Base Language - Feature #6500

add equivalent support for -nb command line option

06/07/2022 10:25 AM - Constantin Asofiei

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**version:**	
vendor_id:	GCD		

Description

History

#2 - 06/07/2022 10:33 AM - Constantin Asofiei

- Tracker changed from Bug to Feature

OpenEdge allows specification of the top-level stack depth via the -nb command line option. This raises a STOP condition when this limit is reached - which can be caught by the application and allow the execution to continue.

In Java, recovering from a StackOverflowError I don't think is possible. I've done some tests with a simple recursive 4GL function call (calls itself into infinite recursion), and this allows ~950 BlockManager.processBody calls on the Java stack.

Although this is an application error, which can be fixed in the original code, FWD should be able to recover from this. A possible way is to add a limit of the processBody calls on the stack, and when this is reached, raise a STOP condition. This limit should be somewhere ~700-800 calls by default, and allow configuration in the directory, per user/group/server (is there a reason to have a bootstrap config on the client?). But some tests should be done with RUN statements, as these consume more stack entries than a simple recursive function call, and adjust this number.

This will be a deviation from -nb - which tracks top-level blocks on the stack - but doing this in FWD is fragile, as we don't have control over the stack consumed by the other blocks. That's why processBody is a better place to track the stack depth.

#3 - 06/07/2022 10:43 AM - Greg Shah

How likely is it that we can find a good default value? I think leaving the default as "uncatchable abend" is not good for production.

#4 - 06/07/2022 10:51 AM - Constantin Asofiei

Greg Shah wrote:

> How likely is it that we can find a good default value?

There are a couple of issues of concern:

- leave the default too small, and FWD raises a STOP condition when it should not
- leave the default too big, and we end up with StackOverflowError

For the 'good default value', we need to analyze the different type of possible calls - like RUN, DYNAMIC-INVOKE, etc - and see at what stack depth FWD raises a StackOverflowError. I don't think there is a way to have a 'this is the exact depth at which a StackOverflowError will be encountered in

Java', we need an approximate value which doesn't limit the stack depth in normal operation, and also prevents StackOverflowError.

> I think leaving the default as "uncatchable abend" is not good for production.

My point here is more to 'harden' FWD so that it does not end up with stack overflow. For production, all these cases should be fixed already in the converted application code - but, if one happens, FWD should be able to recover from it and do all the runtime processing while unwinding the stack (like rollback).

#5 - 06/07/2022 10:53 AM - Greg Shah

> My point here is more to 'harden' FWD so that it does not end up with stack overflow. For production, all these cases should be fixed already in the converted application code - but, if one happens, FWD should be able to recover from it and do all the runtime processing while unwinding the stack (like rollback).

Yes, that is exactly what I'm saying.