

CoCoALib - Feature #721

CheckForInterrupt: string arg to specify where it was called?

29 May 2015 16:16 - John Abbott

Status:	Closed	Start date:	29 May 2015
Priority:	Normal	Due date:	
Assignee:	John Abbott	% Done:	100%
Category:	Improving	Estimated time:	2.20 hours
Target version:	CoCoALib-0.99536 June 2015	Spent time:	2.25 hours
Description Anna indirectly suggested that it might be nice to have an optional arg to CheckForInterrupt which takes a string indicating where it was called from.			
Related issues: Related to CoCoALib - Feature #718: Insert calls to CheckForInterrupt Related to CoCoALib - Feature #385: Design new errors using inheritance Related to CoCoALib - Design #427: Error names and error messages (current de...			
		Closed	21 May 2015
		In Progress	08 Jul 2013
		In Progress	28 Jan 2014

History

#1 - 29 May 2015 16:19 - John Abbott

It looks to be almost trivial to add CheckForInterrupt(const char*) and CheckForInterrupt(const std::string&).

It could indeed be useful on occasion to know where an interrupted program was (e.g. if the interrupt was sent because the program appeared to be in an infinite loop).

If it is as easy as I suspect then it's probably worth doing (quickly, before too many calls to CheckForInterrupt have already been inserted into code).

#2 - 29 May 2015 17:20 - Anna Maria Bigatti

In fact I was suggesting something else: CheckInterrupt throws something which is not a CoCoA_ERROR, so cocoa-5 does interrupt, but doesn't say anything.

That's why I added the try/catch: just to return a CoCoA_ERROR.

By default the cocoa-5 interpreter does not say **where** the error was (and that's why I dislike ERR::BadArg, but that's another topic ;-)

#3 - 30 May 2015 10:01 - John Abbott

- Status changed from New to In Progress
- % Done changed from 0 to 10

I prefer that the exception thrown is not of the same type as a normal CoCoA_ERROR because the reasons behind the exception are different:

- (A) a CoCoA::ErrorInfo is a message from CoCoALib about some internal problem;
- (B) a CoCoA::InterruptReceived is a response to an external signal.

We could have a CoCoA::exception class which is a super-class for the types of all exceptions that CoCoALib will throw; but then I would anyway have separate subclasses for (A) and (B) for the same reason as above.

Having different types makes it easy to catch them separately and thus handle them differently. I don't think it should be too hard to make CoCoA-5 print out a message when it has been interrupted (though some aspects of the interpreter are rather mysterious to me).

Having a common super-class would make it possible to catch all CoCoALib exceptions; I'm not sure how useful that could be. Nevertheless, it does seem a neater design to have a common super-class.

UPDATE (2015-07-31) the CoCoAInterpreter does not distinguish between exceptions from interrupts and exceptions from mathematical errors; this

leads to some messy code with lots of catch (const InterruptExc&) { throw; } catch (const RuntimeExc&) {...}

#4 - 30 May 2015 14:52 - John Abbott

I have implemented my suggestion in note 1.

It occurred to me that if we do allow a string as arg (to where CheckForInterrupt was called) then probably we should **require** there to be a string arg (i.e. it always says where CheckForInterrupt was called). I think it costs the programmer very little to add the string; the run-time cost should be negligible (esp. if a string literal, that it const char*, is used); and it may occasionally be useful.

Consequently, I have removed CheckForInterrupt() and replaced it by two homonyms which require char* or string as arg.

#5 - 25 Jun 2015 17:27 - John Abbott

- Assignee set to John Abbott

- % Done changed from 10 to 30

I have checked in the current version (whose ctor requires a string). Waiting for Anna to check it.

At the moment CoCoA-5 interpreter ignores the "where" information (it's probably not very useful to a normal CoCoA-5 user).

#6 - 26 Jun 2015 17:03 - John Abbott

- Status changed from In Progress to Feedback

- % Done changed from 30 to 90

Code has been updated, tested, documented (with simple example). Checked in.
Changing status to feedback.

#7 - 30 Jun 2015 11:58 - Anna Maria Bigatti

Doesn't seem to be enough to give a suitable error message to cocoa5.

(interrupt code is in TmpGReductor.C and test code, to be interrupted, for cocoa5 is $g := \text{GBasis}(\text{ideal}((x+y+z+1)^{19}, (x+2*y-z)^{20}));$)

#8 - 01 Jul 2015 18:12 - John Abbott

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

- Estimated time set to 2.20 h

After a brief testing period, the mechanism works (but the C5 interpreter needs to be improved so that a helpful message is printed out -- see issue [#744](#)).