Base Language - Bug #4480

Extent does not become fixed when there is an initial list of values

12/16/2019 12:32 PM - 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

#1 - 12/16/2019 12:45 PM - Roger Borrello

- File form_with_extent.p.ast added

Testcase uast/form_tests/form_with_extent.p

20191231 Update: Checked in 4207a-11365

```
define variable cV as character extent initial ["1", "2"].
form cV with frame frX.
display cV with frame frX.
message "Extent="extent(cV) " Done.".
```

The message is **Extent=2 Done.** when executed. Changing up the initial list changes the output accordingly.

AST output is attached

The Javac output is below, but most likely this is related to the conversion.

```
    [javac] ./testcases/uast/uast/buildarea/src/com/goldencode/testcases/abl/FormWithExtent.java:32: error: no
 suitable constructor found for Element(character[],FillInWidget)
    [javac]             new Element(cV, frXFrame.widgetCV())
    [javac]             ^
    [javac]     constructor Element.Element(Accessor,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to Accessor)
    [javac]     constructor Element.Element(BaseDataType,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to BaseDataType)
    [javac]     constructor Element.Element(String,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to String)
    [javac]     constructor Element.Element(int,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to int)
    [javac]     constructor Element.Element(double,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to double)
    [javac]     constructor Element.Element(boolean,GenericWidget) is not applicable
    [javac]       (argument mismatch; character[] cannot be converted to boolean)
    [javac] Note: Some messages have been simplified; recompile with -Xdiags:verbose to get full output
    [javac] 1 error
```

Constantin, do you have any pointers to where I should start?

#3 - 12/16/2019 01:53 PM - Roger Borrello

Roger Borrello wrote:

> # Testcase uast/form_tests/form_with_extent.p
>
> [...]
>
> The message is **Extent=2 Done.** when executed. Changing up the initial list changes the output accordingly.
>
> AST output is attached
>
> The Javac output is below, but most likely this is related to the conversion.
>
> [...]
>
> Constantin, do you have any pointers to where I should start?

The documentation is very explcit in the DEFINE VARIABLE area. There are 2 states to an indeterminate array variable, fixed or unfixed. An unfixed indeterminate array can be fixed in a few ways, one of which is to initialize the array values when you define the variable, using INITIAL. Once the variable is fixed, is is treated as a determinate array.

Could this be handled at the parsing point? KW_EXTENT matching currently does not look ahead to see if there's an INITIAL. Instead, it returns a size of -1.

#4 - 12/16/2019 02:05 PM - Constantin Asofiei

Roger, I agree, I think we need to 'fix' the extent when there is an initializer and no extent value. For variables, see def_var_stmt - initializer should return the 'fixed' extent size, and it should override the extent value, only if is not already set.

But: the same might apply to OO properties, table fields and procedure parameters. So please do some tests, and if this behavior exists, fix them, too.

#5 - 12/16/2019 02:26 PM - Roger Borrello

Constantin Asofiei wrote:

> Roger, I agree, I think we need to 'fix' the extent when there is an initializer and no extent value. For variables, see def_var_stmt - initializer should return the 'fixed' extent size, and it should override the extent value, only if is not already set.
>
> But: the same might apply to OO properties, table fields and procedure parameters. So please do some tests, and if this behavior exists, fix them, too.

I am thinking about adding some sort of return value to initializer_constant which would be the number of initializers that are processed. At least this would give def_var_stmt, temp_table_field_options, and any other calling functions a fighting chance to utilize it to set the extent (if not done explicitly).

#6 - 12/16/2019 03:50 PM - Greg Shah

> But: the same might apply to OO properties, table fields and procedure parameters. So please do some tests, and if this behavior exists, fix them, too.

In regard to table fields, can they really be indeterminate extents?

#7 - 12/17/2019 01:34 PM - Roger Borrello

Greg Shah wrote:

> > But: the same might apply to OO properties, table fields and procedure parameters. So please do some tests, and if this behavior exists, fix them, too.
>
> In regard to table fields, can they really be indeterminate extents?

Having worked with Greg on this, he indicated to add my new code (see below) in the functions that have calls to both size = extent and initcount = initializer[ftype]. I have already updated initializer to return a count.

```
         // Handle the situation where there may have been an initializer but no explicit extent
         // initcount != -1 and size == -1 would be indicative of this.
         size = (initcount != -1 && size == -1) ? initcount : size;
```

There is something very confusing about the use of size in def_var_stmt in that I don't see it actually being used to set an annotation. I walked through the debugger, and size is calculated correctly, but then just goes out of scope.

#8 - 12/19/2019 01:18 PM - Roger Borrello

Roger Borrello wrote:

> Greg Shah wrote:
>
> > But: the same might apply to OO properties, table fields and procedure parameters. So please do some tests, and if this behavior exists, fix them, too.
>
> In regard to table fields, can they really be indeterminate extents?

Having worked with Greg on this, he indicated to add my new code (see below) in the functions that have calls to both size = extent and initcount = initializer[ftype]. I have already updated initializer to return a count.

[...]

There is something very confusing about the use of size in def_var_stmt in that I don't see it actually being used to set an annotation. I walked through the debugger, and size is calculated correctly, but then just goes out of scope.

The updated ended up being in Variable.java, so updated to progress.g are unnecessary. However, the changes could be helpful to keep around, since the change to initializer would at least allow the option, should it be necessary. Let me know what to do with them.

#9 - 12/19/2019 01:42 PM - Roger Borrello

Testcases Converted and Compiled.

form_with_extent.p

```
define variable cV as character extent initial ["1", "2"].
form cV with frame frX.
display cV with frame frX.
message "Extent="extent(cV) " Done.".
```

define_vars_extent.p

```
define variable txt1  as character  extent 3.
define variable dat2  as date       extent 3.
define variable num3  as integer    extent 10  initial [ 14, 30 ].
define variable dat4  as date       extent 10  initial [ ?, 01/01/2000, 12/31/1999 ].
define variable txt5  as character  extent 3   initial [ "hello", "world", "!" ].
define variable bool6 as logical    extent 3   initial [ yes, no, true ].

if txt1[1] eq txt5[1] and dat2[1] eq dat4[1] and num3[1] > 100 and bool6[1] then
   message "nothing".
```

define_vars_extent_shared.p

```
define shared            variable txt1  as character  extent 3.
define new shared        variable dat2  as date       extent 3.
define new global shared variable dec3  as decimal    extent 3.
define shared            variable num4  as integer    extent 10  initial [ 14, 30 ].
define new shared        variable dat5  as date       extent 10  initial [ ?, 01/01/2000, 12/31/1999 ].
define new global shared variable txt6  as character  extent 10  initial [ "hello", "world" ].
define shared            variable txt7  as character  extent 3   initial [ "hello", "world", "!" ].
define new shared        variable bool8 as logical    extent 3   initial [ yes, no, true ].
define new global shared variable num9  as integer    extent 3   initial [ 0, 1, 14 ].

if txt1[1] eq txt7[1] and num4[1] > 100 then message "nothing".
```

dynamic_extent_input_output_param.p

```
function f0 returns int (input-output j as int extent 5).
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 0 0 0 0 */
  j = 10.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 10 10 10 10 */
end.
procedure p0.
  def input-output param j as int extent 5.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 0 0 0 0 */
  j = 10.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 10 10 10 10 */
end.
def var i as int extent.

run p0(input-output i).
message extent(i) i[1] i[2] i[3] i[4] i[5]. /* this will display 5 10 10 10 10 */

extent(i) = ?.

f0(input-output i).
message extent(i) i[1] i[2] i[3] i[4] i[5]. /* this will display 5 10 10 10 10 */
```

dynamic_extent_output_param.p

```
function f0 returns int (output j as int extent 5).
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 0 0 0 0 */
  j = 10.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 10 10 10 10 */
end.
procedure p0.
  def output param j as int extent 5.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 0 0 0 0 */
  j = 10.
  message extent(j) j[1] j[2] j[3] j[4] j[5]. /* this will display 5 10 10 10 10 */
end.
def var i as int extent.
```

```
run p0(output i).
message extent(i) i[1] i[2] i[3] i[4] i[5]. /* this will display 5 10 10 10 10 */

extent(i) = ?.

f0(output i).
message extent(i) i[1] i[2] i[3] i[4] i[5]. /* this will display 5 10 10 10 10 */
```

extent-function-parm.p

```
def var e as integer extent 4 no-undo.

function f1 returns decimal (input d as decimal):
   return d.
end.

f1(e[1]).
```

extent00.p

```
/* extent00.p - Extent handling test case 0 */

def var my-list as int extent 3 init [1, 2, 3].

form my-list[1].
display my-list[1].
```

extent01.p

```
/* extent01.p - Extent handling test case 1 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 1.

form my-list[i].
display my-list[1].
```

extent02.p

```
/* extent02.p - Extent handling test case 2 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 1.

form my-list[1].
display my-list[i].
```

extent03.p

```
/* extent03.p - Extent handling test case 3 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 1.
```

```
form my-list[1] my-list[i].
display my-list[i].
```

extent04.p

```
/* extent04.p - Extent handling test case 4 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 2.

form my-list[1] my-list[i].
display my-list[i] my-list[1].
```

extent05.p

```
/* extent05.p - Extent handling test case 5 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 2.

form my-list[1] my-list[i] my-list[i + 1].
display my-list[2] my-list[1] my-list[3].
```

extent06.p

```
/* extent06.p - Extent handling test case 6 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 2.

form my-list[1] my-list[i] my-list[i + 1].
display my-list[2] my-list[1] my-list[2 + 1].
```

extent07.p

```
/* extent07.p - Extent handling test case 7 */

def var my-list as int extent 3 init [1, 2, 3].
def var i as int init 2.

form my-list[1] my-list[i] my-list[i + 1] my-list[3].
display my-list[2] my-list[1] my-list[2 + 1].
```

extent08.p

```
def var i as int extent 4 init [1, 2, 3, 4].
def var idx as int.

do idx = 1 to 3 with frame m1:
   display i[idx] i[idx + 1] i[1] i[2] with title "M1" 3 down frame m1.
end.

pause.

do idx = 1 to 3 with frame m2 on error undo, leave:
```

```
   display i[1] i[idx] i[2] i[idx + 1] with title "M2" 3 down frame m2.
end.

pause.

do idx = 1 to 3 with frame m3 on error undo, leave:
   display i[1] i[2] i[idx] i[idx + 1] with title "M3" 3 down frame m3.
end.

pause.

do idx = 1 to 3 with frame m4:
   display i[idx] i[1] i[idx + 1] i[2] with title "M4" 3 down frame m4.
end.
```

extent_field_ioob.p

```
def temp-table tt field num as int extent 2.
def temp-table tt2 field f1 as int.

create tt.
tt.num[1] = 14.
tt.num[2] = 30.
release tt.

create tt2.
tt2.f1 = 30.
release tt2.

def var i as int init ? no-undo.
def var txt as char no-undo.

/* Note: Literals less than 1 or greater than the extent size */
/*       cannot be coded because that is a 4GL compile error; */
/*       This is true for both where clauses and for field    */
/*       dereferencing. */

/* complex sub-expression as unknown value as subscript returns unknown */
pause message "test 0".
find tt where num[integer((i + 40) / 2)] = ?.
txt = "test 0 = " + string(available tt).
put screen row 1 col 1 txt.

/* unknown value as subscript returns unknown */
pause message "test 1".
find tt where num[i] = ? no-error.
txt = "test 1 = " + string(available tt).
put screen row 2 col 1 txt.

/* no-error doesn't matter because unknown val as subscript is not an error */
pause message "test 2".
find tt where num[i] = ?.
txt = "test 2 = " + string(available tt).
put screen row 3 col 1 txt.

/* non-unknown comparisons must compare with unknown */
/* must use no-error here else the "not on file" error occurs */
pause message "test 3".
find tt where num[i] > 0 no-error.
txt = "test 3 = " + string(available tt).
put screen row 4 col 1 txt.

/* A true array index OOB is a stop condition! */
i = 0.
repeat on stop undo, leave
       on endkey undo, next
       on error undo, next:
   /* this will create an infinite loop on error or endkey */
   /* since it disables infinite loop protection */
   retry.
   pause message "test 4".
```

```
   find tt where num[i] = ?.
   pause message "You won't see me!".
   leave.
end.
txt = "test 4 = " + string(available tt).
put screen row 5 col 1 txt.

/* no-error doesn't matter because it is a stop, not an error */
repeat on stop undo, leave
       on endkey undo, next
       on error undo, next:
   /* this will create an infinite loop on error or endkey */
   /* since it disables infinite loop protection */
   retry.
   pause message "test 5".
   find tt where num[i] = ? no-error.
   pause message "You won't see me!".
   leave.
end.
txt = "test 5 = " + string(available tt).
put screen row 6 col 1 txt.

/* complex expression as OOB is a stop condition */
repeat on stop undo, leave
       on endkey undo, next
       on error undo, next:
   /* this will create an infinite loop on error or endkey */
   /* since it disables infinite loop protection */
   retry.
   pause message "test 6".
   find tt where num[integer((i + 40) / 2)] = ? no-error.
   pause message "You won't see me!".
   leave.
end.
txt = "test 6 = " + string(available tt).
put screen row 7 col 1 txt.

find first tt.

i = ?.

/* field dereferencing with unknown yields unknown */
pause message "test 7".
txt = "test 7 = " + string(tt.num[i] = ?).
put screen row 8 col 1 txt.

/* field dereferencing with sub-expression unknown yields unknown */
pause message "test 8".
txt = "test 8 = " + string(tt.num[integer((i + 40) / 2)] = ?).
put screen row 9 col 1 txt.

i = 0.

/* field dereferencing with invalid index is an ERROR */
repeat on stop undo, next
       on endkey undo, next
       on error undo, leave:
   /* this will create an infinite loop on error or endkey */
   /* since it disables infinite loop protection */
   retry.
   pause message "test 9".
   txt = "test 9 = ERROR".
   txt = "test 9 = " + string(tt.num[i]).
   pause message "You won't see me!".
   leave.
end.
put screen row 10 col 1 txt.

/* field dereferencing with sub-expression as invalid index is still an ERROR */
repeat on stop undo, next
       on endkey undo, next
       on error undo, leave:
   /* this will create an infinite loop on error or endkey */
   /* since it disables infinite loop protection */
   retry.
```

```
   pause message "test 10".
   txt = "test 10 = ERROR".
   txt = "test 10 = " + string(tt.num[integer((i + 40) / 2)]).
   pause message "You won't see me!".
end.
put screen row 11 col 1 txt.

find first tt.

/* complex sub-expression as unknown value as subscript returns unknown,
   causing no record to be found (record not on file) */
i = ?.
repeat on stop undo, next
       on endkey undo, next
       on error undo, leave:
   pause message "test 11".
   txt = "test 11 = ERROR".
   find tt2 where tt2.f1 = tt.num[integer((i + 40) / 2)].
   txt = "test 11 = " + string(available tt2).
end.
put screen row 12 col 1 txt.

/* out of bounds element index error in unrelated buffer is supressed in
   where clause processing, causing no record to be found (record not on
   file) */
i = 0.
repeat on stop undo, next
       on endkey undo, next
       on error undo, leave:
   pause message "test 12".
   txt = "test 12 = ERROR".
   find tt2 where tt2.f1 = tt.num[integer(i - 10)].
   txt = "test 12 = " + string(available tt2).
end.
put screen row 13 col 1 txt.
```

extent_lazy_load.p

```
def temp-table tt
         field val1 as integer
         field val2 as integer extent 10.

def var i as integer.

on F5 endkey.

do transaction on endkey undo,leave:

   repeat i = 1 to 3 transaction:
      create tt.
      tt.val1 = i.
   end.

   find first tt where tt.val1 = 2.

   pause message "press F5".
end.
```

extent_params_basic1.p

```
def input param sp1 as int extent 10.
def output param sp2 as int extent 20.
def input-output param sp3 as int extent 30.

function func0 returns int(input fp1 as int extent 10, output fp2 as int extent 20, input-output fp3 as int ex
tent 30).
end.
```

```
procedure proc0.
   def input param pp1 as int extent 10.
   def output param pp2 as int extent 20.
   def input-output param pp3 as int extent 30.
end.
```

extent_params_basic2.p

```
def input param sp1 as int extent.
def output param sp2 as int extent.
def input-output param sp3 as int extent.

function func0 returns int(input fp1 as int extent, output fp2 as int extent, input-output fp3 as int extent).
end.

procedure proc0.
   def input param pp1 as int extent.
   def output param pp2 as int extent.
   def input-output param pp3 as int extent.
end.
```

extent_params_field_func.p

```
def var i as int.
def var j as int extent.
def var ch as char extent 5.

def temp-table tt1 field f1 as int extent 5 field f2 as int.

function f1 returns int extent (input i as int extent, input j as int).
end.
function f2 returns int extent (output i as int extent, output j as int, output ch as char extent 5).
end.
function f3 returns int extent (input-output i as int extent, input-output j as int).
end.
function f4 returns int.
end.

find first tt1.
f1(input tt1.f1, input tt1.f2).
f2(output tt1.f1, output tt1.f2, output ch).
f3(input-output tt1.f1, input-output tt1.f2).

dynamic-function("f1", input tt1.f1, input tt1.f2).
dynamic-function("f2", output tt1.f1, output tt1.f2, output ch).
dynamic-function("f3", input-output tt1.f1, input-output tt1.f2).

tt1.f1 = f1(input tt1.f1, input tt1.f2).
tt1.f1 = f2(output tt1.f1, output tt1.f2, output ch).
tt1.f1 = f3(input-output tt1.f1, input-output tt1.f2).

tt1.f1 = dynamic-function("f1", input tt1.f1, input tt1.f2).
tt1.f1 = dynamic-function("f2", output tt1.f1, output tt1.f2, output ch).
tt1.f1 = dynamic-function("f3", input-output tt1.f1, input-output tt1.f2).

tt1.f1[1] = 1.
tt1.f1[i] = 2.

f1(input j, input i).
f2(output j, output i, output ch).
f3(input-output j, input-output i).

tt1.f2 = dynamic-function("f4").
```

extent_params_field_proc.p

```
def var i as int.

def temp-table tt1 field f1 as int extent 5 field f2 as int.

procedure p1.
   def input param i as int extent.
   def input param j as int.
end.
procedure p2.
   def output param i as int extent.
   def output param j as int.
end.
procedure p3.
   def input-output param i as int extent.
   def input-output param j as int.
end.

find first tt1.
run p1(input tt1.f1, input tt1.f2).
run p2(output tt1.f1, output tt1.f2).
run p3(input-output tt1.f1, input-output tt1.f2).

tt1.f1[1] = 1.
tt1.f1[i] = 2.
```

extent_params_promotion.p

```
def temp-table tt1 field f1 as int extent 5.

update tt1.f1 validate(input tt1.f1[1] = 2, "aaa") with frame ft.

def var i as int extent 5.
def var k as int.
def var k2 as int.

function f0 returns int. end.

procedure proc0.
   def output param j as int extent 5.
   def input param k as int.

   update j validate(j[1] = j[2] + k, "a") with frame f1.

   do k = 1 to j[1] + f0() + k:
      message k.
   end.

   run proc0(output j, input k).
end.

procedure proc1.
   def output param j as int extent 5.
   def input param k as int.

   update j validate(input j[1] = input j[2] + k, "b") with frame f1.

   do k = 1 to j[1] + f0() + k:
      message k.
   end.

   run proc0(output j, input k).
end.

function func0 returns int(output j as int extent, input k as int).
   extent(j) = 2.
   do k = 1 to j[1] + f0() + k:
      message k.
   end.
```

```
   func0(output j, input k).
   dynamic-function("func0", output j, input k).
end.

function func1 returns int(output j as int extent, input k as int).
   extent(j) = 2.
   do k = 1 to j[1] + f0() + k:
      message k.
   end.
   func1(output j, input k).
   dynamic-function("func1", output j, input k).
end.

function func3 returns int extent.
end.

function func4 returns int(input j as int extent 5).
   func4(input func3()).
  /*dynamic-function("func4", input func3()).*/
end.

procedure proc2.
   def input param j as int extent 5.
   run proc2(input func3()).
end.

update i validate(i[1] = i[2], "b") with frame f2.
update i[1] validate(i[1] = i[2], "bb") with frame f2b.
update i validate(i[k] = 2, "c") with frame f2c.
update i validate(input i[k] = 2, "d") with frame f2d.
update k validate(k = 3, "c") with frame f3.
update k validate(k = 3 and i[1] = i[2], "d") with frame f4.

do k = 1 to i[1] + f0() + k:
   message k.
end.

func0(output i, input k).
dynamic-function("func0", output i, input k).
run proc0(output i, input k).

update k validate(k2 = 1, "aaa") k2 with frame f5.
update k validate(k2 = 1, "bbb") with frame f6.
```

extent_var_assignment.p

```
define variable num1  as integer    extent 10  initial [ 14 ].
define variable num2  as integer    extent 10  initial [ 30 ].

num1[1] = ?.
num1[2] = num2[1].
num2    = ?.
num2    = num1.
```

extent_var_def.p

```
def var t-date0  as date format "99/99/9999".
def var t-date1  as date format "99/99/9999"          init 01/01/-999.
def var t-date2  as date format "99/99/9999" extent 8 init 01/01/2004.
def var t-date3  as date format "99/99/9999" extent 8 init [ 01/01/2000, 02/02/2000 ].
def var t-date4  as date format "99/99/9999" extent 8 init [ 01/01/2000, 02/02/2000, 03/03/2000, 04/04/2000, 0
5/05/2000, 06/06/2000, 07/07/2000, 08/08/2000 ].
def var t-date5  as date format "99/99/9999" extent 8.
def var t-date6  as date format "99/99/9999" extent 8 init ?.

def var t-char0  as char format "x(20)".
```

```
def var t-char1  as char format "x(20)"               init "hello".
def var t-char2  as char format "x(20)"      extent 3 init "hello".
def var t-char3  as char format "x(20)"      extent 3 init ["hello", "goodbye"].
def var t-char4  as char format "x(20)"      extent 3 init ["hello", "goodbye", "hello again"].
def var t-char5  as char format "x(20)"      extent 3.
def var t-char6  as char format "x(20)"      extent 8 init ?.
```

no_extent_tests.p

```
def var i1 as int extent.
def var i2 as int extent 5.
def var i3 as int extent 0.

def temp-table tt1
   field f1 as int
   field f2 as int extent 5
   field f3 as int extent 0.

def var j1 like i1     extent 0.
def var j2 like i2     extent 0.
def var j3 like i3     extent 0.
def var j4 like tt1.f1 extent 0.
def var j5 like tt1.f2 extent 0.
def var j6 like tt1.f3 extent 0.

def temp-table tt2
   field f1 like tt1.f1 extent 0
   field f2 like tt1.f2 extent 0
   field f3 like tt1.f3 extent 0
   field f4 like i1     extent 0
   field f5 like i2     extent 0
   field f6 like i3     extent 0.

def var k1  like i1.
def var k2  like i2.
def var k3  like i3.
def var k4  like j1.
def var k5  like j2.
def var k6  like j3.
def var k7  like j4.
def var k8  like j5.
def var k9  like j6.
def var k10 like tt1.f1.
def var k11 like tt1.f2.
def var k12 like tt1.f3.
def var k13 like tt2.f1.
def var k14 like tt2.f2.
def var k15 like tt2.f3.
def var k16 like tt2.f4.
def var k17 like tt2.f5.
def var k18 like tt2.f6.

def temp-table tt3
   field f1  like tt1.f1
   field f2  like tt1.f2
   field f3  like tt1.f3
   field f4  like tt2.f1
   field f5  like tt2.f2
   field f6  like tt2.f3
   field f7  like tt2.f4
   field f8  like tt2.f5
   field f9  like tt2.f6
   /*field f10 like i1*/
   field f11 like i2
   field f12 like i3
   field f13 like j1
   field f14 like j2
   field f15 like j3
   field f16 like j4
   field f17 like j5
   field f18 like j6.
```

```
k1 = 0.
k2 = 0.
k3 = 0.
k4 = 0.
k5 = 0.
k6 = 0.
k7 = 0.
k8 = 0.
k9 = 0.
k10 = 0.
k11 = 0.
k12 = 0.
k13 = 0.
k14 = 0.
k15 = 0.
k16 = 0.
k17 = 0.
k18 = 0.
```

parameterized-extents-in-where.p

```
def var i1 as int no-undo.
def var i2 as int no-undo.

assign i1 = 1
       i2 = 4
       .

for each person
   where schedule[i1] <> "" and schedule[i2] = "Thursday":
  display person.
end.
```

trash_in_like_extent_00.p

```
def var array  as int extent 10 init [1, 2, 3, 4, 5, 6, 7, 8, 9, 10].
def var scalar like array[trash].
def var other  like array.
def var txt    as char.
def var i      as int.

/* to print the array we dump the contents into a string */
do i = 1 to extent(other):

   if i > 1 then
      txt = txt + ", ".
   else
      txt = "[".

   txt = txt + string(other[i]).
end.

txt = txt + "]".

message "other extent = " + string(extent(other)) + ", value = " + txt + "; " +
        "scalar extent = " + string(extent(scalar)) + ", value = " + string(scalar) + ";".
```

#10 - 12/19/2019 01:47 PM - Roger Borrello

Code checked into 4207a-11359.

#11 - 12/19/2019 01:47 PM - Roger Borrello

- Status changed from New to WIP

#12 - 12/20/2019 11:42 AM - Greg Shah

> The updated ended up being in Variable.java, so updated to progress.g are unnecessary. However, the changes could be helpful to keep around, since the change to initializer would at least allow the option, should it be necessary. Let me know what to do with them.

The changes are safe right? I think you can keep them there.

#13 - 12/20/2019 12:34 PM - Roger Borrello

Greg Shah wrote:

> > The updated ended up being in Variable.java, so updated to progress.g are unnecessary. However, the changes could be helpful to keep around, since the change to initializer would at least allow the option, should it be necessary. Let me know what to do with them.
>
> The changes are safe right? I think you can keep them there.

They are. Checked in progress.g: 4207a-11365.

#14 - 12/31/2019 11:59 AM - Greg Shah

- % Done changed from 0 to 100

- Status changed from WIP to Test

#15 - 03/03/2020 02:50 PM - Roger Borrello

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

#16 - 03/04/2020 10:30 AM - Greg Shah

- Status changed from Test to Closed

Files

form_with_extent.p.ast	19.8 KB	12/16/2019	Roger Borrello