Conversion Tools - Bug #2356

pphints bug - nested includes are missing from the.pphints file

08/04/2014 07:28 AM - Constantin Asofiei

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	0%
Category:		**Estimated time:**	4.00 hours
Target version:	Reporting 3.0		
billable:	No	**case_num:**	
vendor_id:	GCD		

Description

Related issues:

Related to Conversion Tools - Feature #2251: improve the call graph generation	**Closed**	**03/03/2014**

History

#1 - 08/04/2014 07:30 AM - Constantin Asofiei

Cases like this:

```
e.g. {include-file1.i
&some-foo-var = "{include-file2.i}"
}
```

are not emitted properly in the .pphints file: the nested include file (include-file2.i) is expanded properly, but is not included as a reference via an include node, in the associated .pphints file - and the callgraph misses it.

Original issue is in #2260 note 13.

#2 - 04/27/2017 10:59 AM - Greg Shah

- Target version set to Reporting 3.0

- Status changed from New to Rejected

I've looked deeper at this. Expansions that are inside of preprocessor directives are deliberately excluded from the pphints mechanism. The idea is that these expansions do not map directly to code in the output file, so we they weren't tracked. For now, this really should not affect the call graph processing, since the call graph is based on the AST which is based on the emitted code. Ultimately, if some text did get expanded inside a directive and came out in the code, then it will be associated with the include file that actually generated the code. I think that is OK.

For this reason, I'm closing this issue.

#3 - 04/27/2017 12:19 PM - Constantin Asofiei

Greg Shah wrote:

> I've looked deeper at this. Expansions that are inside of preprocessor directives are deliberately excluded from the pphints mechanism. The idea is that these expansions do not map directly to code in the output file, so we they weren't tracked. For now, this really should not affect the call graph processing, since the call graph is based on the AST which is based on the emitted code. Ultimately, if some text did get expanded inside a directive and came out in the code, then it will be associated with the include file that actually generated the code. I think that is OK.
>
> For this reason, I'm closing this issue.

A problem is that this include (include-file2.i in the example above) will be picked up as "dead" by the list_dead_files.rules report.

#4 - 04/27/2017 12:40 PM - Greg Shah

- Status changed from Rejected to New

OK, that makes sense.