01/20/2017 10:19 AM - Constantin Asofiei

Ovidiu, review for 3225a rev 11141 :

- beside the basepath which must use file-separator specific to the OS where conversion is run, why do you change these legacy values?

```
         /* values from the original system on which Progress 4GL ran */
         "file-separator", "path-separator", "propath",
```

  Greg, please confirm, that these must be the legacy values as found on the OS where the legacy application was run, not where conversion is done. (with the exception that propath is changed to have its entries with P2J_HOME prefix). The Hotel conversion works OK on Windows if unix-style path and file separators are used here.
- please check these values, too, which might be OS paths (I've searched for Configuration.getParameter in Java sources and getConfigParameter in TRPL rules):
  1. the values in SymbolResolver - SKELETON_PATH, OO4GL_PATH, DOTNET_PATH, ASSEMBLY_PATH
  2. CallGraphWorker - callgraph-db-folder
  3. AstGenerator - error-dest
  4. ConversionDriver - codenames
  5. annotations/annotations.xml - name_map_merge
  6. schema/dmo_common.rules - merge-dmo-root, dmo_index_merge

#6 - 01/20/2017 11:15 AM - Greg Shah

> please confirm, that these must be the legacy values as found on the OS where the legacy application was run, not where conversion is done

Yes, the file-separator and path-separator are intended to be the ones from the original system on which the 4GL application code ran. The idea is that we must be able to parse filenames/paths that can be provided at conversion time (e.g. an include file name) or which can be calculated and provided at runtime (e.g. an external procedure passed to a RUN statement). In both cases, they will have coded things to work on the original 4GL installation system.

I agree that the propath can be done in a standard way and doesn't need to match. In fact we really don't want to ever have to change that.

> beside the basepath which must use file-separator specific to the OS where conversion is run

Actually, why not do this in a standard way too (always using /) and then we should automatically deal with Windows if needed. I don't want to have to code this differently depending on where the conversion is running. The idea with this task is to eliminate that from all cases.

#7 - 01/20/2017 11:30 AM - Ovidiu Maxiniuc

I might not have understand it fully, but I had the feeling that the propath, file-separator and path-separator from p2j.cfg.xml are used by the conversion, and the values of propath, file-separator and path-separator from directory: /server/<name>/runtime/default/ are used at runtime.

basepath uses 'neutral' separator / and is converted to '\' only when running on Windows.

#8 - 01/20/2017 11:35 AM - Greg Shah

> I might not have understand it fully, but I had the feeling that the propath, file-separator and path-separator from p2j.cfg.xml are used by the conversion

That is true.

But the idea is that propath can be independent but that file-separator and path-separator still need to tell us about the original system. The good news is that we don't need 2 versions of this. We just need to match the original system. If there are more than one original system then we pick one.

> basepath uses 'neutral' separator / and is converted to '\' only when running on Windows.

Perfect. That is what I was looking for and it is what I am calling the "standard form".

> the values of propath, file-separator and path-separator from directory: /server/<name>/runtime/default/ are used at runtime

Yes. I would want the directory propath to be a standard form. But again, the file-separator and path-separator still need to tell us about the original system, even if we are reading them from the directory.

#9 - 01/20/2017 12:58 PM - Ovidiu Maxiniuc

OK, so file-separator and path-separator are used by conversion to know how to parse the legacy strings found in code and propath. In which case propath must be using separators from the legacy system, ie it must be just copied from there. However, this is not always possible for at least 2 main reasons:

- OS filesytems differ;
- absolute paths.

OTOH, we force in the definition of propath parameters that are dependent on the current machine (like P2J_HOME) that is expanded when the parameter is loaded. So we get the parameter having a mixture of paths from the legacy and current machine.

If we are using this propath just for finding files to be converted, then we should just ignore the separators at all, knowing that the paths are either absolute (composed using P2J_HOME computed on current machine) or relative to basepath.

At runtime, there is another story, and the respective values are read from directory (which is not accessible at conversion time, anyway).

#10 - 01/20/2017 01:06 PM - Greg Shah

> so file-separator and path-separator are used by conversion to know how to parse the legacy strings found in code and propath

Since propath is something that is configured in p2j.cfg.xml and it is not read from a legacy file, we can make things simpler. Please remove the dependency on the configured file-separator and path-separator in this case.

> OTOH, we force in the definition of propath parameters that are dependent on the current machine (like P2J_HOME) that is expanded when the parameter is loaded. So we get the parameter having a mixture of paths from the legacy and current machine.

I don't want things coded to the current machine. I want a standard approach (which would be / and :). Then we should translate to the current system if needed (e.g. Windows).

> If we are using this propath just for finding files to be converted, then we should just ignore the separators at all, knowing that the paths are either absolute (composed using P2J_HOME computed on current machine) or relative to basepath.

Yes. I think my approach as defined above accomplishes this.

#11 - 01/20/2017 01:22 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> OTOH, we force in the definition of propath parameters that are dependent on the current machine (like P2J_HOME) that is expanded when the parameter is loaded. So we get the parameter having a mixture of paths from the legacy and current machine.

I don't want things coded to the current machine. I want a standard approach (which would be / and :). Then we should translate to the current system if needed (e.g. Windows).

This is the exact implementation in rev 11141. I will fix the separators definition also.

#12 - 01/20/2017 01:29 PM - Greg Shah

> I will fix the separators definition also.

What do you mean?

I think the file-separator and path-separator need to match the original 4GL system.

#13 - 01/24/2017 08:14 AM - Ovidiu Maxiniuc

Task branch 3225a was updated yesterday. The current revision is 11142. The latest changes expect that all paths in p2j.cfg.xml to use Unix-style separators for file and paths. The case-sensitive, file-separator and path-separator must be configured with values form legacy OS. Some of the files that were doing additional file/path processing were simplified. The Configuration class now contains a summary of all parameters that configures FWD.

The revision passed flawlessly the regression testing on devsrv01 and converted successfully hotel and hotel_gui projects on both Linux and Windows (of course, the p2j.cfg.xml had to be fixed manually after running prepare_hotel.cmd on Windows). The later script and templates for Windows must be dropped or unified with Linux variant.

The task branch is ready to be merged to trunk. Please review. Meanwhile I will re-run the full ETF set of tests.

#14 - 01/25/2017 08:46 AM - Constantin Asofiei

Ovidiu, 3225a rev 11143 looks OK - I see that you rely on OPSYS=WIN32 to determine if this is a Windows OS. I need to double-check on Windows, the Hotel GUI's p2j.cfg.xml will use always / and :

#15 - 01/25/2017 09:03 AM - Greg Shah

To what value are you defaulting the WINDOWS-SYSTEM preprocessor define?

#16 - 01/25/2017 09:44 AM - Ovidiu Maxiniuc

SESSION:WINDOW-SYSTEM handling is unchanged. It should be TTY as defaulted in com.goldencode.p2j.preproc.Options.java.
This value is supposed to be TTY for Unix because there are no native ABL implementations to support a GUI on UNIX, but it makes sense now with FWD to define other values.

#17 - 01/25/2017 10:51 AM - Greg Shah

For GUI projects, we should default WINDOW-SYSTEM to MS-WIN95 for both:

- LogicalTerminal.getWindowSystem() (this is the runtime SESSION:WINDOW-SYSTEM)
- ...preproc.Options.getWindowSystem() (this is the preprocessor define WINDOW-SYSTEM)

#18 - 01/25/2017 10:58 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> For GUI projects, we should default WINDOW-SYSTEM to MS-WIN95 for both:
>
> - LogicalTerminal.getWindowSystem() (this is the runtime SESSION:WINDOW-SYSTEM)
> - ...preproc.Options.getWindowSystem() (this is the preprocessor define WINDOW-SYSTEM)

How should FWD know which kind of project is processing to use the default setting? It seems to me that this is a mandatory value to be set in configuration files for GUI projects. Am I wrong?

#19 - 01/25/2017 11:06 AM - Greg Shah

I believe that if the original application code is run in prowin32.exe, then WINDOW-SYSTEM will return MS-WIN95 or MS-WINXP (supposedly only in the presence of the manifest file). One can also override to get MS-WINDOWS but this takes extra configuration. It is a reasonable approach to return MS-WIN95 by default for a GUI app. We already know at runtime if we are isChui(), so we should be able to make this determination in LogicalTerminal.

I suspect that running ABL batch, appserver or ChUI code on Windows (Progress) may have some scenarios where TTY is returned. But **I have not tested or proven this**. If correct, checking isChUI() should be sufficient to return a good default.

In regard to the preprocessor, I think it is safe to default to MS-WIN95 when OPSYS is WIN32. Again, we may have some cases where the customer has Windows-specific code that runs in a mode that needs (and gets in Progress) a TTY. But I think that is the less likely case. By default, MS-WIN95 is a good setting.

#20 - 01/25/2017 11:21 AM - Greg Shah

Code Review Task Branch 3225a Revision 11144

I like the changes.

Other than the WINDOW-SYSTEM defaults, what else is left for this task?

#21 - 01/25/2017 11:32 AM - Ovidiu Maxiniuc

I already rebased the branch to latest trunk (rev 11144). Before rebasing, the branch passed all tests (devsrv01 and ETF).

I will add the default value for WINDOW-SYSTEM to MS-WIN95 when when OPSYS is WIN32. Of course, this is a default value and can be overwritten at any moment if the customer wants or the project need some other windowing system. I don't think this will require retesting. I am not aware of any other things to do as part of this task.

#22 - 01/25/2017 11:50 AM - Greg Shah

> I don't think this will require retesting.

Agreed.

Excellent. After the WINDOW-SYSTEM changes are checked in, please test 2 other scenarios using that recent customer's large GUI application:

- the gui-specific project config
- the common (server + gui) project config

Modify the p2j.cfg.xml in both projects to set opsys and remove all the defaulted values. Then run conversion to confirm everything is OK.

If that works, please merge to trunk.

#23 - 01/25/2017 11:51 AM - Constantin Asofiei

Ovidiu, can you take a look at searchpath in directory.xml, too? From what I see, this is OS-dependent, too...

#24 - 01/26/2017 01:21 PM - Greg Shah

How is testing going?

#25 - 01/26/2017 01:51 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> How is testing going?

I started the over night conversion for GUI, hoping to have it ready in the morning. Unfortunately, I forgot to disable the hibernate feature in Power Manager so after a couple of hours the conversion was suspended to disk. The good part is that there were no conversion errors. I managed to get it started and play around with a couple of testcases in the webpage.

Meanwhile, after the GUI conversion ended, I started the conversion of combined project (server + GUI). It is running for 9+ hours already and seems to have at least 2 more hours to go :(. I really do not understand why is it so sluggish. This was a little tricky to configure. The server was configured with opsys=UNIX and the GUI with opsys=win32. Since in the combined project they share the same parameter, I've chosen the opsys=win32, mainly because of GUI requirements.

#26 - 01/26/2017 01:58 PM - Greg Shah

> Since in the combined project they share the same parameter, I've chosen the opsys=win32, mainly because of GUI requirements.

Although it may work for that particular customer, we definitely need to allow for multiple configurations at conversion-time. Let's see if there is any issue with the test environment. If not, then we can move ahead now but we will need to add a task to allow different opsys values (and different overrides for the individual values) for different parts of the project.

#27 - 01/27/2017 06:02 AM - Ovidiu Maxiniuc

The conversion of combined project finished without errors, configured with opsys=win32.
I ran a combined test (navigated through GUI in web interface while allSearchTests was executed in background). The ETF test was successful and I did not encounter any issue while working with GUI client. I guess this task is ready to be merged to trunk.

#28 - 01/27/2017 06:59 AM - Greg Shah

I agree, let's get this into the trunk. Go ahead with the merge.

1. Did you look at searchpath (CA's question in note 23)?

2. Please create a new task to allow different opsys values (and different overrides for the individual values) for different parts of a conversion project (for the same purpose as the work in #2475). We don't have to work that task right now, but I want to make sure we don't forget the requirement.

#29 - 01/27/2017 08:48 AM - Ovidiu Maxiniuc

Greg Shah wrote:

> I agree, let's get this into the trunk. Go ahead with the merge.

The changes from branch were merged to trunk as rev 11139. The branch and test results were archived. Notification was sent to team.

> 1. Did you look at searchpath (CA's question in note 23)?

Now I am working on this. I'll be back later.

> 2. Please create a new task to allow different opsys values (and different overrides for the individual values) for different parts of a conversion project (for the same purpose as the work in #2475). We don't have to work that task right now, but I want to make sure we don't forget the requirement.

OK

#30 - 01/27/2017 08:57 AM - Greg Shah

Please update the Conversion Handbook Project Setup chapter with the changes to the various configuration values. It probably makes sense to update the example p2j.cfg.xml too.

Do you have simplified p2j.cfg.xml files for the 3 customer projects that you tested (server, gui and combined)? If so, they can be checked in.

#31 - 01/27/2017 12:09 PM - Ovidiu Maxiniuc

Greg Shah wrote:

> Please update the Conversion Handbook Project Setup chapter with the changes to the various configuration values. It probably makes sense to update the example p2j.cfg.xml too.

The Global Configuration section of Project Setup was updated to reflect the new semantics of p2j.cfg.xml.

> Do you have simplified p2j.cfg.xml files for the 3 customer projects that you tested (server, gui and combined)? If so, they can be checked in.

Yes, all 3 files are committed. I wondered whether to do the same for the project used in regression testing on devsrv01, but I think it can stay unchanged, the final configuration is, evidently, the same.

#32 - 02/10/2017 01:56 PM - Ovidiu Maxiniuc

- Related to Feature #3247: need for different configuration values for different parts of a conversion project added

#33 - 02/13/2017 12:24 PM - Ovidiu Maxiniuc

Constantin Asofiei wrote:

> Ovidiu, can you take a look at searchpath in directory.xml, too? From what I see, this is OS-dependent, too...

I analysed the searchpath along with related other two, searchpath-fixed and file-system/propath from directory.xml. I created some basic testcases to test all four combinations of OS. With this opportunity I discovered a flaw in the implementation of the PROPATH statement. I fixed it, but going back to runtime configuration, I am afraid we need to keep OS dependency for these directory entries for a simple reason: we cannot add Windows absolute paths using OS neutral paths &n file separators (Ex: .:d:/dlc/src:d:/dlc/src/common) because : is interpreted as a 'neutral' path separator (so, for this example, we get 5 incorrect paths instead of 3).

Note the following of my discoveries from working with propath in p4gl:

- the propath is simplified when set: adding multiple times the same path, only the first occurrence is kept. When comparing and deciding whether paths are equivalent:
  - paths are case sensitive on Linux and case insensitive on Windows (as expected);
  - / and \ are considered equivalent. Once you added \home\om\work, you cannot append /home/om/work for both OSes. There is a catch here: on Windows, paths using / are valid and work, but on Linux the path are invalid and programs cannot be found in paths with reverted separators;
- both , and native path separators can be used.

We have at least two other alternate solutions:

- use the legacy separators for searchpath and file-system/propath. searchpath-fixed should use local OS separators because it is programatically added to existing path list. This has the disadvantage of not being consistent between the three of them;
- use progress-style separators, this way being OS neutral.

#34 - 05/02/2017 04:22 PM - Greg Shah

> use progress-style separators, this way being OS neutral

This seems to make good sense.

#35 - 11/30/2020 02:59 PM - Greg Shah

- Related to Bug #5021: improve/fix file-ignore-list.txt ('X' conversion mode) under Windows added

#36 - 11/30/2020 03:01 PM - Greg Shah

The changes in this task need to extend to the hints processing for the related values. For example, you can define propath overrides in hints files which should be shifted to a platform neutral approach. The current code is broken when such entries are defined for Linux or Windows and then used on the other platform.

#37 - 11/30/2020 03:01 PM - Greg Shah

- Related to Feature #3396: shift searchpath and other pathing lists in the directory to use the platform neutral comma added