Automated Build Generator - Bug #2021
Projects that contain pom.xml and CMakeLists.txt get treated as maven project only

09/24/2014 10:38 AM - S. Meyer zu Borgsen

Status:	Resolved	**Start date:**	09/24/2014
Priority:	Normal	**Due date:**	
Assignee:	J. Moringen	**% Done:**	100%
Category:	Project Analysis	**Estimated time:**	0.00 hour
Target version:	0.3		

Description

The BTL project (https://projects.cit-ec.uni-bielefeld.de/git/btl.git)
contains a pom for building the java part and a CMakeLists for the cpp part.

This results in incorrect dependency resolution.
Even if the template is set to cmake/cmake-cpp, the project gets only analyzed for the maven part.

See btl-cpp.project

Related issues:

Related to Automated Build Generator - Bug # 2022: Incorrect pkg-config requi...	**Rejected**	**09/24/2014**

Associated revisions

Revision c970a5e4 - 10/08/2014 05:26 PM - J. Moringen

Multiple natures/nature specification in src/analysis/*.lisp

fixes #2021

- src/analysis/analysis.lisp (guess-project-natures): renamed
  project-kind -> guess-project-natures; return a list of all applicable
  natures
  (analyze pathname eql :auto): handle multiple natures
  (analyze pathname cons): new method: analyze according to all natures
- src/analysis/git.lisp (analyze uri eql :git): pass unknown keyword
  arguments to `analyze' :auto call
- src/analysis/git.lisp (analyze uri eql :svn): likewise
- src/analysis/archive.lisp (analyze uri eql :archive): likewise
- src/commandline-interface/main.lisp (analyze-project): pass list of
  natures to `analyze'

Revision 4a874e6b - 10/09/2014 07:12 PM - J. Moringen

Backport: Fixes for multi-nature analysis in src/analysis/analysis.lisp

refs #2021

- src/analysis/analysis.lisp (analyze pathname eql :auto): when merging
  analysis results, produce a plist, not an alist; fixed logging

Revision 3b8f2655 - 10/09/2014 07:16 PM - J. Moringen

Fixes for multi-nature analysis in src/analysis/analysis.lisp

refs #2021

- src/analysis/analysis.lisp (analyze pathname eql :auto): when merging
  analysis results, produce a plist, not an alist; fixed logging

History

#1 - 09/24/2014 08:18 PM - J. Moringen

- Related to Bug #2022: Incorrect pkg-config requirement extraction for cmake-cpp projects added

#2 - 09/26/2014 06:09 PM - J. Moringen

- Status changed from New to Feedback

- Target version set to 0.3

One solution could be performing all applicable analyses and merging the results.

@Johannes: do you see any problems (e.g. for RST) with this approach?

For example, for BTL these results could be merged:

```
CL-USER> (jenkins.analysis:analyze #P"/tmp/btl/" :auto)

(

 ; Maven result
 (:VERSIONS
  ((:MAIN ("de.unibi.citec.clf/btl" "2.0-SNAPSHOT")))
  :PROVIDES
  ((:MAVEN "de.unibi.citec.clf/btl" (2 0 "SNAPSHOT"))
   (:MAVEN "de.unibi.citec.clf/btl-xml" (2 0 "SNAPSHOT"))
   (:MAVEN "de.unibi.citec.clf/btl-base" (2 0 "SNAPSHOT"))
   (:MAVEN "de.unibi.citec.clf/btl-rst" (2 0 "SNAPSHOT")))
  :REQUIRES
  ((:MAVEN "log4j/log4j" (1 2 16))
   (:MAVEN "junit/junit/test" (4 10))
   (:MAVEN "java3d/vecmath" (1 3 1))
   (:MAVEN "org.apache.commons/commons-lang3" (3 1))
   (:MAVEN "rsb/rsb" (0 11 "SNAPSHOT"))
   (:MAVEN "rsb/rst-sandbox" (0 11 "SNAPSHOT"))
   (:MAVEN "rsb/rst" (0 11 "SNAPSHOT"))
   (:MAVEN "org.reflections/reflections" (0 9 8))
   (:MAVEN "de.unibi.citec.clf/btl-base" (2 0 "SNAPSHOT"))
   (:MAVEN "xom/xom" (1 2 5))
   (:MAVEN "org.easymock/easymockclassextension/test" (2 5 2))
   (:MAVEN "org.easymock/easymock/test" (2 5 2))
   (:MAVEN "org.apache.commons/commons-lang3" (3 1))
   (:MAVEN "java3d/vecmath" (1 3 1))
   (:MAVEN "junit/junit/test" (4 10))
   (:MAVEN "log4j/log4j" (1 2 16)))
  :URL "https://projects.cit-ec.uni-bielefeld.de/projects/btl"
  :LICENSE "LGPL-3"
```

```
 :PROPERTIES
 (("project.build.sourceEncoding" . "UTF-8")
  ("license.licenseName" . "btl_license")
  ("license.licenseResolver" . "${project.baseUri}/src/license")))

; CMake result
(:VERSIONS ((:MAIN NIL))
 :PROVIDES
 ((:CMAKE "BTL" NIL)
  (:CMAKE "btl" NIL)
  (:CMAKE "BTL-BASE" NIL)
  (:CMAKE "btl-base" NIL)
  (:CMAKE "BTL-RST" NIL)
  (:CMAKE "btl-rst" NIL)
  (:CMAKE "BTL-XML" NIL)
  (:CMAKE "btl-xml" NIL)
  (:PKG-CONFIG "BielefeldTypeLibrary" ("@BTL_VERSION@"))
  (:PKG-CONFIG "BielefeldTypeLibrary" ("@BTL_VERSION@"))
  (:PKG-CONFIG "BielefeldTypeLibrary" ("@BTL_VERSION@"))
  (:PKG-CONFIG "BielefeldTypeLibrary" ("@BTL_VERSION@")))
 :REQUIRES NIL :PROPERTIES ((:LICENSE . "LGPL-3")))

)
```

#3 - 09/26/2014 06:49 PM - J. Wienke

I think that for RST this doesn't occur since the pom file is generated by cmake and hidden in subfolders.

Merging might also be the wrong behavior since in case the project only builds the C++ parts, it should not depend on java artifacts. Maybe there should be a way to couple the template to the performed repository analysis so that for a cpp project only analysis of cmake is performed and vice versa.

#4 - 09/26/2014 06:52 PM - J. Moringen

> *Merging might also be the wrong behavior since in case the project only builds the C++ parts, it should not depend on java artifacts. Maybe there should be a way to couple the template to the performed repository analysis so that for a cpp project only analysis of cmake is performed and vice versa.*

Your suggestion implies having two or more recipes, one for each "nature" (in Eclipse speak), of such projects?

#5 - 09/26/2014 06:57 PM - J. Wienke

Jan Moringen wrote:

> *Merging might also be the wrong behavior since in case the project only builds the C++ parts, it should not depend on java artifacts. Maybe there should be a way to couple the template to the performed repository analysis so that for a cpp project only analysis of cmake is performed and vice versa.*

> *Your suggestion implies having two or more recipes, one for each "nature" (in Eclipse speak), of such projects?*

That's probably necessary? Or does it work to have a project that uses the maven template as well as the cmake template in parallel? I would suspect a lot of unexpected results from that.

Actually looking at the BTL source code, it seems that the CMake script triggers maven manually. So in this case, merging the dependencies is probably ok, but I don't think one can generalize this across all different projects. So it would probably be good if there was a variable in the generator to select the analyzers to use. Each template can then set the one analyzer it is designed for. In case of BTL, the user then still has the option to additionally configure the maven analyzer. In case multiple analyzers exists as chosen by the user, merging dependencies is then ok, to my mind.

#6 - 10/09/2014 01:10 PM - J. Moringen

- Status changed from Feedback to Resolved

- % Done changed from 0 to 100

Applied in changeset commit:c970a5e422ea540920a3b0e8a7a150772c366a43.