Base Language - Bug #4483

Unable to use logical variable check in underline statement

12/17/2019 11:39 AM - Roger Borrello

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#2 - 12/17/2019 11:55 AM - Roger Borrello

Testcase

20191231 Update: Fixed in 4207a-11357

Below is the testcase uast/underline_logic_uses_when.p:

```
def var v1 as character.
def var v2 as integer.
def var l1 as logical initial false.
form
   v1 v2
   with frame f1.

/* This yields logical cannot be converted to boolean */
underline
  v1 when l1 = true
  v2 when l1 = false
  with frame f1.
assign v1 = if l1 then "" else "Hello".

/* This yields "error: not a statement"
underline
  v1 when wip-postcode-opt = true
  with frame f1.
assign v1 = if l1 then "" else "Hello".
*/

message "Done.".
```

It generates this error:

```
    [javac] testcases/uast/uast/buildarea/src/com/goldencode/testcases/abl/UnderlineLogic.java:36: error: inco
mpatible types: logical cannot be converted to boolean
    [javac]             isEqual(l1, true) ? f1Frame.widgetV1() : null,
    [javac]                     ^
    [javac] /home/rfb/projects/VirtualBox-VMs/shared/projects/testcases/uast/uast/buildarea/src/com/goldencode
/testcases/abl/UnderlineLogic.java:37: error: incompatible types: logical cannot be converted to boolean
    [javac]             isEqual(l1, false) ? f1Frame.widgetV2() : null
    [javac]                     ^
```

#3 - 12/17/2019 11:59 AM - Roger Borrello

Roger Borrello wrote:

> # Testcase
>
> Below is the testcase uast/underline_logic_uses_when.p:
>
> [...]
>
> It generates this error:
>
> [...]

It looks to me like rules/convert/operators.rules has the logic for deciding whether to use **isEqual** or **_isEqual**. But it only checks for that if getNoteBoolean("unknown_override") is true (I only see my **NOT** printfln):

```
         <!-- EQUALS -->
         <rule>type == prog.equals
            <rule>getNoteBoolean("unknown_override")
                 <action>printfln("operators.walk_rules: unknown_override %s", this.dumpTree())</action>
                 <action on="false">printfln("operators.walk_rules: NOT unknown_override %s", this.dumpTree())<
/action>
               <!-- simpler unknown value path -->
               <rule>noUnwrap
                  <action>jtext = "_isUnknown"</action>
                  <action on="false">jtext = "isUnknown"</action>
               </rule>

               <rule>javaMethodNames.contains(jtext)
                  <action>jtext = sprintf("CompareOps.%s", jtext)</action>
               </rule>

               <!-- normal path -->
               <rule on="false">noUnwrap
                  <action>jtext = "_isEqual"</action>
                  <action on="false">jtext = "isEqual"</action>
               </rule>
            </rule>
            <action>compimp = true</action>
         </rule>
```

Am I in the right ballpark? Should I more focus on the WHEN?

#4 - 12/17/2019 12:41 PM - Constantin Asofiei

Roger, see the noUnwrap logic - this should decide if _isEqual or isEqual is emitted, and looks like it depends on parentCanAvoidUnwrap function in common-progress.rules; check why this function doesn't return true in your when case.

#5 - 12/17/2019 02:06 PM - Roger Borrello

Constantin Asofiei wrote:

> Roger, see the noUnwrap logic - this should decide if _isEqual or isEqual is emitted, and looks like it depends on parentCanAvoidUnwrap function in common-progress.rules; check why this function doesn't return true in your when case.

Very good hint! :-)

The existing WHEN check was focused on WHEN_LIST/KW_WHEN and KW_COLOR/CONTENT_ARRAY/KW_WHEN. When I added STATEMENT/KW_UNDERLIN/CONTENT_ARRAY/KW_WHEN the correct logical _isEqual is emitted. Thanks! However, I'd still like to know if my code addition could be improved. Is it necessary for me to go all the way back to STATEMENT? The existing code didn't go back that far. Which is best practice?

```
<!-- WHEN clauses in UI statements -->
<rule>pref.upPath("WHEN_LIST/KW_WHEN") or
      pref.upPath("KW_COLOR/CONTENT_ARRAY/KW_WHEN") or
      pref.upPath("STATEMENT/KW_UNDERLIN/CONTENT_ARRAY/KW_WHEN")
   <action>avoid = true</action>
</rule>
```

#6 - 12/17/2019 02:59 PM - Greg Shah

The CONTENT_ARRAY is only found in the MESSAGE, ENABLE, DISABLE, COLOR and UNDERLINE statements, but only COLOR and UNDERLINE can have WHEN clauses. For this reason, it is safe to remove the new code you added and also remove the KW_COLOR:

```
<!-- WHEN clauses in UI statements -->
<rule>pref.upPath("WHEN_LIST/KW_WHEN") or
      pref.upPath("CONTENT_ARRAY/KW_WHEN")
   <action>avoid = true</action>
</rule>
```

#7 - 12/17/2019 03:16 PM - Roger Borrello

Greg Shah wrote:

> The CONTENT_ARRAY is only found in the MESSAGE, ENABLE, DISABLE, COLOR and UNDERLINE statements, but only COLOR and UNDERLINE can have WHEN clauses. For this reason, it is safe to remove the new code you added and also remove the KW_COLOR:
>
> [...]

Made that update, and validated against testcase and customer code.

Check-in 4207a-11357.

#8 - 12/17/2019 03:18 PM - Roger Borrello

- Status changed from New to WIP

#9 - 12/31/2019 12:00 PM - Greg Shah

- Status changed from WIP to Test

- % Done changed from 0 to 100

#10 - 03/03/2020 02:53 PM - Roger Borrello

Task branch 4207a was merged to trunk as revision 11344.

#11 - 03/04/2020 10:31 AM - Greg Shah

- Status changed from Test to Closed