Base Language - Bug #4766

fix CHR and ASC

07/15/2020 02:48 PM - Greg Shah

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	**version:**	

Description

Related issues:

Related to Base Language - Feature #6428: implement IS-LEAD-BYTE() built-in f...	**New**

History

#1 - 07/15/2020 02:49 PM - Greg Shah

As noted during #4384 work, Marian's team found that CHR and ASC were not double byte enabled, they use the wrong default codepage and validation is incomplete. This task is for tracking the cleanup work on those built-ins.

#2 - 04/06/2021 07:52 AM - Greg Shah

Please see #4761 for more details.

#3 - 04/09/2021 03:24 PM - Ovidiu Maxiniuc

Some work has been done and now 3821c/12257 has improved support for CHR and ASC functions. Both of them will handle 'normal' parameters as expected (tested with UTF-8, 8859-1, 8859-15, 1252). The problem are the exceptions, because 4GL is a bit chaotic. Here are some issues I discovered:

- if a character code is not defined/supported for 8859-x or 1252 a non-empty character is still returned. In the case of UTF-8, an empty string will be returned instead;
- the functions are multi-byte, meaning that they will successfully encode/decode characters like € in UTF-8 which occupies 3 bytes (14844588 / 0x00E282AC);
- a funny thing: running
  message asc("www", "UTF-8", "UTF-8").
  will print 7829367. In hexadecimal this number is 0x00777777. Now, looking at character map we can see that asc("w") = 119 = 0x77. Running
  message chr(7829367, "UTF-8", "UTF-8")
  will print back www! Of course, this is valid not only for w character, but seems limited to multi-byte codepages.
- chr(-1, "1252", "1252") will actually return a value but it is meaningless (ÿÿÿÿÿÿÿÿ1). Also it seems to alter over time;
- executing chr(188, "ISO8859-15", "ISO8859-15") with the default CPINTERNAL of "ISO8859-1" will print ¼. However, ¼ is not part of the target CP "ISO8859-15", Œ should have been printed instead. But this character is not part of the CPINTERNAL, so the character with same code is printed.
- "1252" defines at code 140 the character Œ (\u0152). The same character can be found in 8859-15 at position 188 (as noted in previous item). Theoretically the character should have been printed by
  message chr(140, "ISO8859-15", "1252").
  But it is not, instead errors 6063 and 1586 are issued.

#4 - 04/09/2021 04:32 PM - Greg Shah

> chr(-1, "1252", "1252") will actually return a value but it is meaningless (ÿÿÿÿÿÿÿÿ1). Also it seems to alter over time;

Yikes! I wonder if this is memory overflow/underflow problem. If the 4GL directly adds the value (e.g. -1) to a C/C++ pointer (memory address) then a negative value might be looking outside of the conversion tables.

Unless this proves to be stable in some way that an application can use, we will probably consider this an unimplemented "quirk".

#5 - 09/07/2021 07:06 AM - Marian Edu

This is probably still a work in progress but just mentioned this here since we've found some of our tests in OO implementation were failing and that turned out to be because of some CHR related changes. Previously the CHR returned empty string, as it should or at least this is what 4GL does, now it returns space (%20). There was previously a condition in I18nOps that returned null for codes less or equal to zero, that was removed - not sure about the high watermark test (65535) though.

#6 - 09/07/2021 08:14 AM - Ovidiu Maxiniuc

Marian, please provide the (isolated) testcases you refer to in note #4766-5. Please specify the active CP you are working with.

#7 - 09/07/2021 08:26 AM - Ovidiu Maxiniuc

Marian Edu wrote:

> [...] There was previously a condition in I18nOps that returned null for codes less or equal to zero, that was removed

It was not, see I18nOps.java:570

```
   private static String get4glCharacter(int ascCode, Charset charset)
   {
      if (ascCode < 0)
      {
         return null;
      }
```

> not sure about the high watermark test (65535) though.

it returns the empty string. I do not think the new implementation will return a space except when the parameter is 0x20 / 32dec.

#8 - 09/07/2021 08:39 AM - Marian Edu

Ovidiu Maxiniuc wrote:

> Marian Edu wrote:
>
> > [...] There was previously a condition in I18nOps that returned null for codes less or equal to zero, that was removed
>
> It was not, see I18nOps.java:570

The test there is for less than zero, I was (trying) to refer to the case when the code number is actually zero, sorry for not being clear in my message :(

CHR gives " " (space) now while in 4GL is "" (empty).

#9 - 09/07/2021 09:12 AM - Ovidiu Maxiniuc

Nice catch. The problem is not actually in I18nOps (which converts chr(0) to "\0") but in character constructor. Before assigning the value to its internal value, it is a bit processed by Text.javaSpacifyNull(). Apparently the instances of character data type are *spacified* in 4GL. Since there is a single \0 character, it will be converted to space. This is new for me. I will address this issue in the next commit.

#10 - 09/07/2021 04:09 PM - Ovidiu Maxiniuc

The fix for chr(0) was committed in revision 12898/3821c.

#11 - 09/08/2021 10:08 AM - Greg Shah

Code Review Task Branch 3821c Revision 12898

The changes look good.

#12 - 04/05/2023 12:43 PM - Greg Shah

What is left to do in this task? Please note that in #6428 Joe is fixing an issue related to lead-byte processing in CHR. I don't know what else is needed but would like to list the items here.

#13 - 04/05/2023 12:43 PM - Greg Shah

- Related to Feature #6428: implement IS-LEAD-BYTE() built-in function added