Base Language - Bug #5104

Cannot cast integer input parameter to dynamic function character parameter

01/26/2021 02:29 PM - Roger Borrello

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

Description

History

#2 - 01/26/2021 03:10 PM - Roger Borrello

This testcase can duplicate the problem:

```
function mydyn1 returns character (
   input pi1 as integer,
   input pc2 as character,
   input pc3 as character ) :
   return "mydyn1".
end function.

function mydyn2 returns character (input pi1 as character) : return "mydyn2".
end function.

pause.

message dynamic-function('mydyn1', input 1, input 'int', input 1).

/*message dynamic-function('mydyn2', input 1).*/

message "Done.".
```

Interestingly, if you uncomment the second call, it will succeed.

#3 - 01/26/2021 03:16 PM - Constantin Asofiei

I see what you mean now. I read the last argument as 'l' and not value '1', for some reason.

You are correct - the FWD code needs to convert the Java value to a BDT, and work with that.

Somewhere after line 8746 in ControlFlowOps, for the INPUT mode, if the first phase of c'tor lookup fails, and the argument is a Java type (Integer, Double, String, etc), convert this to a BDT instance re-apply the c'tors.

#4 - 01/26/2021 03:23 PM - Roger Borrello

Greg wants me to triage the other high priority tasks related to the customer's application, so I won't get to this until a little while.

I am curious about why it doesn't fail if the other line is in there?

#6 - 01/27/2021 11:27 AM - Roger Borrello

Are you referring to line number 8746 of 3821c_11955 version?

That version's line is:

```
                        if (bwCtor != null)
                        {
                           // in the event the OUTPUT value must be converted back:
                           FieldAssigner.update(((BaseDataType) param[i]),
                                                ((BaseDataType) newPars[i]),
                                                bwCtor);
                        }
                     }                  <-- line 8746
                     if (wa.em.isPending())
                     {
                        // if any errors occurred during parameter matching
                        // (eg.: INTEGER overflow) procedure cannot be called
                        ok = false;
                        break;
                     }
```

#7 - 01/27/2021 12:16 PM - Constantin Asofiei

Sorry, I had some changes in, the line is 8808 for 11955.

#8 - 01/27/2021 01:55 PM - Roger Borrello

From Constantin: To see the validation of the parameters occur, set a breakpoint in ControlFlowOps.valid at 8365 (if (value != null)) and set it to null in

the debugger. Then go through the validation loops to see where it fails.

#9 - 01/27/2021 05:26 PM - Roger Borrello

Constantin Asofiei wrote:

> I see what you mean now. I read the last argument as 'l' and not value '1', for some reason.
>
> You are correct - the FWD code needs to convert the Java value to a BDT, and work with that.
>
> Somewhere after line 8746 in ControlFlowOps, for the INPUT mode, if the first phase of c'tor lookup fails, and the argument is a Java type (Integer, Double, String, etc), convert this to a BDT instance re-apply the c'tors.

Notes from my debug session... At the start of the line 8667 switch, toCompType=com.goldencode.p2j.util.character) and fromCompType=com.goldencode.p2j.util.integer.

In Utils.findConstructor, looking for bdtConstructors.get("class com.goldencode.p2j.util.character#class com.goldencode.p2j.util.integer") we eventually get to parent=class com.goldencode.p2j.util.BaseDataType and public com.goldencode.p2j.util.character(com.goldencode.p2j.util.BaseDataType) is returned as a constructor.

At this point, it worked. The constructor was able to be used.

And now I cannot seem to duplicate the issue. I am trying several things, but no dice. I hadn't upgraded. Also, the customer application isn't throwing the error.

#10 - 01/27/2021 06:18 PM - Roger Borrello

Strangely enough, when I change the code from:

```
message dynamic-function('mydyn1', input 1, input 'int', input 1).
```

to

```
def var myc as char.
assign myc = dynamic-function('mydyn1', input 1, input 'int', input 1).
message myc.
```

the issue returns.

#11 - 01/27/2021 06:26 PM - Roger Borrello

Here is the broken Java:

```
        myc.assign(ControlFlowOps.invokeDynamicFunctionWithMode(character.class, "mydyn1", "III", 1, "int", 1
));
        message(myc);
```

Here is the working Java:

```
        message(ControlFlowOps.invokeDynamicFunctionWithMode(character.class, new character("mydyn1"), "III",
 new integer(1), new character("int"), new integer(1)));
```

So it looks like something in the conversion.

#12 - 01/27/2021 06:48 PM - Roger Borrello

Comparing the AST files, the version that doesn't work has this annotation in the dynamic-function node:

```
            <annotation datatype="java.lang.Boolean" key="wrap" value="true"/>
```

That seems to be the only major difference, aside from the obvious ones related to different code.

#13 - 01/27/2021 07:01 PM - Roger Borrello

What is the wrap annotation used for? It looks like it is used in may rules.

#14 - 01/28/2021 06:52 AM - Constantin Asofiei

Roger, I agree, the literal should be wrapped to integer BDT class.

Add the -Drules.tracing=true JVM argument to your conversion command and see who sets the wrap annotation for the MESSAGE case.

#15 - 01/28/2021 10:32 AM - Roger Borrello

The culprit is convert/expressions.rules:

```
      <!-- value() construct is used to convert an expression (of any
           wrapper type) into a string -->
      <rule>type == prog.kw_value and numImmediateChildren > 0

         <!-- by default, return VALUE() expressions as strings -->
         <action>asString = true</action>

         <!-- change the default if explicitly specified -->
         <rule>isNote("as-string") and !getNoteBoolean("as-string")
            <action>asString = false</action>
         </rule>

         <!-- bypass remaining logic if not returning a string -->
         <rule>asString

            <action>ref = copy.getChildAt(0)</action>

            <!-- safety code (we always expect an expression node) -->
            <rule>ref.type == prog.expression

               <rule>ref.getChildAt(0).type != prog.string
                  <action>
                     grpid = createJavaAst(java.method_call, "toStringMessage", closestPeerId)
                  </action>
                  <action>createPeerAst(java.lparens, "", grpid)</action>
                  <action>ref.putAnnotation("wrap", true)</action>
               </rule>
            </rule>

         </rule>
      </rule>

      <!-- message content array with non-string children needs wrapping -->
      <rule>type == prog.expression and parent.type == prog.content_array
         <action>ref = copy.getChildAt(0)</action>
         <rule>ref.type != prog.string
            <action>ref.putAnnotation("wrap", true)</action>            <-- Location
         </rule>
      </rule>
```

#16 - 01/28/2021 11:02 AM - Greg Shah

That won't be the solution for the larger issue. It just highlights that the MESSAGE statement uses wrapping to make this kind of thing safe. Your changes will need to be somewhere else.

#17 - 01/28/2021 11:14 AM - Roger Borrello

You're probably correct about the "wrap" annotation, since I get the same error with this testcase:

```
function mydyn1 returns character (
   input pc1 as character ) :
   return "1".
end function.

def var myc as int no-undo.

assign myc = integer(dynamic-function('mydyn1', input 1)).

message "Done.".
```

Java:

```
myc.assign(new integer(ControlFlowOps.invokeDynamicFunctionWithMode(character.class, "mydyn1", "I", 1)));
```

And in the AST, there isn't a "wrap" annotation, but there is a "force_no_wrap=true" added by annotations/collect_parameters.rules. Would it be normal to always wrap parameters to a dynamic function, or is that susceptible to over utilization of memory?

```
          <ast col="0" id="17179869244" line="0" text="expression" type="EXPRESSION">
            <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
            <annotation datatype="java.lang.String" key="support_level-byrule" value="gaps/gap_analysis_markin
g.xml:434"/>
            <ast col="14" id="17179869245" line="8" text="integer" type="FUNC_INT">
              <annotation datatype="java.lang.Long" key="oldtype" value="1641"/>
              <annotation datatype="java.lang.Boolean" key="builtin" value="true"/>
              <annotation datatype="java.lang.Boolean" key="returnsunknown" value="true"/>
              <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
              <annotation datatype="java.lang.String" key="support_level-byrule" value="gaps/gap_analysis_mark
ing.xml:434"/>
              <annotation datatype="java.lang.Long" key="peerid" value="188978561147"/>
              <annotation datatype="java.lang.String" key="peerid-byrule" value="convert/builtin_functions.rul
es:1605"/>
              <ast col="22" id="17179869247" line="8" text="dynamic-function" type="FUNC_POLY">
                <annotation datatype="java.lang.Long" key="oldtype" value="621"/>
                <annotation datatype="java.lang.Boolean" key="builtin" value="true"/>
                <annotation datatype="java.lang.Boolean" key="returnsunknown" value="true"/>
                <annotation datatype="java.lang.Long" key="param_index" value="0"/>
                <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
                <annotation datatype="java.lang.String" key="support_level-byrule" value="gaps/gap_analysis_ma
rking.xml:434"/>
                <annotation datatype="java.lang.String" key="casttype" value="character"/>
                <annotation datatype="java.lang.String" key="casttype-byrule" value="annotations/functions.rul
es:590"/>
                <annotation datatype="java.lang.String" key="param_modes" value="I"/>
                <annotation datatype="java.lang.String" key="param_modes-byrule" value="annotations/collect_pa
rameters.rules:432"/>
                <annotation datatype="java.lang.Boolean" key="ignorecast" value="true"/>
                <annotation datatype="java.lang.String" key="ignorecast-byrule" value="convert/builtin_functio
ns.rules:667"/>
                <annotation datatype="java.lang.Long" key="peerid" value="188978561148"/>
                <annotation datatype="java.lang.String" key="peerid-byrule" value="convert/builtin_functions.r
ules:1605"/>
                <ast col="39" id="17179869249" line="8" text="'mydyn1'" type="STRING">
                  <annotation datatype="java.lang.Boolean" key="is-literal" value="true"/>
                  <annotation datatype="java.lang.Long" key="param_index" value="0"/>
                  <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
                  <annotation datatype="java.lang.String" key="support_level-byrule" value="gaps/gap_analysis_
```

```
marking.xml:434"/>
                  <annotation datatype="java.lang.Long" key="peerid" value="188978561150"/>
                  <annotation datatype="java.lang.String" key="peerid-byrule" value="convert/literals.rules:10
76"/>
                </ast>
                <ast col="0" id="17179869266" line="0" text="&amp;quot;I&amp;quot;" type="STRING">
                  <annotation datatype="java.lang.String" key="byrule" value="annotations/collect_parameters.r
ules:433"/>
                  <annotation datatype="java.lang.Boolean" key="force_no_wrap" value="true"/>
                  <annotation datatype="java.lang.String" key="force_no_wrap-byrule" value="annotations/collec
t_parameters.rules:439"/>
                  <annotation datatype="java.lang.Long" key="peerid" value="188978561151"/>
                  <annotation datatype="java.lang.String" key="peerid-byrule" value="convert/literals.rules:10
76"/>
                </ast>
                <ast col="55" id="17179869254" line="8" text="1" type="NUM_LITERAL">
                  <annotation datatype="java.lang.Boolean" key="is-literal" value="true"/>
                  <annotation datatype="java.lang.Long" key="param_index" value="1"/>
                  <annotation datatype="java.lang.Long" key="support_level" value="16400"/>
                  <annotation datatype="java.lang.String" key="support_level-byrule" value="gaps/gap_analysis_
marking.xml:434"/>
                  <annotation datatype="java.lang.Boolean" key="use64bit" value="false"/>
                  <annotation datatype="java.lang.String" key="use64bit-byrule" value="annotations/cleanup.rul
es:434"/>
                  <annotation datatype="java.lang.Long" key="parmtype" value="698"/>
                  <annotation datatype="java.lang.String" key="parmtype-byrule" value="annotations/collect_par
ameters.rules:233"/>
                  <annotation datatype="java.lang.Long" key="peerid" value="188978561152"/>
                  <annotation datatype="java.lang.String" key="peerid-byrule" value="convert/literals.rules:81
0"/>
                </ast>
```