CoCoALib - Design #1221

Reconsider design for accessing headers and libs of external libraries

16 Aug 2018 09:22 - John Abbott

Status:	Closed	**Start date:**	16 Aug 2018
Priority:	Normal	**Due date:**	
Assignee:	John Abbott	**% Done:**	100%
Category:	Portability	**Estimated time:**	0.22 hour
Target version:	CoCoALib-0.99800	**Spent time:**	0.30 hour

Description

The current design for accessing headers of external libs more or less assumes that the external lib has a single header file.

It may be better to maintain an external reference to a directory containing the header files of each external lib.

Consider; and implement if the idea seems good.

Related issues:

Related to CoCoALib - Feature #1219: Frobby version number	**In Progress**	**16 Aug 2018**

History

#1 - 16 Aug 2018 09:22 - John Abbott

- Related to Feature #1219: Frobby version number added

#2 - 23 Sep 2019 12:46 - John Abbott

- Target version changed from CoCoALib-0.99650 November 2019 to CoCoALib-0.99700

#3 - 08 Jan 2020 22:56 - John Abbott

- Target version changed from CoCoALib-0.99700 to CoCoALib-0.99800

#4 - 06 Oct 2020 14:57 - John Abbott

- Status changed from New to In Progress

- Assignee set to John Abbott

- % Done changed from 0 to 50

I wonder exactly what I had in mind when creating this issue.

The current situation is as follows: in ExternalLibs/include the symbolic links for **normaliz** and **gfanlib** actually point to **directories.**
This structure seems to be necessary since the unified header files are actually full of include directives which assume that the directory containing all the gfanlib/normaliz header files are in the search path.
NOTE CoCoALib also uses this approach for the unified header file.

The current design seems to work fine (for me and for others I know); I have heard of no complaints.

Should we regard this as actually resolved? Perhaps even simply close the issue?

#5 - 09 Oct 2020 13:54 - John Abbott

- Status changed from In Progress to Closed

- % Done changed from 50 to 100

- Estimated time set to 0.22 h

Closing because the current design seems to work well (and has not changed for many months).

Surely the recorded time for this issue is wrong!