

IBM i Programming Tool

User Guide

Contents

Contents	2
Introduction.....	4
Distribution.....	4
Overview	5
Application parameters.....	6
<i>User name / IBM i server.....</i>	<i>6</i>
<i>Connect/Reconnect.....</i>	<i>6</i>
<i>IBM i library prefix.....</i>	<i>7</i>
<i>IBM i source type.....</i>	<i>7</i>
Standard names of source files and their types	7
<i>PC charset.....</i>	<i>8</i>
<i>IBM i CCSID.....</i>	<i>8</i>
<i>Source line length.....</i>	<i>9</i>
<i>Complete source record.....</i>	<i>9</i>
<i>Overwrite data.....</i>	<i>9</i>
<i>Windows disks.....</i>	<i>9</i>
File systems as trees.....	9
<i>Left tree – PC.....</i>	<i>10</i>
<i>Right tree – IBM i.....</i>	<i>10</i>
Creating directories and files	11
Copy methods	11
Copy from PC to IBM i.....	11
<i>PC file → source member.....</i>	<i>11</i>
<i>PC file → IFS stream file</i>	<i>12</i>
<i>PC file → save file</i>	<i>12</i>
Copy from IBM i to PC.....	13
<i>Source member → PC file.....</i>	<i>13</i>
<i>IFS stream file → PC file</i>	<i>13</i>
<i>Save file → PC file</i>	<i>13</i>
Copy from PC to PC	13
Copy from IBM i to IBM i	14
<i>Source member → source member</i>	<i>14</i>
<i>Source member → IFS stream file</i>	<i>14</i>
<i>IFS stream file → source member.....</i>	<i>14</i>
<i>IFS stream file → IFS stream file.....</i>	<i>14</i>
<i>Save file in library → IFS.....</i>	<i>15</i>
<i>Save file in IFS → library</i>	<i>15</i>
<i>Library → library</i>	<i>15</i>
Displaying.....	16
Editing.....	17
<i>Highlight blocks</i>	<i>18</i>
Displaying and editing in PC – character sets.....	18
Displaying and editing in IBM i – character sets	18
Spooled files	19
Compilation	20

<i>Source type</i>	21
Source members	21
IFS stream files	22
<i>Compile command</i>	22
<i>Change library list</i>	22
Library prefix	23
Current library	23
Creating a user library list	23
<i>Compiled object</i>	23
Library	23
Object.....	23
Library prefix	23
<i>Perform command</i>	23
<i>Spooled files</i>	24
<i>Job log</i>	24
<i>Edit</i>	24

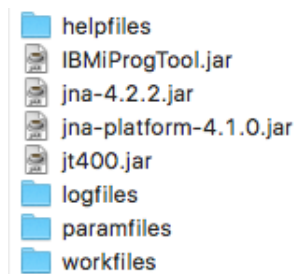
Introduction

This application replaces some functions of the *System i Navigator*, which ceased to work in Windows 10, especially simple transfer of files between IBM i and PC, displaying and editing of files and the like. In addition, the application enables compiling source members or stream files and finding errors from the compilation listing.

Application programs are written in Java and require version *Java SE 8*. They cooperate with classes in *IBM Toolbox for Java* (or JTOpen). The application has been created and tested in systems macOS and Windows 10. Remote connection to the system IBM i, version 7.3 has been used.

Distribution

The application does not require an installation. It is delivered as a directory containing other directories and files:



- The *IBMiProgTool.jar* file is a main, starting file. The application starts after double click with the primary mouse button.
- The other *.jar* files are auxiliary programs.
- Directory *helpfiles* contains files for help (this guide).
- Directory *logfiles* contains text files *err.txt* and *out.txt*, the redirected output from System.err and System.out files.
- Directory *paramfiles* contains text file *Parameters.txt* with application parameters.
- Directory *workfiles* contains auxiliary files to retain the user library listst, current library name and the recently created spooled file.

Note: Files *err.txt* and *out.txt* can be used for revealing the cause of a program error.

Overview

This application allows to

- create files and directories in PC and IBM i,
- remove files and directories in PC and IBM i,
- renaming files and directories in PC and IBM i,
- copying files and directories between PC and IBM i,
- displaying and editing text files in PC and IBM i,
- copying, clearing and deleting libraries,
- compile source members and IFS stream files,
- displaying spooled files.

The application in IBM i works with the following object types:

- Source physical file type *FILE, attribute PF,
- Source member Source physical file member,
- Save file type *FILE, attribute SAVF,
- IFS directory type *DIR,
- IFS stream file type *STMF,
- Output queue type *OUTQ with spooled files inside,
- Library type *LIB.

The source physical file is considered a directory in the application.

The source member is considered a file in the application.

The application does not allow creating new libraries.

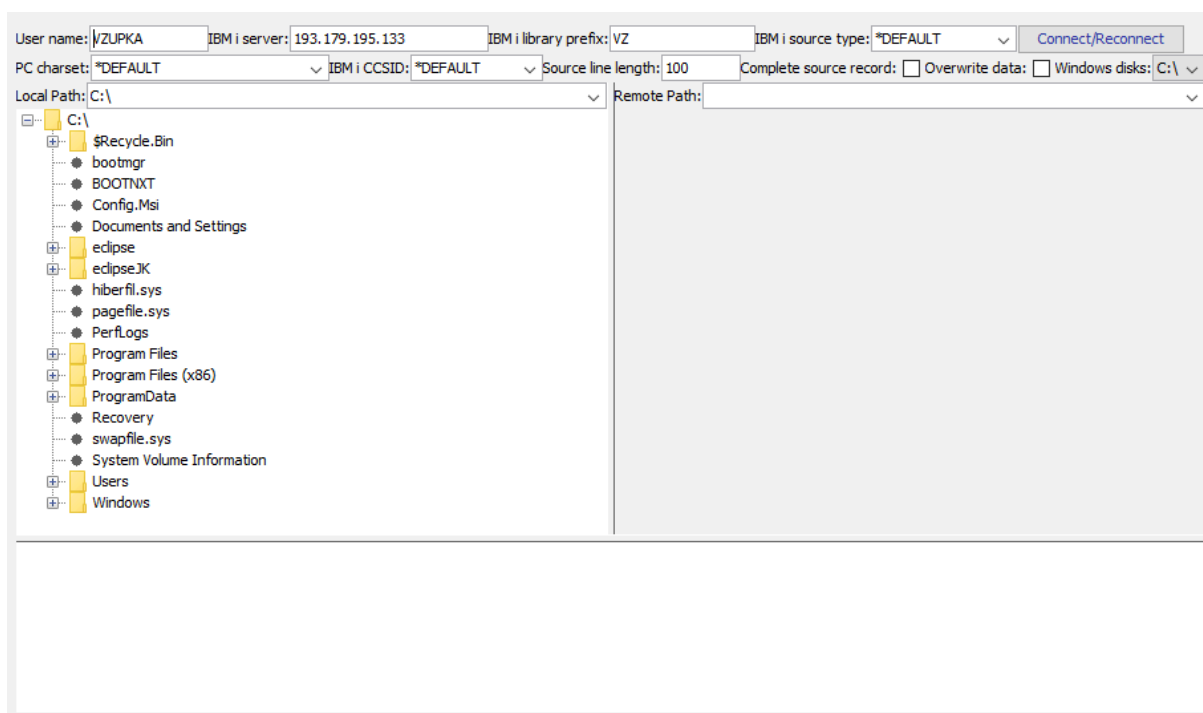
Application parameters

After the application starts, a window is displayed, where the left side shows the tree representing the PC file system. The right side is empty for the time being, it is filled after connection to the system IBM i.

The upper part of the window contains parameters that define application behavior. They have form of input text fields, combo boxes, and check boxes. Also a button for connection to the IBM i server is in the upper part.

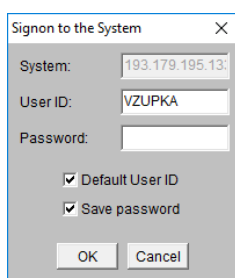
User name / IBM i server

The user must enter a user name and an address of IBM i server and then press the Connect/Reconnect button to connect to the server. The other parameters may be adjusted later.

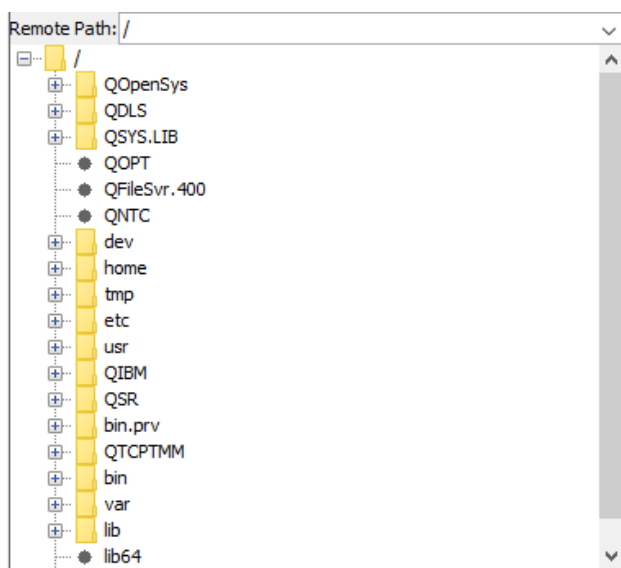


Connect/Reconnect

After pressing the Connect/Reconnect button the user will sign on and connect to the server. When the user connects for the first time, the dialog *Signon to the System* is displayed where the user enters the password. The process of connection may last longer, according to the speed of the line. Next pressing the button the server is connected again. The new connection may be without the dialog and last shorter, if the user did not change the server address in the meantime. The user can connect another server if he changes the address and presses the button or the Enter key.



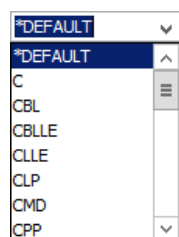
After the successful connection, a tree representing the IBM i file system is shown in the right side of the window. All directories of the IFS file system are contained in the tree.



IBM i library prefix

The user enters starting characters of the library names that are to be listed under the library QSYS.LIB. For example, entering letters VZ selects libraries whose names start with VZ.

IBM i source type



The user selects or enters the source type of the source member that is to be copied to PC or to IFS. If a *new file* is created, it gets a suffix of this type. For example, if the user selects type RPGLE when copying member PROG01.MBR, the newly created file gets name PROG01.RPGLE.

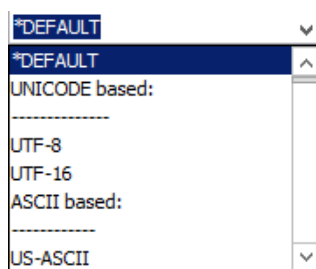
The special value *DEFAULT assigns the type automatically if the source file has a standard name. For example, if the source file is QRPGLSRC (standard name for ILE/RPG programs), type RPGLE is assigned. If the source file has not a standard name, type TXT is assigned.

Standard names of source files and their types

QBASSRC	BAS	Basic
QCBLESRC	CBLLE	ILE/Cobol
QCLSRC	CLLE	CL
QCMDSRC	CMD	Command
QCSRC		C C language
QDDSSRC	PF	Physical file
QFTNSRC	FTN	Fortran
QCBLSRC	CBL	Cobol/400

QMAPSRC	CICS MAP	CICS Customer Information Control System
QMENUSRC	MNUDDS	Menu DDS
QMNUSRC	MENU	UIM menu
QPASSRC	PAS	Pascal language
QPLISRC	PLI	PL/I language
QPNLSRC	PNLGRP	Panel group
QREXSRC	REXX	REXX
QRMCSRC	RMC	RM/Cobol-85
QRPGLSRC	RPGLE	ILE/RPG language
QRPGSRC	RPG	RPG/400
QS36PRC	OCL36	System/36 Operator Control Language
QS36SRC	UNS36	S36 unspecified
QSRVSR	BND	Binder source
QTBLSRC	TBL	Table
QXTSRC	TXT	Text
QUSSRC	QRY38	Query/38

PC charset

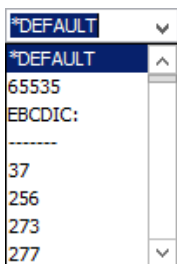


The user can select a character set from the combo box or enter a code in the input text field. A special value `*DEFAULT` is also in the combo box list.

The *PC charset* parameter is applied when copying data and when displaying files in some situations (see below).

Note: Unlike IFS files, PC files have no character set attribute. Therefore the *PC charset* parameter is not applied in creating a PC file.

IBM i CCSID



The user can select a CCSID code ((Coded Character Set ID) from the combo box or enter a code in the input text field.

The *IBM i CCSID* parameter is applied when copying data and when displaying files. It is applied also in creating a new source physical file, in creating a new IFS stream file, or in displaying and copying spooled files. (See details below.)

Source line length

This value defines length of the source text line when creating a new source physical file. For example, for the source physical file QDDSSRC, length 80 is defined as a standard, for the file QRPGLSRC, length 100 is defined, etc. Another length can be chosen, though.

When copying a PC text file to a source member, the line (or its text part) of the PC file can be longer than the length of the data part in the source member (defined in creating the source file). Then the line is shorten so that it fits in the data part of the source record.

Complete source record

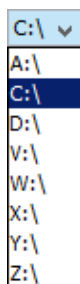
The check box, if checked, means that a 12-character data is prepended to each output line when copying a source member to PC. This data contains the sequential number and the date from the source record. If the check box is unchecked, only the data part from the source member is copied.

Opposite, when copying a PC text file to a source member, this check box is not applied. The application is driven by the first line of the PC file. If the first two 6-character values are whole numbers, they are prepended to the source record. If not, missing data is derived as a computed sequential number and the PC sytem date. These numbers are prepended to the line from the PC file and written together as a record to the source member.

Overwrite data

The check box, if checked, means that it is allowed to overwrite data in existing files. If unchecked, data in existing files cannot be overwritten.

Windows disks



The list of disks is available only in Windows. In fact, it is a list of root directories of the Windows file system. The user selects one from the list. Disk C:\ is default.

File systems as trees

The PC file system on the left and the IBM i file system on the right are presented as trees. The *root* of the tree is placed in the first row.

The root of the file system is the root of the tree on the first application start.

For Windows it is C:\.

For unix type systems it is the forward slash / (called root).

For IBM i it is the forward slash / like in unix type systems.

The user can change the tree root using the combo box or by entering a path to an object – a directory or a file. Gradually, as the user expands individual *nodes* of the tree, paths to

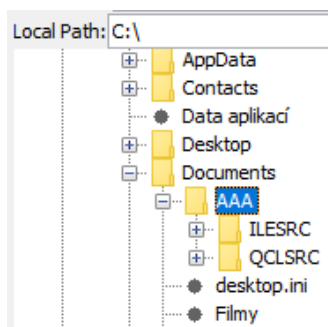
corresponding objects are entered to the list in the combo box. If the user then selects a path from the list (usually a directory), the path becomes the root of the new tree.

The application saves the actual root. On the next application start, the recently saved root and the corresponding tree is displayed.

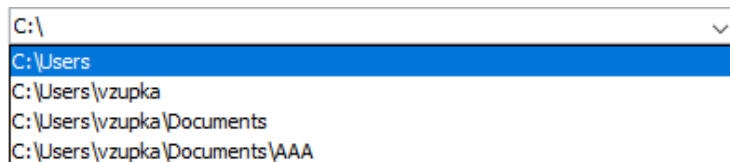
Important: It is necessary to expand the node by *clicking on its text or icon*. Only then the subordinate nodes are filled with information about next subordinate elements. Clicking on *expansion symbol* (handle) or *double clicking* on the text or icon works only after the node was expanded before regularly by clicking on the text or icon. After that, the node can be collapsed or expanded.

Left tree – PC

Click with the *left mouse button* on a node (AAA in the picture) reveals its objects (directories ILESRC and QCLSRC).

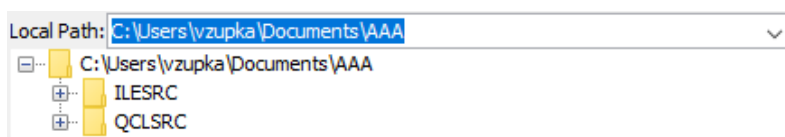


At the same time, the path leading to the node is entered to the list. If we expand the list "Local Path", we will see all paths leading to the nodes expanded until now:



If we select the last directory, it is entered in the input field and the tree is displayed from the new root:

C:\Users\vizupka\Documents\AAA.



Right mouse button is used to display context menus with different commands.

Right tree – IBM i

The way of object and root selection in the right tree is the same as in the left tree. Commands in the context menus are more varied, though. Difference is apparent in objects under the node QSYS.LIB, that is, in the library/object system. Libraries that we are not interested in, cannot be expanded (see above).

Creating directories and files

Select command *New*. . . from the context menu on a selected node that is not a leaf (ending node).

New source physical file gets its CCSID from the *IBM i CCSID* parameter.

New source member gets its CCSID from its source physical file in which it is created as its attribute.

New IFS directory get its CCSID from the *IBM i CCSID* parameter as its attribute.

New IFS file gets the CCSID from the *IBM i CCSID* parameter as its attribute.

Directories and file in PC do not contain any information about a character set.

Copy methods

Files and directories can be copied in all directions:

- PC ↔ IBM i
- IBM i ↔ PC
- PC ↔ PC
- IBM i ↔ IBM i

Two copy methods can be used. The first method makes use of commands *Copy* and *Paste* in context menus. The other method is *drag and drop*.

Copy from PC to IBM i

- PC file → IFS stream file
- PC file → IFS directory
- PC directory → IFS directory
- PC file → source member
- PC file → source physical file
- PC directory → source physical file
- PC file → save file

PC file → source member

Data from the PC file is translated from the character set defined in *PC charset* parameter to the character set defined by the *CCSID attribute* of the source member. Parameter *IBM i CCSID* is *ignored*. Resulting characters need not be in compliance with the target character set.

Note 1: If an error is reported in copying files, an empty file may be created.

Note 2: Data transfer may last quite long, if character sets differ and the source member is long (3000 rows about 2 minutes). The program must read individual input lines and translate each line before writing.

PC file → IFS stream file

Data from the PC file is translated from the character set defined in *PC charset* parameter to the character set defined by the *IBM i CCSID* parameter.

- For *DEFAULT in the *PC charset* parameter, character set *ISO-8859-1* (ASCII ISO-8859-1, Latin Alphabet No. 1) is assigned.
- For *DEFAULT in the *IBM i CCSID* parameter, character set *500* (EBCDIC International Latin-1) is assigned.
- If character sets correspond on both sides, data is transferred without translation in the following cases:
 - ISO-8859-1 → CCSID 819, CCSID 858
 - windows-1250, Cp1250 → CCSID 1250
 - windows-1251, Cp1251 → CCSID 1251
 - UTF-8 → CCSID 1208
 - UTF-16 → CCSID 1200, CCSID 13488
 - IBM500, Cp500 → CCSID 819
 - IBM870, Cp870 → CCSID 870

PC file → save file

Any PC file denoted by suffix *.savf* is considered a save file and can be copied to the save file in a library with suffix *.FILE*. The same PC file can also be copied to an IFS file or directory without changing the suffix, and still retain its content as a save file.

Copy from IBM i to PC

- IFS stream file → PC file
- IFS stream file → PC directory
- IFS directory → PC directory
- source member → PC file
- source member → PC directory
- source physical file → PC directory
- save file → PC file
- save file → PC directory

Source member → PC file

Data from the source member is first translated from character set defined in the *IBM i CCSID* parameter to an auxiliary text coded UTF-16 and then translated to the character set defined in the *PC charset* parameter.

- If **DEFAULT* is entered in the *IBM i CCSID* parameter, data is translated into the auxiliary text using *CCSID attribute* of the source member.
- If **DEFAULT* is entered in the *PC charset* parameter, data from the auxiliary text is translated using character set *ISO-8859-1* (Latin-1).

Note: The source physical file can be created with the CCSID attribute 1208. The source member created in this source physical file has the same attribute. Then, if the IBM i CCSID parameter or CCSID attribute is 1208 (UTF-8 equivalent) it is replaced by value 65535 in translation to the auxiliary text. Thus, data is not translated to UTF-16 as individual bytes but the whole UTF-8 characters.

IFS stream file → PC file

Data from the *IFS stream file* is translated from the character set defined in the *IBM i CCSID* parameter to the character set defined in the *PC charset*.

If **DEFAULT* is entered in both parameters, data is transferred without change (binary).

Otherwise, data is translated using the *IBM i CCSID* parameter into an auxiliary text encoded in UTF-16 and then translated using the *PC charset* parameter.

- If **DEFAULT* is entered only in the *IBM i CCSID* parameter, data is translated into the auxiliary text from the stream file using the CCSID attribute.
- If **DEFAULT* is entered only in the *PC charset* parameter, data from the auxiliary text is translated to the PC file using character set *ISO-8859-1* (Latin-1).

Save file → PC file

A save file is placed in a library and has suffix *.FILE*. After copying to PC, the newly created file gets suffix *.savf*. Existing PC file must have suffix *.savf* in order that the save file can be copied into it.

Copy from PC to PC

- PC file → PC file
- PC file → PC directory
- PC directory → PC directory

No conversion is performed in copying data.

Copy from IBM i to IBM i

- source member → IFS stream file
- source member → IFS directory
- zdrojový soubor → IFS directory
- IFS stream file → IFS stream file
- IFS stream file → IFS directory
- IFS directory → IFS directory
- save file in library → IFS
- save file in IFS → library
- library → library

Source member → source member

Source member data is *translated according to source physical files* in which they reside. That is, from the CCSID attribute of the input source physical file to the CCSID attribute of the output source physical file.

The *IBM i CCSID* parameter is *ignored*.

Source member → IFS stream file

Data from the source member to an *existing* IFS stream file is translated to the CCSID attribute of the IFS stream file.

Data from the source member to an IFS stream file, that is *just being created*, is not translated and the new IFS stream file takes over the CCSID attribute from the *source physical file*, in which the source member resides.

In both cases the *IBM i CCSID* parameter is *ignored*.

IFS stream file → source member

Data from IFS stream file is translated from the CCSID *attribute* of the IFS stream file into the CCSID *attribute* of the *source physical file*, no matter if the source member already exists or is just being created.

The *IBM i CCSID* parameter is *ignored*.

IFS stream file → IFS stream file

Data of the input file is translated from its CCSID attribute to the CCSID attribute of the *existing* output file.

Data is not translated if the output file is *just being created*. The newly created IFS stream file takes over the CCSID attribute of the input file.

If both files have identical CCSID attributes, data is transferred without change (binary).

The *IBM i CCSID* parameter is *ignored*.

Save file in library → IFS

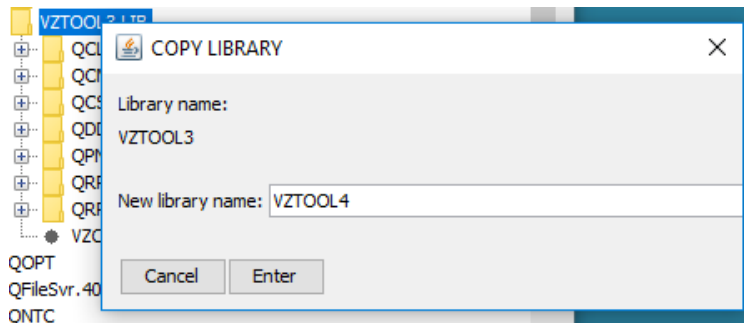
A save file in a library has suffix *.FILE*. If the output IFS stream file already exists, it must have suffix *.savf*. The newly created output IFS stream file gets suffix *.savf*.

Save file in IFS → library

An IFS directory with suffix *.savf* is considered save file. If the output save file already exists in the library, it has suffix *.FILE*. The newly created save file gets suffix *.FILE*.

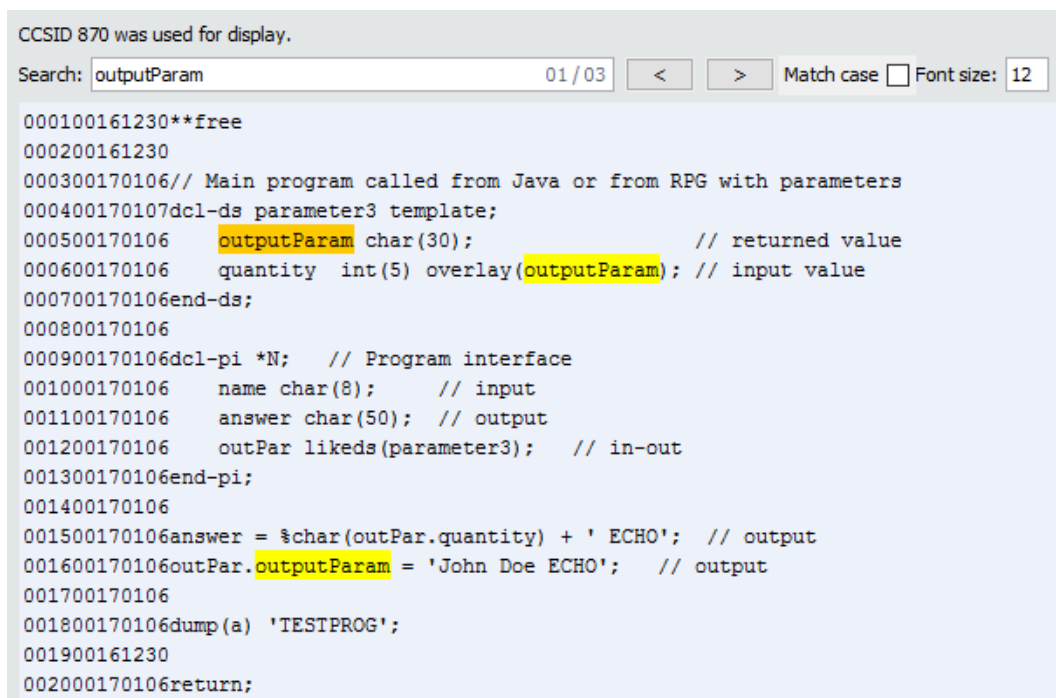
Library → library

A library can be copied under a different, user defined name.



Displaying

Contents of a file is displayed (on Display command) in a separate window.



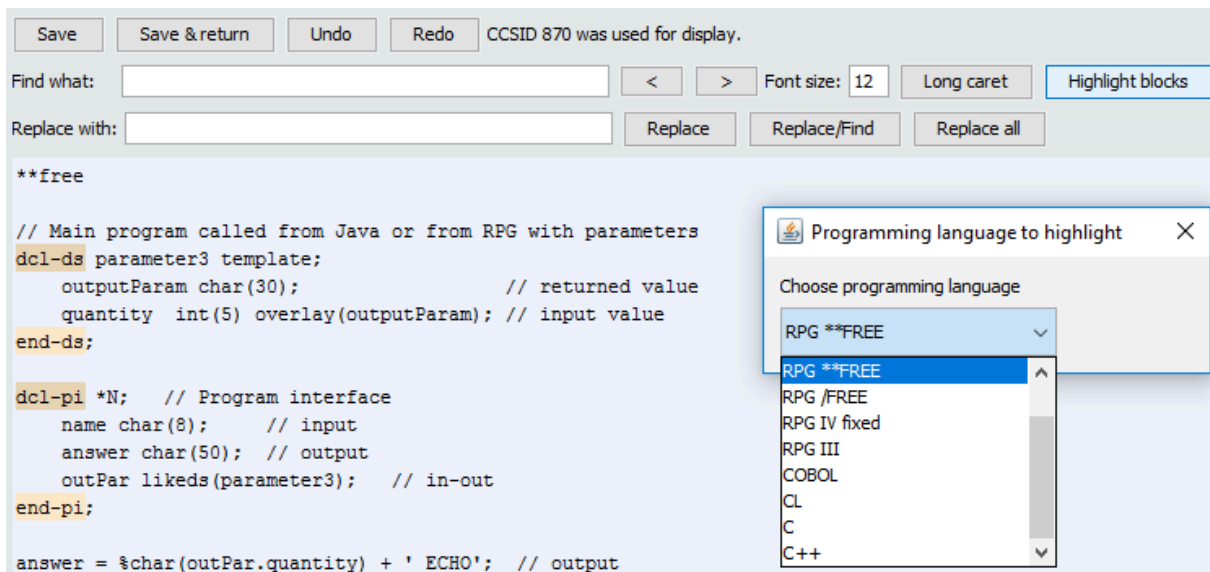
```
CCSID 870 was used for display.
Search: outputParam 01/03 < > Match case ☐ Font size: 12
000100161230**free
000200161230
000300170106// Main program called from Java or from RPG with parameters
000400170107dcl-ds parameter3 template;
000500170106 outputParam char(30); // returned value
000600170106 quantity int(5) overlay(outputParam); // input value
000700170106end-ds;
000800170106
000900170106dcl-pi *N; // Program interface
001000170106 name char(8); // input
001100170106 answer char(50); // output
001200170106 outPar likes(parameter3); // in-out
001300170106end-pi;
001400170106
001500170106answer = %char(outPar.quantity) + ' ECHO'; // output
001600170106outPar.outputParam = 'John Doe ECHO'; // output
001700170106
001800170106dump(a) 'TESTPROG';
001900161230
002000170106return;
```

Two rows of parameters are in the window header. The first line of the window tells what *character set* is used to display the file. The second line contains input fields and buttons.

- *Search:* – a text pattern in the Search field enables finding matching texts. Numbers in the Search field show sequence number and number of matches.
- *Buttons with arrows* – find preceding or following matches in the file. The same function is provided by key shortcuts with the modifier key Ctrl (Windows)/Cmd(Mac):
arrow up (previous)
arrow down (next).
- *Match case* – the check box, if checked, ensures that the text to find must be identical to the text pattern. If not checked, the letters of the text to find need not have identical case.
- *Font size:* – the input field defines font size of the text in the displayed file.

Editing

Contents of a file is displayed (on Edit command) in a separate window.



Extra parameters are present in the window header in comparison to simple display.

Input fields:

- *Find what:* – enter a pattern to find.
- *Replace with:* – enter a replacement text.

Buttons with commands:

- *Cancel* – return to previous work.
- *Save* – save changes.
- *Save & return* – save changes and return to previous work.
- *Undo* – remove changes.
- *Redo* – restore changes.
- *Long caret / Short caret* – define pointer in text as a long vertical line or standard short vertical line (see the picture above).
- *Highlight blocks* – the button shows a dialog window with a combo box to select if and how to highlight compound statements (blocks) in different programming languages (see below).
- *Replace/Find* – replace the matching text just found by the replacement text (from the field *Replace with*) and find the next matching text.
- *Replace all* – replace all matching texts by the replacement text (from the input field *Replace with*).

Highlight blocks

The combo box in the dialog window contains list of programming languages whose compound statements (blocks) should be highlighted. The list contains the following entries:

***NONE** – no highlighting,

***ALL** – blocks for all languages are highlighted,

RPG **FREE – RPG version with completely free statement entries,

RPG /FREE – RPG version allowing both fixed and free statements in calculations,

RPG IV fixed – RPG version allowing extended fixed form statements,

RPG III – RPG version (RPG/400) allowing traditional fixed form statements,

COBOL – COBOL language,

CL – ILE/CL Control language,

C – C language,

C++ – C++ language.

The method of highlighting of compound statements used in this application may have some unwanted effects, because short sequences of letters – namely IF, DO, FOR, may appear also in other places in the text. This might be acceptable for relevant programming languages, however. The lowest efficiency in this sense is for C and C++ languages and for option ***ALL**.

Example of blocks from an RPG program with *RPG /free* option:

```
begsr browseAttributes;
  dou *in03;
    exfmt ATRWIN;
    if *in03;
      exsr endpgm;
    elseif *in12;
      leavesr;
    endif;
  enddo;
  // ???
endsr; // browseAttributes
```

Displaying and editing in PC – character sets

Files are displayed and edited using *PC charset* parameter. If ***DEFAULT** is entered, *ISO-8859-1* is assigned. If the file contains invalid characters, an error message is reported. The user can change the parameter and try again.

Displaying and editing in IBM i – character sets

Source members are displayed and edited using their *CCSID attributes* without regard to the value of the *IBM i CCSID* parameter. Characters are displayed incorrectly if they do not conform to the CCSID.

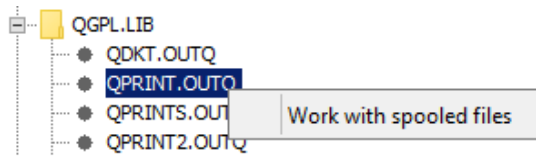
IFS stream files are displayed and edited using their *CCSID attributes* without regard to the value of the *IBM i CCSID* parameter. Characters are displayed incorrectly if they do not conform to the CCSID.

Spooled files are displayed (but not edited) using the *IBM i CCSID* parameter. If incorrect characters are displayed, using ***DEFAULT** or 65535 may help. The program then tries to guess a correct encoding.

Spooled files

Spooled files can be displayed and can also copied to PC. There are two ways how to proceed.

The first way requires knowledge about *what libraries* contain output queue objects (*OUTQ type), for example libraries QGPL or QUSRSYS. Click with the right mouse button on the node having suffix .OUTQ, which invokes menu with a single command *Work with spooled files*.



The other way does not require a library name. Use the system library QSYS.LIB instead. The system library does not contain any output queues directly but it serves here as a placeholder of the menu with the command *Work with spooled files*. Thus *all* spooled files can be accessed.

The command *Work with spooled files* invokes a window with the table of spooled files.

File name	File num.	Pages	Job name	User	Job num.	Date	Time	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	
				QSYS VZUPKA				
File name	File num.	Pages	Job name	User	Job num.	Date	Time	Output queue
QPRINT	1	1	QSLPSVR	QSYS	032508	1170331	134516	QGPL/QPRINT
CENPOR	6	10	QPRTJOB	VZUPKA	033626	1170404	183428	QGPL/QPRINT
TESTPROG	7	7	QPRTJOB	VZUPKA	033626	1170405	102552	QGPL/QPRINT

To limit range of the table, enter conditions to input fields above the table and press Enter key. The text in the field is sought in the corresponding table column. Conditions entered in more fields are evaluated at the same time. An empty field does not limit the table.

The *User* field is a combo box of all profiles (user names) to whom the spooled files belong. Selecting a user name limits the table to spooled files belonging to this user.

Select a row or multiple rows from the table by *left mouse click*, possibly holding Shift or Ctrl keys. Then *right click* reveals the following menu with commands:

Display

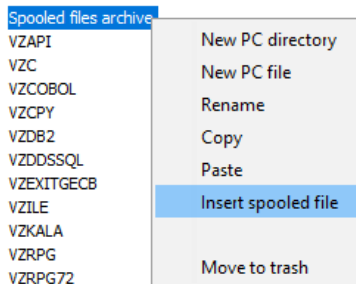
Copy

QPJOBLOG	104	4	QPRTJOB	QSYS	033626	1170414	000023	QUSRSYS/QEZJOBLOG
QPJOBLOG	105	7	QPRTJOB	QSYS	033626	1170414	000025	QUSRSYS/QEZJOBLOG
QPJOBLOG	20	14	QPRTJOB	VZUPKA	033626	1170414	094333	QUSRSYS/QEZJOBLOG
QPJOBLOG	106	4	QPRTJOB	QSYS	033626	1170415	000031	QUSRSYS/QEZJOBLOG
QPJOBLOG	107	7	QPRTJOB	QSYS	033626	1170415	000033	QUSRSYS/QEZJOBLOG
VEKDS04	21	8	QPRTJOB	VZUPKA	033626	1170415	072849	QGPL/QPRINT

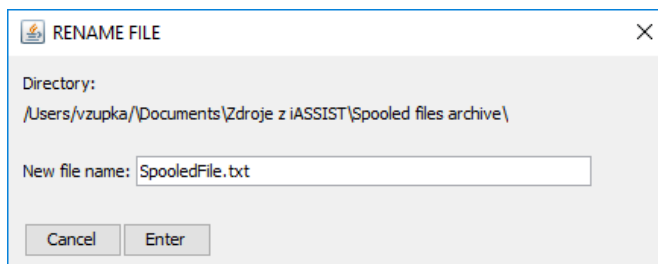
Delete

The commands *Display* and *Copy* transform the spooled file from its binary form to text, that is written into an internal file *SpoiledFile.txt* in directory *workfiles*. The *Display* command also displays the text in a separate window (see [Displaying](#) above).

The internal file can be copied under the same or another name into a PC directory of choice so that it can be further manipulated (e. g. printed or sent by e-mail). Select command *Insert spooled file* from the menu under a PC directory node in the left tree:



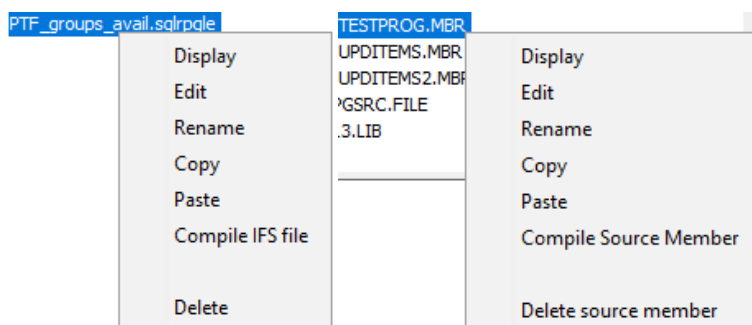
A dialog is shown prompting for a file name:



After changing or leaving the name and pressing the Enter key the file is written into the directory.

Compilation

The context menu on some nodes for IBM i files contains the command to compile:



For IFS stream files it is *Compile IFS file*, for source members it is *Compile source member*.

The command shows a window with information obtained from the source:

Compile source member VZTOOL2/QRPGLESRC(TESTPROG)

Source type: Compile command:

Compiled object Library: Object: Library prefix:

CRTBNDRPG PGM(*CURLIB/TESTPROG) SRCFILE(VZTOOL2/QRPGLESRC) OUTPUT(*PRINT)

The window has a few rows.

- The first row shows the path to the source file or the IFS stream file.
- The second row contains parameters necessary to identify input to compilation.
- The third row contains parameters necessary to identify output from compilation.
- The fourth row shows the actual text of the compilation command.
- The fifth row contains buttons:
 - Cancel* – cancel the work,
 - Perform command* – perform the compilation command,
 - Job log* – print the actual contents of the job log,
 - Spooled files* – invoke table of spooled files for the actual user,
 - Edit* – invoke a window with the source text to edit.

Explanation of the individual parts of rows follows.

Source type

The source type can be selected from the combo box or entered in the input field. Programs and description files of the following source types can be compiled:

CLLE, CLP,
RPG, RPGLE, SQLRPG, SQLRPGLE,
CBL, CBLLE, SQLCBL, SQLCBLLE,
C, CPP, SQLC, SQLCPP,
CMD,
DSPF, LF, PF, PRTF,
TBL

Source members

For source members, all suffixes are the same: *MBR*. The source type of a member is derived from the name of its source physical file. If the name is standard, the member gets its source type. If the name is not standard, the member gets type TXT. Standard names of source physical files are as follows:

QCLSRC	CLLE	Control language
QRPGLESRC	RPGLE	ILE/RPG
QRPGSRC	RPG	RPG/400
QCBLLSRC	CBLLE	ILE/Cobol
QCBLSRC	CBL	Cobol/400
QCSRC	C	C language
QCMDSRC	CMD	Command
QDDSSRC	PF	Physical file
QTBLSRC	TBL	Table

IFS stream files

For IFS stream files, suffixes listed above must be explicitly written, not necessarily with capital letters. IFS stream files of the following source types *cannot* be compiled:

CLLE, CLP, CMD, RPG, CBL, SQLRPG, SQLCBL
 DSPF, LF, PF, PRTF,
 TBL

Compile command

The actual compilation command can be left unchanged or it may need to be selected from the combo box, or entered in the input field. Changing is sometimes necessary because some source types can be compiled as a *program* or a *module* or a *service program*.

A program or a module can be created from the ILE source types: CLLE, RPGLE, CBLLE, C, CPP, SQLRPGLE, SQLCBLLE, SQLC a SQLCPP.

A service program can be created from source types with SQL statements: SQLRPGLE, SQLCBLLE a SQLC.

Change library list

This button invokes a window with an overview of the *user library list* and the *current library*:

Library prefix

Entering characters (e. g. VZT) in the input field *Library prefix* and pressing the Enter key writes a list of libraries whose names start with these characters to the *left* frame. If the input field is empty, the complete list of libraries is shown.

Current library

The current library can be selected from the *Current library* combo box or entered in the input field. The symbol *CRTDFT is a special entry which means that the job library list does *not* contain a current library.

Creating a user library list

A single library or multiple libraries can be selected from the left frame and copied to the right frame either by *drag and drop*, or by *Copy* → button. The right frame represents the *user library list* of the job.

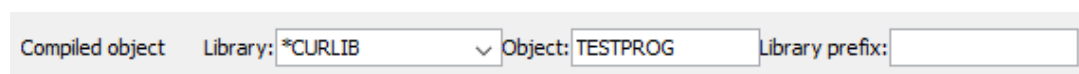
The *Remove* button removes selected libraries from the right frame.

The *Clear* button clears the right frame.

The *Save & return* button saves the changes.

Compiled object

The parameters in this line can be left unchanged, or entered, or changed according to the name and library under which the created object is to be stored.



The screenshot shows a form titled 'Compiled object'. It contains three input fields: 'Library:' with a dropdown menu showing '*CURLIB', 'Object:' with a text box containing 'TESTPROG', and 'Library prefix:' with an empty text box.

Library

The library can be selected from the combo box or entered in the input field.

Object

The name of the object created by compilation can be left unchanged or changed in the input field.

Library prefix

Entering characters in the input field and pressing the Enter key writes a list of libraries whose names start with these characters to the combo box.

Perform command

The *Perform command* button *runs the compilation*. One or more messages are reported about the result of the compilation in the low part of the window, e. g.:

RNS9304 *INFORMATIONAL: Program TESTPROG placed in library QGPL. 00 highest severity. Created on 02/08/17 at 14:09:16.

Cause : Program TESTPROG was successfully created in library QGPL. The highest message severity that resulted was 00. The program creation date and time are 02/08/17 and 14:09:16.

A compilation protocol (listing) is printed in the print file QPRINT.

Spooled files

The Spooled files button shows a table of spooled files for the *current user*. Selection, displaying, copying, and deleting is performed the same as in OUTQ object types (see [Spooled files](#) above). The window contains the *Refresh* button which refreshes the table of the spooled files without using the Spooled files button repeatedly.

File name	File num.	Job name	User	Job num.	Date	Time	
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text" value="v"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Refresh"/>

File name	File num.	Job name	User	Job num.	Date	Time	Output queue
CENY2	1	QPRTJOB	VZUPKA	029937	1170311	094622	QGPL/QPRINT

Job log

The *Job log* button prints the actual contents of the job log in the print file QEZJOBLOG. This file can be found using the Spooled files button and displayed like other text files.

Edit

The *Edit* button displays contents of the text for editing. The text can be changed, saved, and compiled again.