Conversion Tools - Feature #2503

improve ConversionDriver.main() to provide back a return code and/or better handle exceptions so that using it in ant or scripts will provide the caller with failure notification

01/26/2015 11:37 AM - Greg Shah

Status:	WIP	**Start date:**	01/26/2015
Priority:	Normal	**Due date:**	
Assignee:	Eric Faulhaber	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	Deployment and Management Improvements		
billable:	No	**vendor_id:**	GCD

Description

History

#1 - 01/26/2015 11:38 AM - Greg Shah

From a customer:

> Conversion failed last night (because we checked something in that we shouldn't have done - a change to the keyword test).
> This is good - we've already found the mistake and fixed it.
>
> What's less good is the conversion job reported success and the rest of the pipeline (building converted code; deploying; running tests) went ahead regardless.
>
> Would it be possible for a non-zero exit code to be returned in the event of conversion exceptions?
>
> Specifically, it would be good if ConversionDriver.main() didn't swallow exceptions. Perhaps the catch block at the end of that method could be changed to throw a RuntimeException?

#2 - 08/03/2015 05:26 AM - Paul E

If this is a small job (I think it is, unless I'm missing something), then I'd really appreciate it being a high priority.
I'm happy to look at it again and submit a pull request if you like?

I've just spent 1hr trying to work out why all of our test servers failed to pass a basic sanity test on each run of our pipeline over the weekend.
The reason is that conversion failed and so we built incomplete customer app jars.

#3 - 08/04/2015 06:03 AM - Paul E

Can you please add the following line of code in the exception catch block at the end of the main() method in ConversionDriver?:

```
System.exit(1);
```

I don't care about any specific error codes for why it failed, I just care about seeing a non-zero exit code so it's easy for me to script around it rather than having to get into parsing the log.

This would have saved me quite a bit of time on three occasions now.

Can you please let me know if you're not happy to do this soon, so that I can patch it myself if that's the case?

#4 - 08/04/2015 08:44 AM - Greg Shah

This won't be a problem. Eric will include this change shortly.

#5 - 08/04/2015 05:32 PM - Eric Faulhaber

- Assignee set to Eric Faulhaber

- Status changed from New to WIP

The update is applied to branch 2503a, rev. 10909, based on trunk rev. 10908. Performed ad-hoc testing only, regression testing not required.

#6 - 08/04/2015 06:38 PM - Paul E

Great, thanks guys!

#7 - 08/04/2015 07:27 PM - Eric Faulhaber

Paul Eames wrote:

> Great, thanks guys!

No problem.

Branch 2503a was merged into trunk rev. 10909 and the branch was archived.

Greg: I don't know if you had something more elaborate in mind for the long term w.r.t. implementing more specific return codes. If not, please go ahead and close this issue.

#8 - 08/05/2015 08:17 AM - Greg Shah

We intend to provide a detailed return code that will allow detection of specific failures.

#9 - 02/26/2016 08:56 AM - Paul E

I was happy about 7 months ago that the fix in rev 10909 gave me what I needed. It appears there is at least one case where this alone was not enough. We checked in some broken Progress code (code that would not compile correctly in Progress) to our test branch and the P2J parser quite reasonably failed to parse it, throwing an exception. This exception is then swallowed on line 206 of com.goldencode.p2j.uast.ScanDriver. I can see why that is done - in earlier stages of a project you'll want to progress with conversion irrespective of errors. Can this behaviour be toggleable please? I don't really mind how it's achieved: either it could propagate the exception out (i.e. fail conversion on the first failure of this kind), or you could

implement an information gatherer that gathers a list of failures of this kind before failing after this conversion pass. The main thing for us is that we need this to fail as it's substantially lower effort to find a failure at this stage than it is at runtime.

The relevant portion of the stacktrace is as follows:

```
com.goldencode.ast.AstException: Error processing <redacted>
        at com.goldencode.p2j.uast.AstGenerator.processFile(AstGenerator.java:949)
        at com.goldencode.p2j.uast.AstGenerator.processFile(AstGenerator.java:816)
        at com.goldencode.p2j.uast.ScanDriver.scan(ScanDriver.java:203)
        at com.goldencode.p2j.uast.ScanDriver.scan(ScanDriver.java:122)
        at com.goldencode.p2j.convert.ConversionDriver.runScanDriver(ConversionDriver.java:397)
        at com.goldencode.p2j.convert.ConversionDriver.front(ConversionDriver.java:294)
        at com.goldencode.p2j.convert.ConversionDriver.main(ConversionDriver.java:1795)
```

#10 - 03/28/2016 11:24 AM - Eric Faulhaber

I've recently been making changes in this area and so I tried to implement this behavior quickly as part of this effort. Unfortunately, fully implementing this feature will require a higher level of effort than I had hoped. My initial attempt catches some errors at one of the higher level classes and aborts the scan if a new flag is set. However, many lower level errors are handled/reported and then suppressed in a number of places deeper in the code. To adjust all these locations will require a greater refactoring effort than I can take on at the moment, but we will come back to this.

#11 - 11/16/2016 01:14 PM - Greg Shah

- Target version changed from 24 to Deployment and Management Improvements