isCOBOL^{TM} Evolve

isCOBOL Evolve 2014 Release 1 Overview

isCOBOL Evolve 2014 Release 1 Overview

Introduction

Veryant is pleased to announce the latest release of isCOBOL™ Evolve, isCOBOL Evolve 2014 R1.

isCOBOL Evolve provides a complete environment for the development, deployment, maintenance, and modernization of COBOL applications. isCOBOL 2014 R1 includes a new product called isCOBOL EIS (Enterprise Information System), several enhancements to the isCOBOL IDE and the Utilities and several optimizations in the isCOBOL Server product; isCOBOL 2014 R1 also includes many Debugger improvements, and other new features.

Details on these enhancements and updates are included below.

isCOBOL EIS (Enterprise Information System)

isCOBOL Enterprise Information System (EIS) is an umbrella of tools and features available in the isCOBOL Evolve Suite for the development and execution of web based applications in a J2EE container. There are several options and features included in isCOBOL EIS:

- COBOL REST producers™ (server-side)

  This option allows the writing of a REST Web Service in COBOL. The COBOL program has to be transformed in a WebService REST stateful. This objective is achieved through the *HTTPHandler* class that allows communicating with HTML pages retrieving data and printing results. The features to be consumed could be implemented using COBOL ENTRY points. The data can be passed using Plain text, JavaScript Object Annotation (JSON) or eXtensible Markup Language (XML) stream.

  Code Snippet using JSON stream:

```
repository.
    class web-area as "com.iscobol.rts.HTTPHandler"
working-storage section.
01  isfunction-getZipCode identified by "".
    03 identified by "get_Zip_Code".
       05 city-zipCode  pic x any length.
01  isfunction-returnZipCode identified by "".
    03 identified by "Zip_Code".
       05 returnZipCode  pic x any length.
linkage section.
01  comm-area object reference web-area.
procedure division using comm-area.
GETZIPCODE.
    entry "ISFUNCTION_GETZIP" using comm-area.
    comm-area:>accept (isfunction-getZipCode).
    move 1 to idx.
    search array-data varying idx
       at end
          move "Zip code not Found"  to returnZipCode
       when city-zipCode = a-city(idx)
          move a-zipcode(idx)  to returnZipCode
    end-search.
    comm-area:>displayJSON (isfunction-returnZipCode).
    goback.
```

- COBOL REST consumers™ (client-side)

  This option allows consuming REST Web Services from COBOL client programs. This objective is achieved through the *HTTPClient* class.

  Code Snippet to consume the previous COBOL REST Web Service:

```
repository.
    class http-client as "com.iscobol.rts.HTTPClient"
    class http-param  as "com.iscobol.rts.HTTPData.Params".
working-storage section.
77  http object reference http-client.
77  params object reference http-param.
01  response-code pic S9(3).
01  isfunction-receivedZipCode identified by "".
    03 identified by "Zip_Code".
       05 zipCode  pic x any length.
procedure division.
    http:>doGet (function-url, params)
    http:>getResponseCode (response-code);;
    if response-code = 200
       http:>getResponseJSON (isfunction-receivedZipCode)
       display "The zip code is " zipCode.
```

  The Working structure is the way to describe the parameters received or passed to the Web Service in the Cobol program. So we can define different structures to be used for different purposes.

  The following working storage structures allow communication with OAuth (Open standard for Authorization) authentication REST Web Services provided by Facebook and Google or to access Twitter's APIs.

  Code Snippet for structure used by Facebook and Google:

```
01  user-info identified by "_".
    03 identified by "id".
       05 user-id  pic x any length.
    03 identified by "email".
       05 user-email  pic x any length.
    03 identified by "verified".
       05 user-verified-email  pic x any length.
    03 identified by "name".
       05 user-name  pic x any length.
    03 identified by "first_name".
       05 user-given-name  pic x any length.
    03 identified by "last_name".
       05 user-family-name  pic x any length.
    03 identified by "link".
       05 user-link  pic x any length.
    03 identified by "picture".
       05 user-picture  pic x any length.
    03 identified by "gender".
       05 user-gender  pic x any length.
```

Code Snippet for TWITTER's structures:

```
01  twitter-auth identified by "".
    03 identified by "token_type".
       05 token-type  pic x any length.
    03 identified by "access_token".
       05 access-token  pic x any length.
01  twitter identified by "root".
    03 identified by "errors" occurs dynamic capacity err-cnt.
       05 identified by "message".
          07 error-mesg pic x any length.
       05 identified by "code".
          07 error-code pic x any length.
    03 array identified by "element" occurs dynamic capacity c.
       05 identified by "text".
          07 twittext pic x any length.
       05 identified by "user".
          07 identified by "screen_name".
             09 screen-name pic x any length.
```

- COBOL SOAP consumers™ (client-side)

  The *HTTPClient* class also allows for consuming SOAP Web Services. To simplify the complexity of the SOAP structure, a new utility named *WSDL2Wrk* is provided to generate the input and output structures that define the XML streams.

  Code Snippet of structures generated in the copy file:

```
*> binding name=IP2GeoSoap, style=
 01  soap-ResolveIP-input identified by 'Envelope'
          namespace 'http://www.w3.org/2003/05/soap-envelope'.
     03 identified by 'Body'.
         06 identified by 'ResolveIP'
            namespace 'http://ws.cdyne.com/'.
          07 identified by 'ipAddress'.
           08 a-ipAddress pic x any length.
          07 identified by 'licenseKey'.
           08 a-licenseKey pic x any length.
 01  soap-ResolveIP-output identified by 'Envelope'
          namespace 'http://www.w3.org/2003/05/soap-envelope'.
     03 identified by 'Body'.
         06 identified by 'ResolveIPResponse'
            namespace 'http://ws.cdyne.com/'.
          07 identified by 'ResolveIPResult'.
           08 identified by 'City'.
            09 a-City pic x any length.
           08 identified by 'StateProvince'.
            09 a-StateProvince pic x any length.
           08 identified by 'Country'.
            09 a-Country pic x any length.
           08 identified by 'Organization'.
            09 a-Organization pic x any length.
           08 identified by 'Latitude'.
            09 a-Latitude pic x any length.
           08 identified by 'Longitude'.
            09 a-Longitude pic x any length.
           08 identified by 'AreaCode'.
            09 a-AreaCode pic x any length.
           08 identified by 'TimeZone'.
            09 a-TimeZone pic x any length.
           08 identified by 'HasDaylightSavings'.
            09 a-HasDaylightSavings pic x any length.
           08 identified by 'Certainty'.
            09 a-Certainty pic x any length.
           08 identified by 'RegionName'.
            09 a-RegionName pic x any length.
           08 identified by 'CountryCode'.
            09 a-CountryCode pic x any length.
```

- COBOL Servlet™ (HTTP GET/POST management)

  This option allows writing a Servlet in COBOL using the same *HTTPHandler* class. Similar to a CGI program, a COBOL Servlet™ takes requests from a client such as a Web browser, then accesses data, applies business logic, and returns the results back to the Web browser. The COBOL servlet is loaded and executed by the Web server (Servlet container), and the client communicates with the COBOL servlet through the Web server using HTTP requests. HTTP GET/POST requests are automatically managed from the *HTTPHandler* accept method.

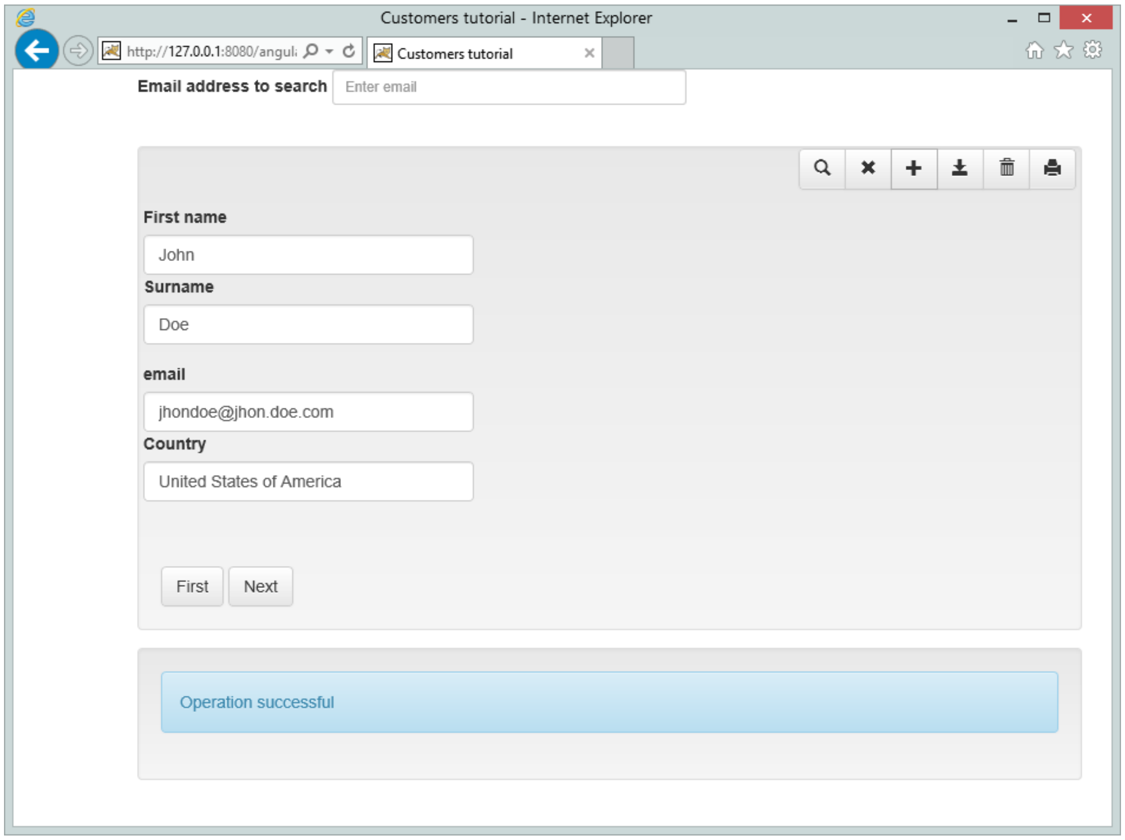
  Code Snippet of HTTP GET/POST management:

```
REPOSITORY.
    class web-area as "com.iscobol.rts.HTTPHandler".
WORKING-STORAGE SECTION.
01  request-page identified by "request".
    03 filler identified by "fld1".
       05 form-fld1 pic x any length.
    03 filler identified by "fld2".
       05 form-fld2 pic x any length.
01  response-page identified by "html".
    03 filler identified by "body".
       05 filler pic x(16) value "Text parameter: ".
       05 fld1   pic x any length.
       05 filler identified by "BR".
          07 pic x any length.
       05 filler pic x(11) value "File name: ".
       05 fld2 pic x any length.
LINKAGE SECTION.
01  comm-area object reference web-area.
PROCEDURE DIVISION USING comm-area.
main.
    comm-area:>accept (request-page).
    move form-fld1 to fld1.
    move form-fld2 to fld2.
    comm-area:>displayHTML (response-page null).
```

  Also the COBOL Servlet technology was developed to improve and replace easily existing CGI COBOL programs. See an example provided in the standard isCOBOL EIS distribution.

With isCOBOL EIS taking advantage of COBOL REST producer™ and JSON COBOL integration, it's also possible to write a Rich GUI Client Desktop application based on HTML5 and CSS3. Figure 1, *UI with AngularJS*, is an example of the UI obtained using AngularJS by Google.

Figure 1. UI with AngularJS

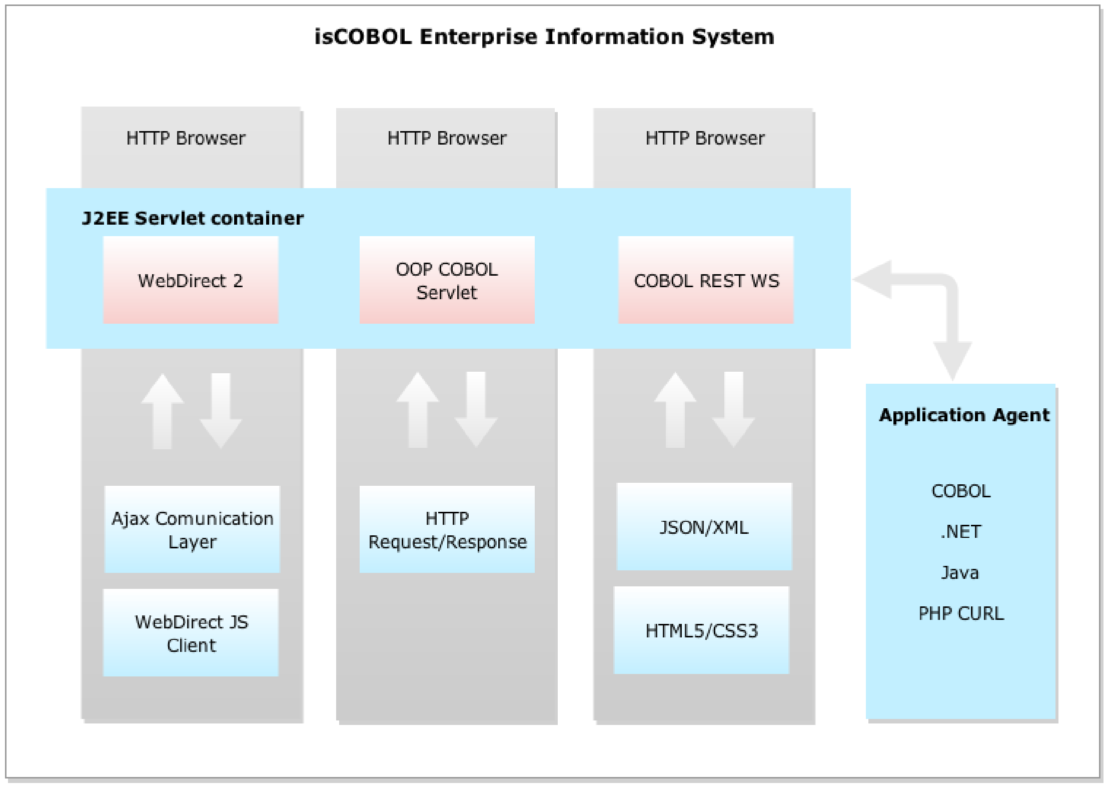

Customers tutorial - Internet Explorer
http://127.0.0.1:8080/anguli
Customers tutorial
Email address to search
Enter email
First
Next

First name: John
Surname: Doe
email: jhondoe@jhon.doe.com
Country: United States of America

Operation successful

Now, as shown in Figure 2, *isCOBOL EIS architecture*, isCOBOL WebDirect 2.0 is part of isCOBOL EIS.

Figure 2. isCOBOL EIS architecture



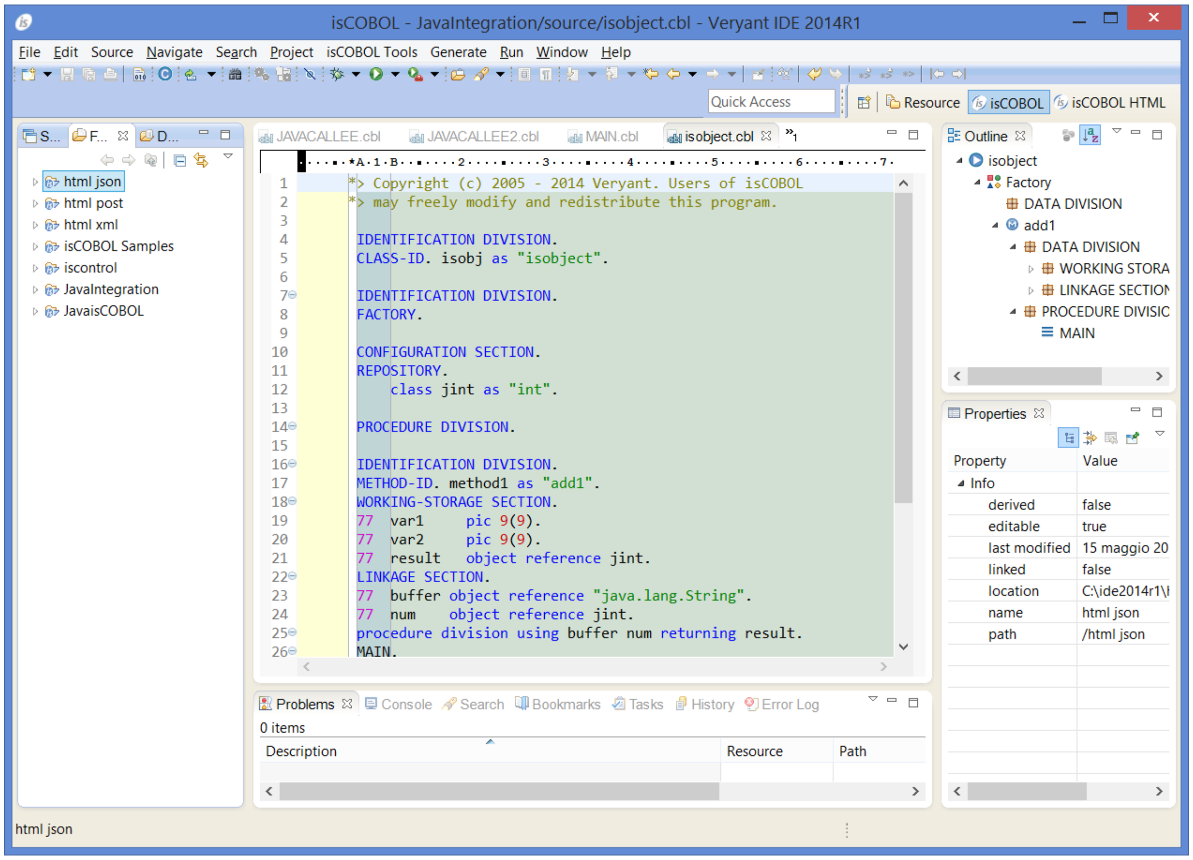
isCOBOL IDE Enhancements

In isCOBOL Evolve 2014 R1, the isCOBOL IDE has been created on the top of Eclipse KEPLER 4.3 to provide an state-of-the-art Integrated Developer Environment. Starting from isCOBOL 2014R1, the isCOBOL IDE will be available also for Linux and Apple Mac OSX.

New Look for the workbench

As shown on Figure 3, *isCOBOL IDE new look for the workbench*, the default appearance of the isCOBOL IDE user interface has been refreshed. The goals of this update were to adopt a more modern visual style, reduce clutter, and use whitespace rather than extra keylines to separate user interface elements. Tabs and part stacks have been redesigned to reduce emphasis on non-selected tabs and stacks.

Figure 3. isCOBOL IDE new look for workbench



```
*> Copyright (c) 2005 - 2014 Veryant. Users of isCOBOL
*> may freely modify and redistribute this program.

IDENTIFICATION DIVISION.
CLASS-ID. isobj as "isobject".

IDENTIFICATION DIVISION.
FACTORY.

CONFIGURATION SECTION.
REPOSITORY.
    class jint as "int".

PROCEDURE DIVISION.

IDENTIFICATION DIVISION.
METHOD-ID. method1 as "add1".
WORKING-STORAGE SECTION.
77  var1     pic 9(9).
77  var2     pic 9(9).
77  result   object reference jint.
LINKAGE SECTION.
77  buffer object reference "java.lang.String".
77  num    object reference jint.
procedure division using buffer num returning result.
MAIN.
```

Property	Value
Info	
derived	false
editable	true
last modified	15 maggio 20
linked	false
location	C:\ide2014r1\
name	html json
path	/html json

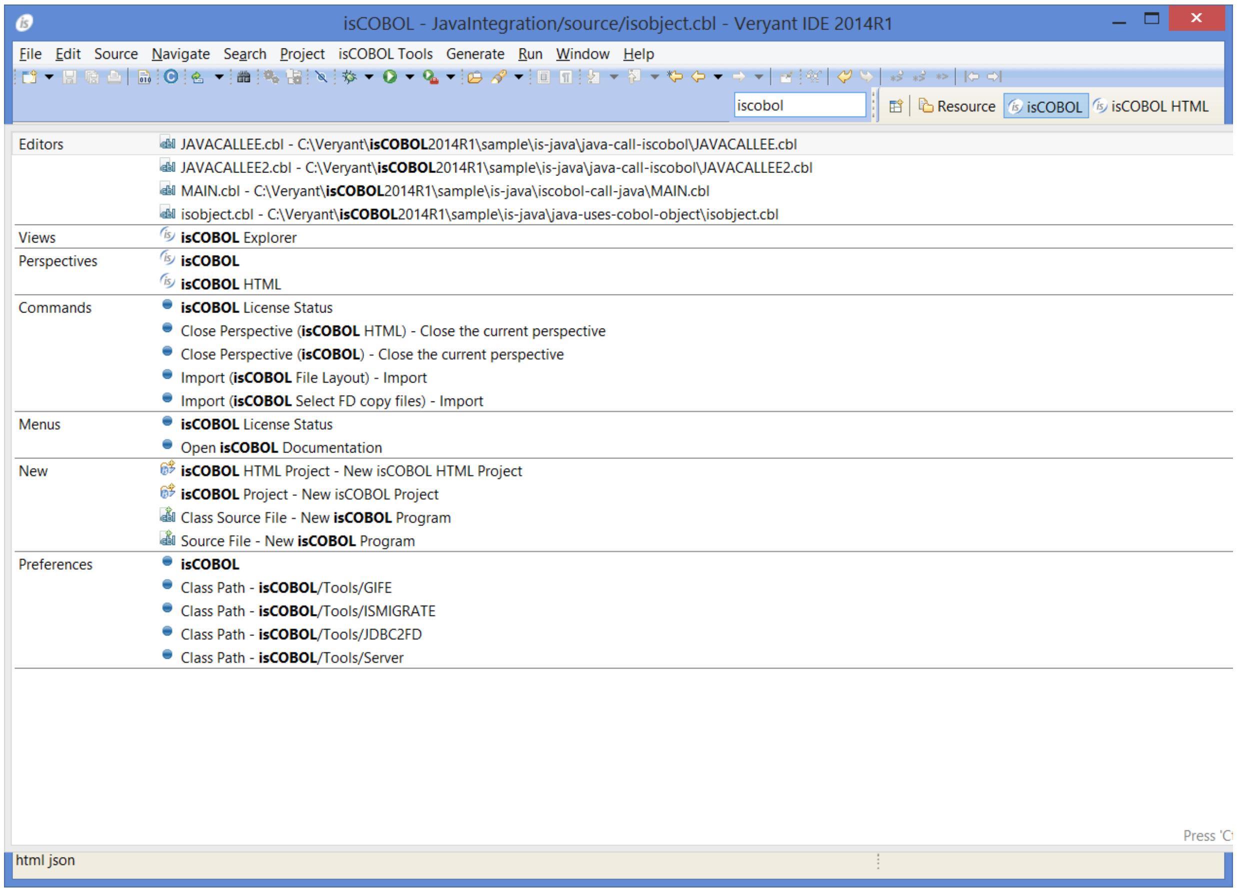
0 items

Description	Resource	Path

Global Search bar

The isCOBOL IDE now has a global search field in the tool bar. As shown in Figure 4, *Global search bar*, this provides a visible affordance for the "Quick Access" feature, and may be used in the future to combine other kind of searches into a single location.

Figure 4. Global search bar

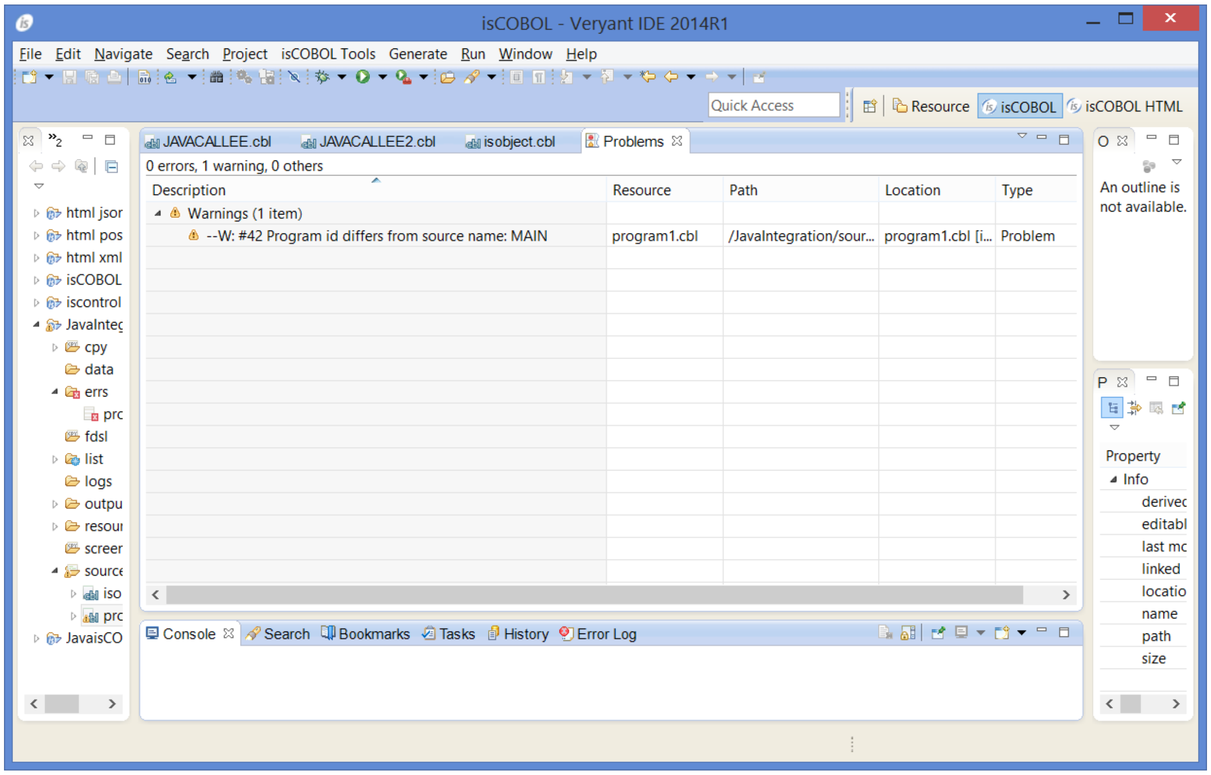

isCOBOL - JavaIntegration/source/isobject.cbl - Veryant IDE 2014R1
File Edit Source Navigate Search Project isCOBOL Tools Generate Run Window Help
iscobol
Resource
isCOBOL
isCOBOL HTML
Press 'C
html json

Editors	JAVACALLEE.cbl - C:\Veryant\isCOBOL2014R1\sample\is-java\java-call-iscobol\JAVACALLEE.cbl
	JAVACALLEE2.cbl - C:\Veryant\isCOBOL2014R1\sample\is-java\java-call-iscobol\JAVACALLEE2.cbl
	MAIN.cbl - C:\Veryant\isCOBOL2014R1\sample\is-java\iscobol-call-java\MAIN.cbl
	isobject.cbl - C:\Veryant\isCOBOL2014R1\sample\is-java\java-uses-cobol-object\isobject.cbl
Views	**isCOBOL** Explorer
Perspectives	**isCOBOL**
	isCOBOL HTML
Commands	**isCOBOL** License Status
	Close Perspective (**isCOBOL** HTML) - Close the current perspective
	Close Perspective (**isCOBOL**) - Close the current perspective
	Import (**isCOBOL** File Layout) - Import
	Import (**isCOBOL** Select FD copy files) - Import
Menus	**isCOBOL** License Status
	Open **isCOBOL** Documentation
New	**isCOBOL** HTML Project - New isCOBOL HTML Project
	isCOBOL Project - New isCOBOL Project
	Class Source File - New **isCOBOL** Program
	Source File - New **isCOBOL** Program
Preferences	**isCOBOL**
	Class Path - **isCOBOL**/Tools/GIFE
	Class Path - **isCOBOL**/Tools/ISMIGRATE
	Class Path - **isCOBOL**/Tools/JDBC2FD
	Class Path - **isCOBOL**/Tools/Server

More flexible part layout

As shown in Figure 5, *Flexible part layout*, you can now stack views and editors together by default. For example a view that requires a large working area can be stacked in the editor area to provide more spaces. Not only can you mix views and editors, but also you can split the editor area to put a view beside an editor in the editor area, and then maximize the entire editor area to work with both at the same time.

Figure 5. Flexible part layout



0 errors, 1 warning, 0 others

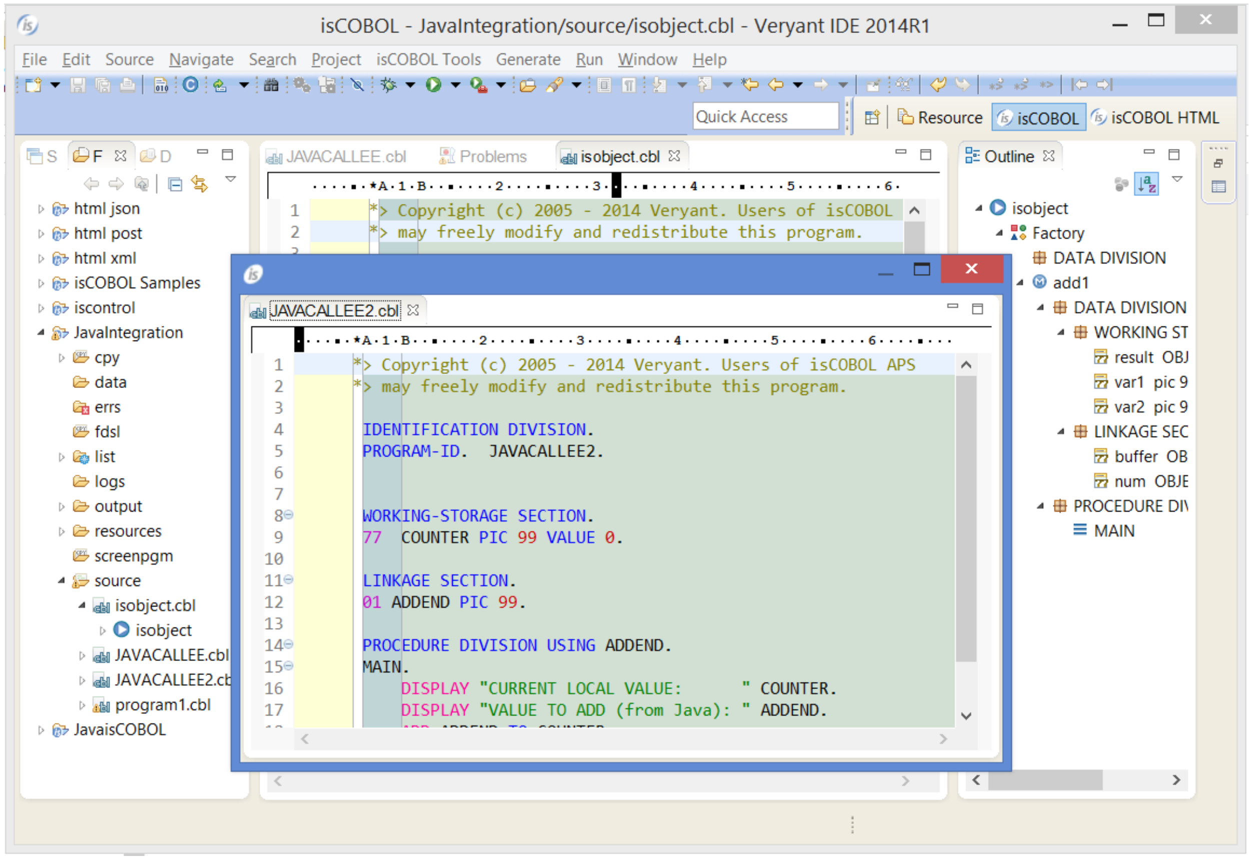
Description	Resource	Path	Location	Type
Warnings (1 item)				
--W: #42 Program id differs from source name: MAIN	program1.cbl	/JavaIntegration/sour...	program1.cbl [i...	Problem

An outline is not available.

Detached editors

The Figure 6, *Detached editors*, shows how an editors can now be detached into its own windows. The detached editor will not be treated any differently from an editor contained within the workbench window.

Figure 6. Detached Editors

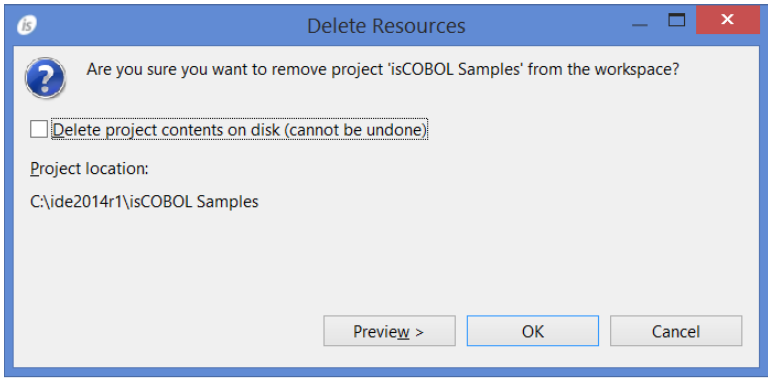


```
*> Copyright (c) 2005 - 2014 Veryant. Users of isCOBOL
*> may freely modify and redistribute this program.
```

```
*> Copyright (c) 2005 - 2014 Veryant. Users of isCOBOL APS
*> may freely modify and redistribute this program.

IDENTIFICATION DIVISION.
PROGRAM-ID.  JAVACALLEE2.


WORKING-STORAGE SECTION.
77  COUNTER PIC 99 VALUE 0.

LINKAGE SECTION.
01 ADDEND PIC 99.

PROCEDURE DIVISION USING ADDEND.
MAIN.
    DISPLAY "CURRENT LOCAL VALUE:      " COUNTER.
    DISPLAY "VALUE TO ADD (from Java): " ADDEND.
```

Delete project dialog shows project location

The Delete Resources dialog now shows the location of projects to be deleted on the file system:



Are you sure you want to remove project 'isCOBOL Samples' from the workspace?

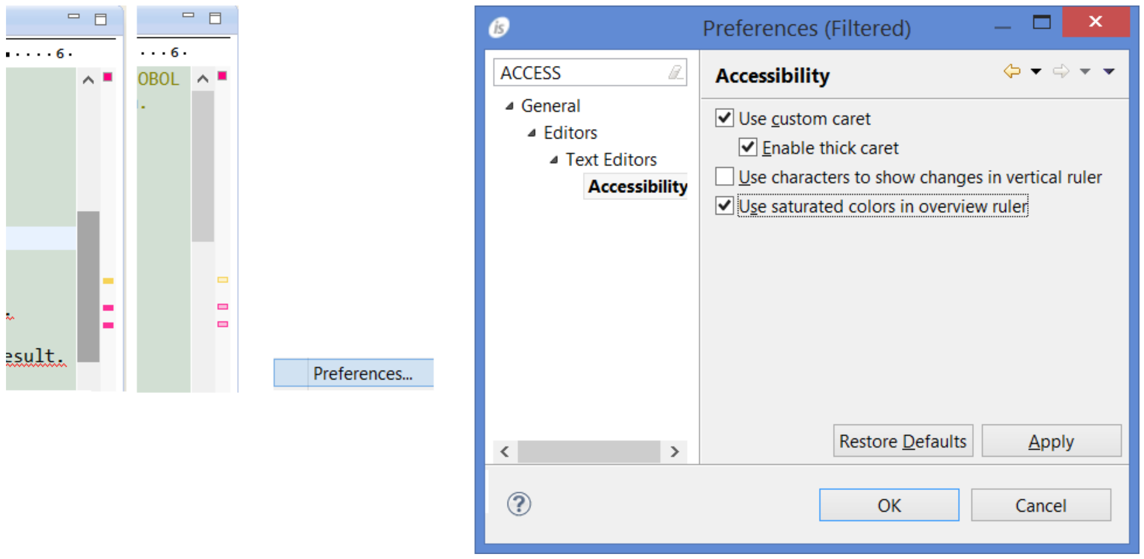
☐ Delete project contents on disk (cannot be undone)

Project location:

C:\ide2014r1\isCOBOL Samples

Use saturated colors in overview ruler

The new Use saturated colors in overview ruler preference allows showing the annotations in the Overview ruler using saturated colors:



Accessibility

☑ Use custom caret

☑ Enable thick caret

☐ Use characters to show changes in vertical ruler

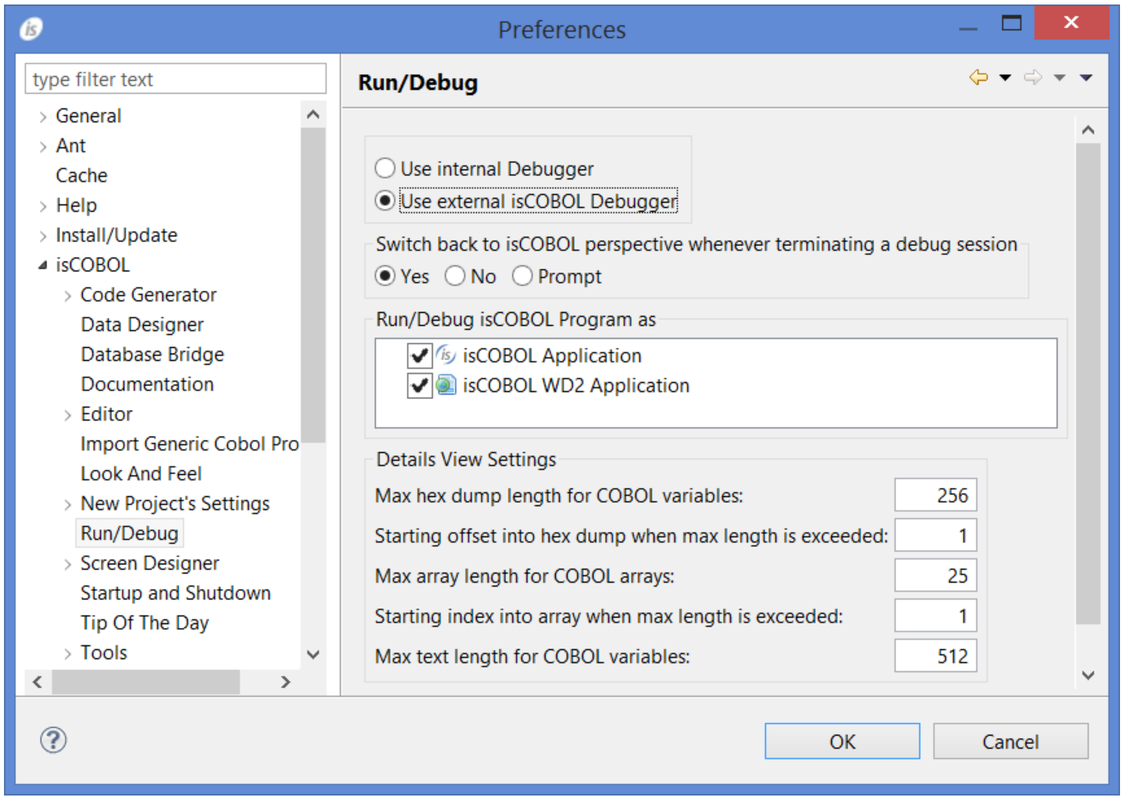
☑ Use saturated colors in overview ruler

This option is turned off by default and can be enabled on the General > Editors > Text Editors > Accessibility page.

New Preferences and Import properties

Added new options in the Window Preferences to use the external isCOBOL Debugger as shown in Figure 7, *Preference to use the external isCOBOL Debugger,* and to set the cell measurements of the imported Generic Cobol Program as shown in Figure 8, Preferences for Import Generic Cobol Program

Figure 7. Preference to use the external isCOBOL Debugger



Run/Debug

Use internal Debugger [✓ Use external isCOBOL Debugger]

Switch back to isCOBOL perspective whenever terminating a debug session

[✓ Yes] No Prompt

Run/Debug isCOBOL Program as

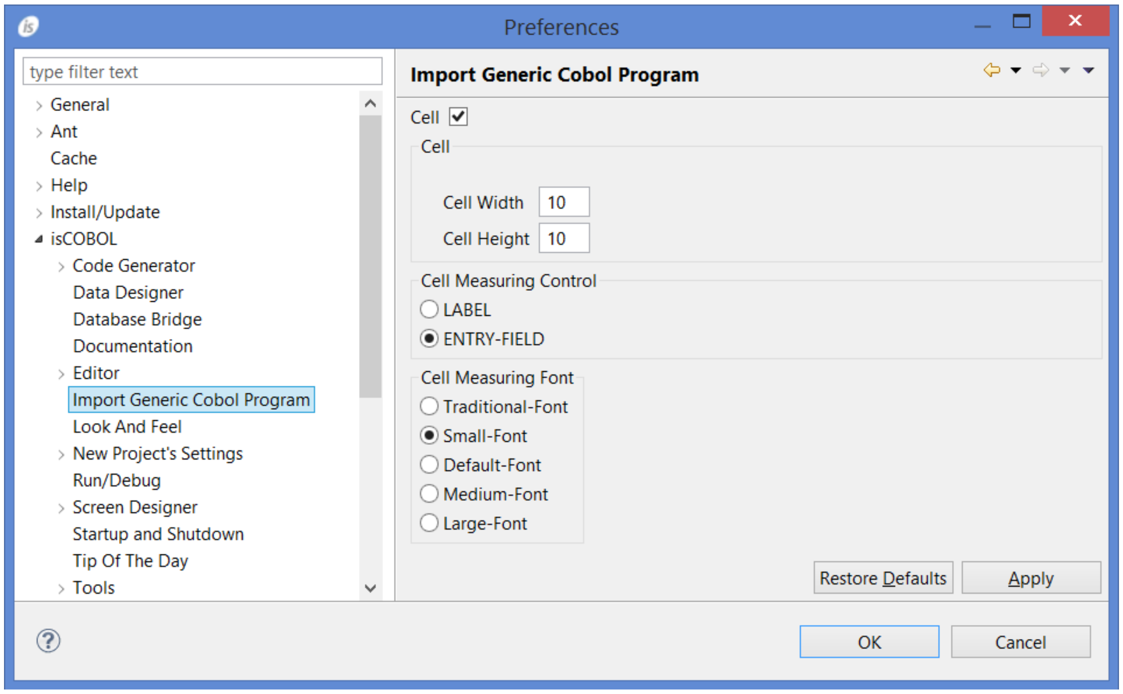
✓ isCOBOL Application

✓ isCOBOL WD2 Application

Details View Settings

Max hex dump length for COBOL variables:	256
Starting offset into hex dump when max length is exceeded:	1
Max array length for COBOL arrays:	25
Starting index into array when max length is exceeded:	1
Max text length for COBOL variables:	512

Figure 8. Preferences for Import Generic COBOL Program



Import Generic Cobol Program

Cell [✓]

Cell

Cell Width 10

Cell Height 10

Cell Measuring Control

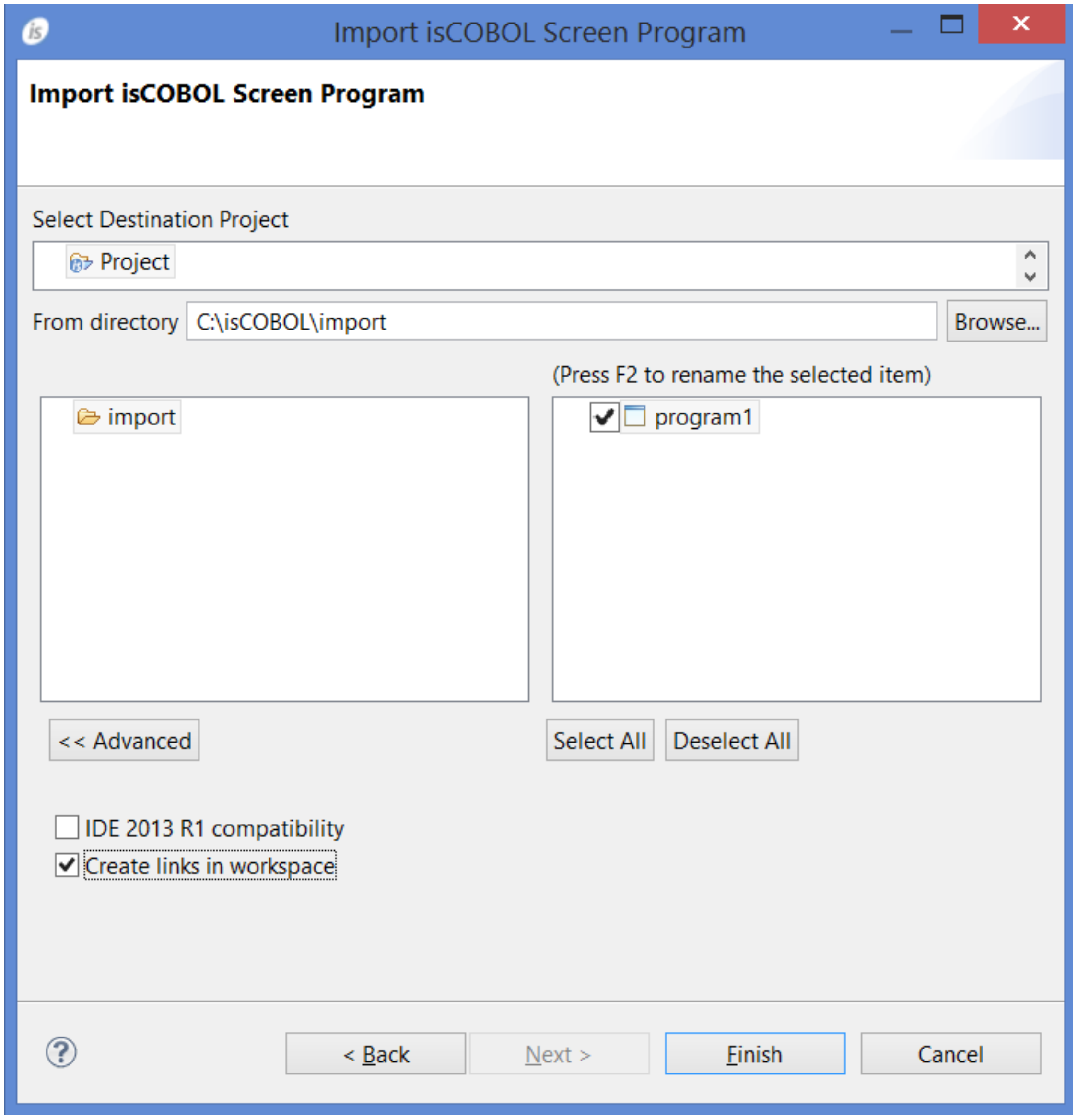
LABEL [✓ ENTRY-FIELD]

Cell Measuring Font

Traditional-Font [✓ Small-Font] Default-Font Medium-Font Large-Font

Also new options are now available during import to link an isCOBOL Screen Program as shown in Figure 9, *Import a Screen Program as a link*. This allows for managing the same isCOBOL Program from different workspaces.

Figure 9. Import a Screen Program as a link


Import isCOBOL Screen Program

Import isCOBOL Screen Program

Select Destination Project: Project

From directory: C:\isCOBOL\import

(Press F2 to rename the selected item)

- import
- [✓] program1

- [ ] IDE 2013 R1 compatibility
- [✓] Create links in workspace

Compiler Enhancements

Improvements to the isCOBOL Compiler in 2014 R1 include:

Embedded ESQL improvements

The EXEC SQL CALL statement is improved to support parameter types when calling a stored procedure. Also the return type ResultSet is now supported. Code example:

```
EXEC SQL
    CALL MyTestStoredProc(:PAR1 IN,
                          :PAR2 INOUT,
                          :PAR3 OUT)
END-EXEC
EXEC SQL
     DECLARE C1 CURSOR FOR CALL SP_SERVER_INFO(1)
END-EXEC
```

Support for CODE-SET on sequential files

The isCOBOL compiler now supports CODE-SET clause on FD to specify the alphabet to use for a sequential file. Code example to produce an EBCDIC sequential file:

```
special-names.
    alphabet do-ebcdic is ebcdic.
input-output section.
file-control.
    select ebcout assign to "edbcdic.txt"
                  organization is binary sequential
                  access mode  is sequential
                  file status  is iostatus.
data division.
file section.
fd  ebcout
    code-set is do-ebcdic.
01  ebcout-rec.
    03 ebc-rec pic x(10).
```

Support for MBP COBOL sign encoding

The isCOBOL compiler is now able to support MBP COBOL sign encoding. Compiler option –dcb should be provided when using data files stored with that convention.

Enhanced compatibility on library routines

In terms of compatibility with ACUCOBOL-GT and Micro Focus, isCOBOL 2014 R1 includes these features:

- New compiler option -fl to set single record locking by default on WITH ROLLBACK
- Library routine S$IO to access sequential files without FD. Code example:

```
set s-open-function to true
move fio to open-mode
call "s$io" using sio-function,
                  f-name,
                  open-mode,
                  1,
                  seq-type,
                  0,
                  0
```

- Enhanced C$SOCKET routine with new op-code AGS-READ-LINE and *timeval* parameter Code example:

```
call "c$socket" using ags-read,
                      socket-handle,
                      data-from-client,
                      100,
                      5000
call "c$socket" using ags-read-line,
                      socket-handle,
                      data-from-client,
                      data-length,
                      500
```

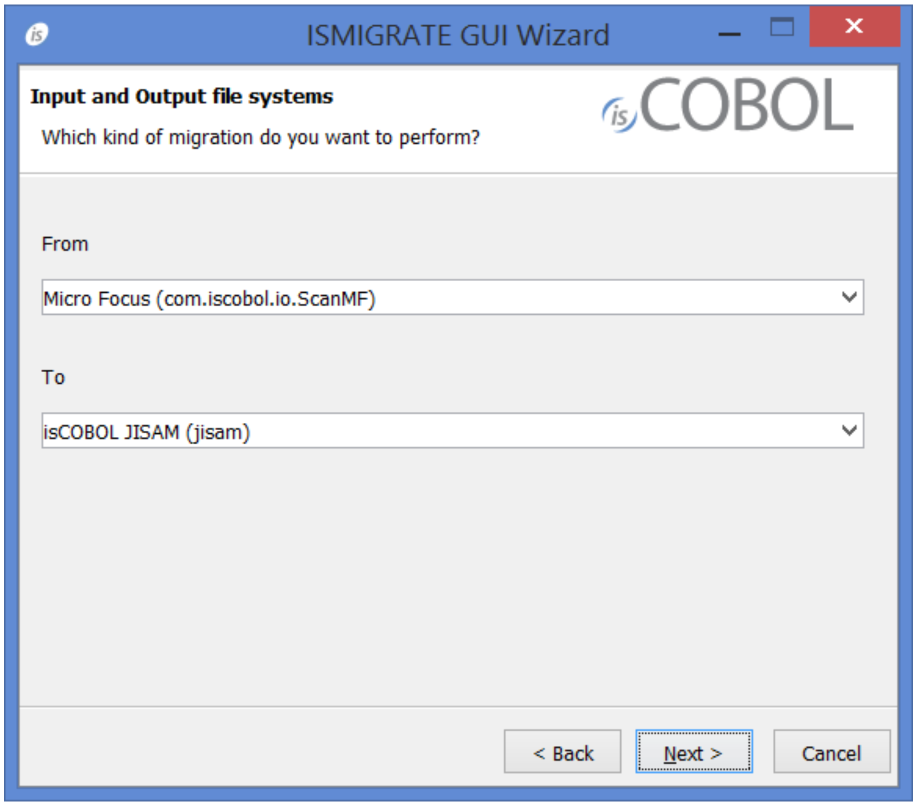
- Library routines: CBL_READ_SCR_CHATTRS and CBL_WRITE_SCR_CHATTRS to store and replace a character area on video. Code example:

```
move 1 to row-number, column-number
move 25 to string-length
call "cbl_read_scr_chattrs" using screen-position,
                                  character-buffer,
                                  attribute-buffer,
                                  string-length
call"cbl_write_scr_chattrs" using screen-position,
                                  character-buffer,
                                  attribute-buffer,
                                  string-length
```

Support for MF indexed files

The isCOBOL Framework is now able to read Micro Focus indexed files natively; this feature is particularly useful during the conversion from Micro Focus to easily convert data files to one of the file systems supported by isCOBOL. This is fully integrated in the isCOBOL utility ISMIGRATE used for the migration of indexed files as shown in Figure 10, *ISMIGRATE utility to convert Micro Focus indexed files*.

Figure 10. ISMIGRATE utility to convert Micro Focus indexed files



Input and Output file systems

Which kind of migration do you want to perform?

From

Micro Focus (com.iscobol.io.ScanMF)

To

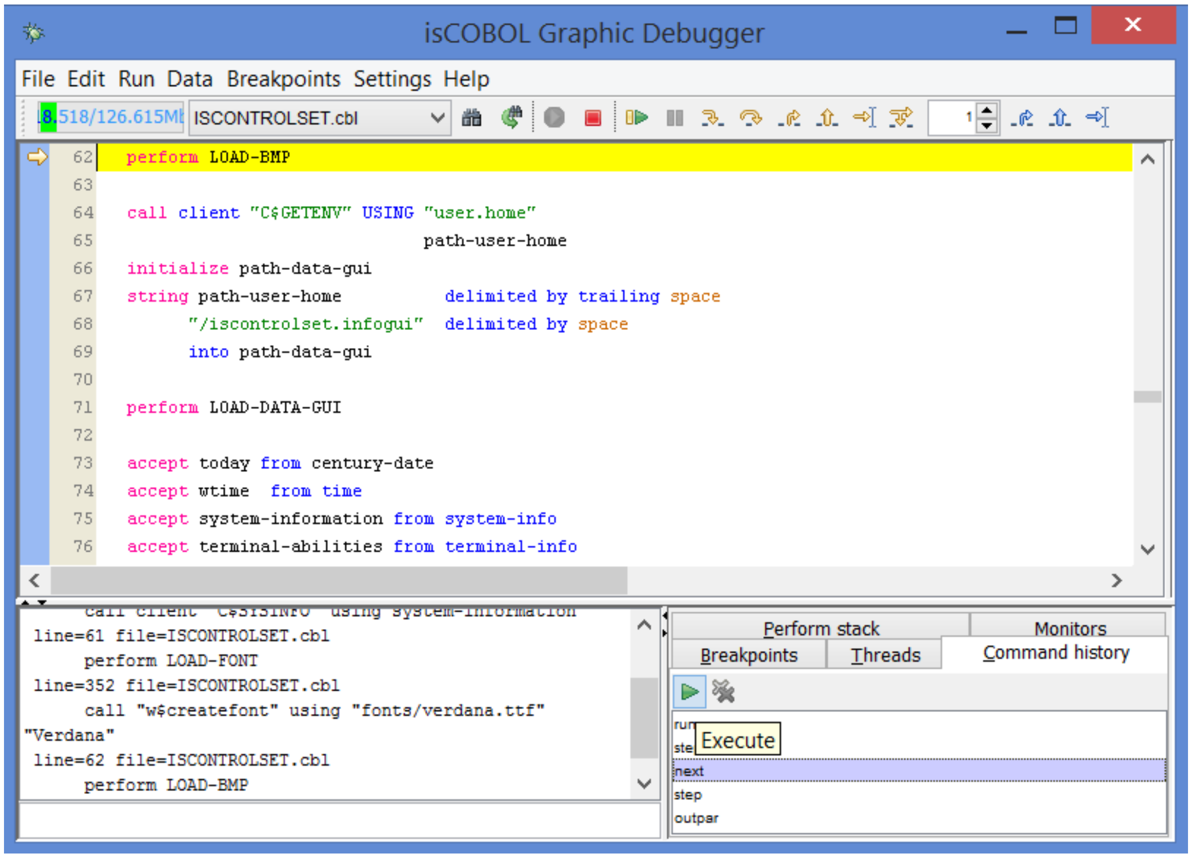
isCOBOL JISAM (jisam)

Debugger Improvements

Command history

In isCOBOL 2014 R1, developers have the ability to repeat previous commands in isCOBOL Debugger using *ctrl+up* / *ctrl+down* or choosing an item in the Command history. As shown in Figure 11, *Command history in isCOBOL Debugger*, the Command history feature allows for easily seeing the previous commands and to select one of them to be repeated.

Figure 11. Command history in isCOBOL Debugger



```
perform LOAD-BMP

call client "C$GETENV" USING "user.home"
                                   path-user-home
initialize path-data-gui
string path-user-home          delimited by trailing space
       "/iscontrolset.infogui"  delimited by space
       into path-data-gui

perform LOAD-DATA-GUI

accept today from century-date
accept wtime  from time
accept system-information from system-info
accept terminal-abilities from terminal-info
```

```
     call client  C$SYSINFO  using system-information
 line=61 file=ISCONTROLSET.cbl
     perform LOAD-FONT
 line=352 file=ISCONTROLSET.cbl
     call "w$createfont" using "fonts/verdana.ttf"
"Verdana"
 line=62 file=ISCONTROLSET.cbl
     perform LOAD-BMP
```

```
run
ste
next
step
outpar
```

Sources on remote machine

Ability to remotely debug having source files on a remote machine through new properties:

> *iscobol.debug.remote_source*=1 to be set on the client side and
> *iscobol.debug.remote_source_enabled*=1 to be set on the server side

This allows maintaining source files on the server machine while using the Remote Debugger on a client machine.

isCOBOL Database Bridge Improvements

isCOBOL Database Bridge, the tool that Enables COBOL programs to interact with a RDBMS without changing COBOL source code or learning ESQL was enhanced to better satisfy user's requests.

Point the correct table directly during Start on multi-record tables

COBOL files with multiple record definitions are split to different tables (one for each record definition) on the database. When a START is performed, previous Database Bridge versions performed a query on all of these tables in order to find the first valid record. Now, by using the new option *–esst*, the EDBI routines point directly to the table associated to the current record type. This will enhance the performance. As a side effect, only the records of the same record type can be read after the START.

Better management of duplicate values in alternate keys

In the past when the file pointer was positioned between duplicate key values it was difficult for Database Bridge routines to get the correct record if the program performed a READ in the opposite direction. This situation has been managed with the introduction of a new option: *-pdo*. If used, this option generates additional code in the subroutine to append the unique primary key value to the duplicate alternate key values so that READ operations always point to a precise record.

New User Interface Features

Enhanced message box

The message box now supports color and font. Also the type allows having additional values. Code example:

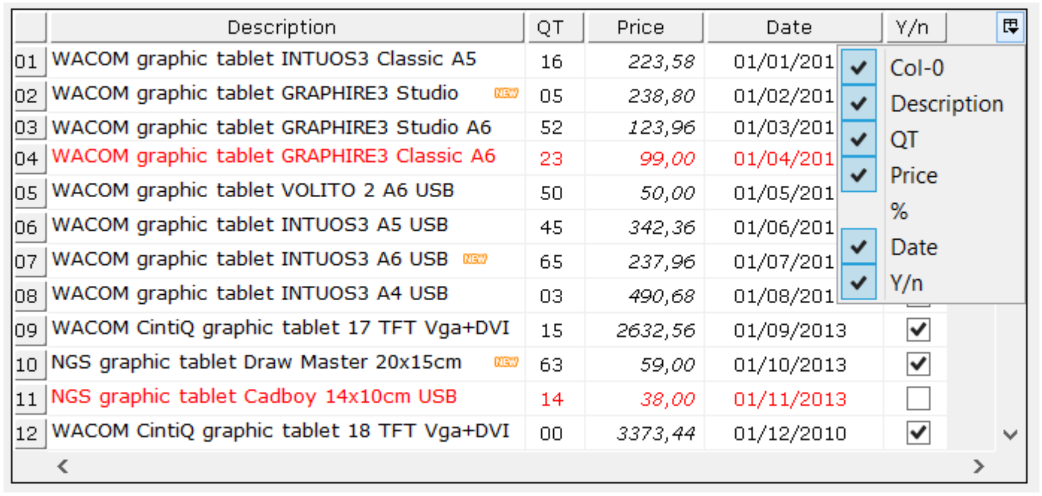
```
display message box "This is a message with type retry-cancel"
        font font1
        foreground-color RGB x#CEE3F6
        background-color RGB x#395A9D
        type mb-retry-cancel
         giving mb-return
```

New properties on grid control

The grid now allows having an automatic pop-up menu on the grid to hide columns with the property HEADING-MENU-POPUP set to 2 as shown in Figure 12, *Automatic pop-up menu shown on the grid with property HEADING-MENU-POPUP set to 2*. In addition a color can be set to be applied on the current cell during editing through new properties CELL-ENTRY-COLOR, CELL-ENTRY-BACKGROUND-COLOR, CELL-ENTRY-FOREGROUND-COLOR. Code example:

```
03 my-grid grid line 2, col 2, lines 10, size 60 cells
           column-headings, tiled-headings, sortable-columns
           display-columns (1, 15, 25, 40, 50)
           heading-menu-popup 2
           cell-entry-foreground-color RGB x#CEE3F6
           cell-entry-background-color RGB x#395A9D
```

Figure 12. Automatic pop-up menu shown on the grid with property HEADING-MENU-POPUP set to 2



	Description	QT	Price	Date	Y/n
01	WACOM graphic tablet INTUOS3 Classic A5	16	223,58	01/01/201	
02	WACOM graphic tablet GRAPHIRE3 Studio	05	238,80	01/02/201	
03	WACOM graphic tablet GRAPHIRE3 Studio A6	52	123,96	01/03/201	
04	WACOM graphic tablet GRAPHIRE3 Classic A6	23	99,00	01/04/201	
05	WACOM graphic tablet VOLITO 2 A6 USB	50	50,00	01/05/201	
06	WACOM graphic tablet INTUOS3 A5 USB	45	342,36	01/06/201	
07	WACOM graphic tablet INTUOS3 A6 USB	65	237,96	01/07/201	
08	WACOM graphic tablet INTUOS3 A4 USB	03	490,68	01/08/201	
09	WACOM CintiQ graphic tablet 17 TFT Vga+DVI	15	2632,56	01/09/2013	✓
10	NGS graphic tablet Draw Master 20x15cm	63	59,00	01/10/2013	✓
11	NGS graphic tablet Cadboy 14x10cm USB	14	38,00	01/11/2013	☐
12	WACOM CintiQ graphic tablet 18 TFT Vga+DVI	00	3373,44	01/12/2010	✓

New events on grid and tree-view controls

Additional events are returned on grid and tree-view controls when the user tries to navigate outside the control, so the program logic can manage this situation. Name of additional events:

MSG-GOTO-CELL-OUT-PREV and MSG-GOTO-CELL-OUT-NEXT on grid,

MSG-TV-SELCHANGE-OUT-PREV and MSG-TV-SELCHANGE-OUT-NEXT on tree-view

In addition, only when the grid has set the property HEADING-MENU-POPUP to 2 (previous example) these new events are returned:

MSG-BEGIN-HEADING-MENU-POPUP and MSG-HEADING-MENU-POPUP

New library routine

The new routine "J$GETFROMLAF" returns fonts and colors from the used LAF during execution. This allows alignment with Look And Feel fonts and color in a graphical isCOBOL application. Code example:

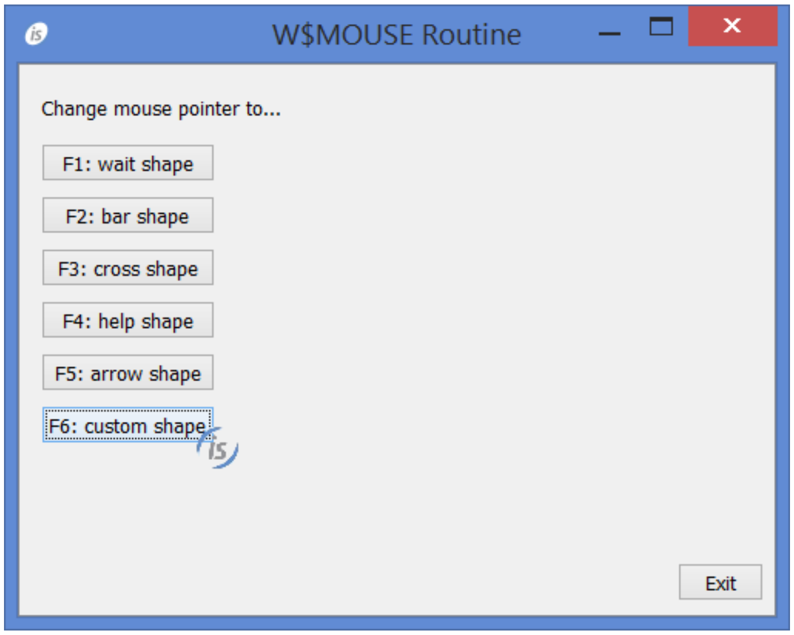
```
 77   f  handle of font.
 77   bg pic s9(9).
 77   fg pic s9(9).
   …
      call "J$GETFROMLAF" using jget-laf-font,
                                "Label.font",
                                f.
      call "J$GETFROMLAF" using jget-laf-color,
                                "List.selectionBackground",
                                bg.
      call "J$GETFROMLAF" using jget-laf-color,
                                "List.selectionForeground",
                                fg.
```

Enhanced W$MOUSE and W$MENU library routines

A new value CUSTOM-POINTER is now supported for the SET-MOUSE-SHAPE op-code of W$MOUSE routine to show a customized mouse shape as shown in the Figure 13, *Custom pointer shown as shape with W$MOUSE*. Code example:

```
call "W$BITMAP" using wbitmap-load,
                       "cursor.png"
             returning cursor-handle
call "W$MOUSE" using set-mouse-shape,
                      custom-pointer,
                      cursor-handle,
                      10,
                      10
```

Figure 13. Custom pointer shown as shape with W$MOUSE



A new op-code WMENU-ENSURE-VISIBLE is available in W$MENU routine to automatically expand the necessary menus to show a given item. Code example:

```
call "W$MENU" using wmenu-ensure-visible,
                     menu-handle,
                     1010
```

Framework Improvements

Enhanced HTTPHandler

Enhanced *HTTPHandler* class adding the ability to save binary files sent with HTTP POST with the "multipart/form-data" HTML form. The behavior is managed by new properties (iscobol.http.upload.max_size, iscobol.http.upload.directory, iscobol.http.upload.prefix, iscobol.http.form.encoding)

New library routine

The new routine "C$PRELOAD" runs an asynchronous thread that loads all COBOL subroutines found in a jar file. This increases the performance of the next CALL statements on an application with a lot of programs. Code example:

```
move "CobProgs.jar" to w-file-name
call "C$PRELOAD" using w-file-name giving ret
```

Enhanced Win$Printer and Print Preview

New op-code in Win$Printer routine (WINPRINT-SET-CUSTOM-PAPER) to set custom paper size. Code example to set a "Letter" format:

```
call "win$printer" using winprint-set-custom-paper ,
                         8.5,
                         11,
                         wprtunits-inches.
```

Added ability to have duplex printing with WIN$PRINTER

New attributes supported to be used when creating pdf files:

- "JPEG" to manage image compression
- "PDFA" to create PDF/A-1a and PDF/A-1b formats

New features added in the Print Preview:

- New buttons to open the page setup and the printer selection
- All hints are now localized
- Ability to hide buttons programmatically with the isCOBOL class *com.iscobol.rts.print.SpoolPrinter*.

Code example to hide the PrintSetup button on the Print Preview window shown on the PC when running locally or on the client machine when running through Application Server:

```
repository.
class j-boolean as "java.lang.Boolean"
class is-java-bean as "com.iscobol.gui.server.CobolGUIJavaBean".
working-storagesection.
77  jb object reference is-java-bean.
77  class-name pic x any length.
procedure division.
   move "com.iscobol.rts.print.SpoolPrinter" to class-name
   jb:>callStaticMethod
      (class-name, "setShowPrintSetupButton", j-boolean:>FALSE).
```

Other enhancements in isCOBOL 2014R1:

- Ability to have a 36 digits decimal precision on internal computations (iscobol.math.fpp36=1)
- Ability to set the max size and number of logfile (iscobol.logfile.maxlen=n and iscobol.logfile.number=n)
- Ability to get the exception 4 pressing F4 in a combo-box (iscobol.gui.f4_drops_combobox=0)
- Ability to set the timeout for selection by keyboard on the tree-view control (iscobol.treeview.selection_delay=n)
- Ability to configure the date cutoff value on a date-entry control (iscobol.gui.date_entry.cutoff=n)
- Ability to set color on the selected combo-box (iscobol.gui.curr_bcolor and iscobol.gui.curr_fcolor)

Ability to automatically skip editing characters during numeric edited character accept (iscobol.terminal.edited_formatted=1)Theability to have alphanumeric fields in update mode like the numeric fields during character accept (iscobol.terminal.alpha_autoclear=1)

isCOBOL Server Improvements

Starting from isCOBOL 2014R1, new features are available:

- Ability to invoke static methods remotely through the new methods of the isCOBOL class "com.iscobol.gui.server.CobolGUIJavaBean". Code example:

```
class cg-java-bean as "com.iscobol.gui.server.CobolGUIJavaBean"
…
set result to cg-java-bean:>callStaticMethod
              ("java.lang.System", "getProperty",
               "java.version" as string, "" as string)
set result to cg-java-bean:>callStaticMethodOnServer
              ("RemoteClassTest", "method1" arg)
```

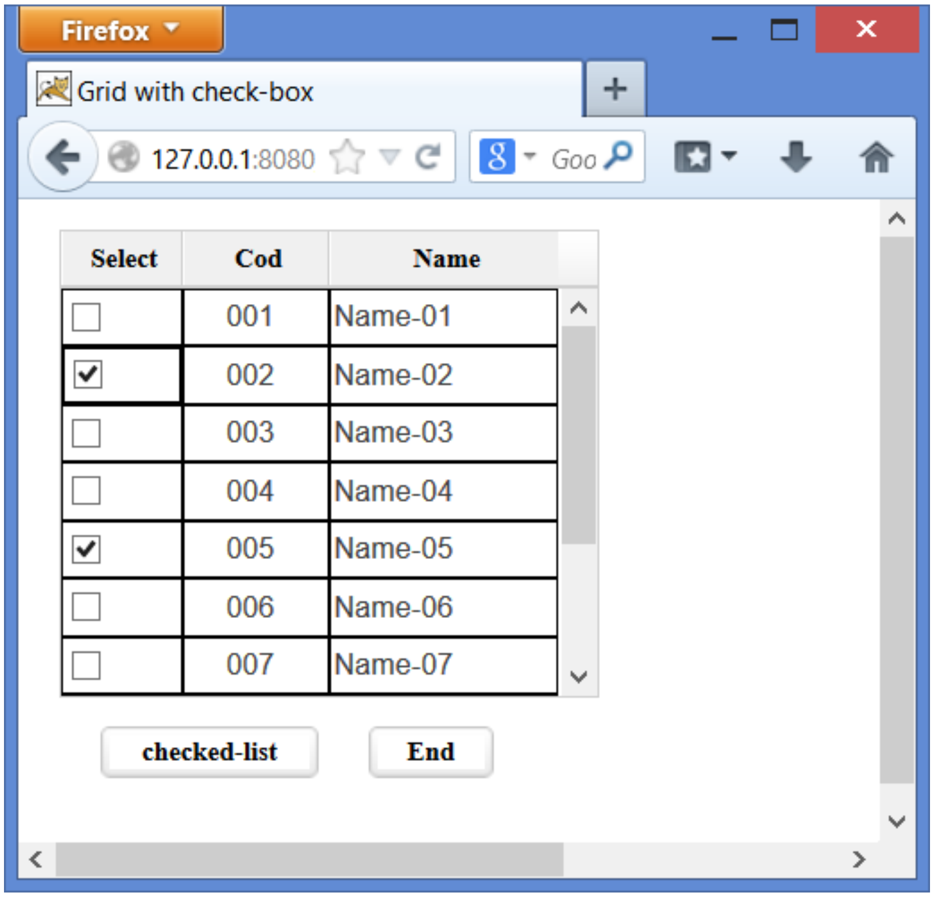
- Ability to invoke a subroutine on the file server as you call a stored procedure on a database.

Web Direct 2.0 Improvements

Web Direct 2.0 has been improved and now offers Controls over cells even outside msg-begin-entry to easily input some columns with appropriate control. Code example to have a check-box to be accepted on all lines of the first column as shown in Figure 14, *Grid with first column to input as a check-box under Web Direct 2.0*:

```
display check-box upon grid-name(-1, 1)
```

Figure 14. Grid with first column to input as a check-box under Web Direct 2.0



Select	Cod	Name
☐	001	Name-01
☑	002	Name-02
☐	003	Name-03
☐	004	Name-04
☑	005	Name-05
☐	006	Name-06
☐	007	Name-07