Replacing the RxMessageBox() RexxUtil Function (Windows, OS/2) with BSF4ooRexx for Windows, Linux and MacOSX

2019 – International Rexx Symposium

Hursley, September 2019

Rony G. Flatscher (Rony.Flatscher@wu.ac.at, http://www.ronyRexx.net)
Wirtschaftsuniversität Wien, Austria (http://www.wu.ac.at)

Overview

- RxMessageBox()
- BSF4ooRexx replacement
  - ooRexx class BSF.Dialog
- BSF4ooRexx enabled alternatives
  - javax.swing.JOptionPane
  - javafx.scene.control.Alert
  - Windows only: .Net dialogs! :-)
- Roundup

RxMessageBox(), 1

- Allows Rexx programmers to use a GUI popup dialog to communicate with the user
- Originally introduced with OS/2
  - Cf. "From Bark to Bytes", p. 42 (as of 2019-09-09):
    <https://archive.org/stream/GG2441990/GG24-4199-0%20-%20OS2%20REXX%20From%20Bark%20to%20Byte_djvu.txt>
  - Supported in the Windows version of ooRexx
    - Cf. ooRexx reference documentation (rexxref.pdf)
      - "8.3. RxMessageBox (Windows only)"
- Not available for Linux or MacOS

RxMessageBox(), 2
ooRexx Reference, 1

- Syntax RxMessageBox(text[,title][,button][,icon])
  - text: the string to be displayed to the user
  - title: optional message box title, defaults to "Error!"
  - button: optional, one of:
    - "OK" (default), "OKCANCEL", "RETRYCANCEL", "ABORTRETRYIGNORE", "YESNO", "YESNOCANCEL"
    - OS/2 in addition defines "ENTER" and "ENTERCANCEL", which are not available in the Windows implementation
  - icon: an icon is displayed in the dialog, one of:
    - "NONE" (default)
    - "ASTERISK" = "INFORMATION"
    - "EXCLAMATION" = "WARNING"
    - "HAND" = "STOP" = "ERROR"
    - "QUESTION" = "QUERY"

RxMessageBox(), 3
ooRexx Reference, 2

- Syntax RxMessageBox(text[,title][,button][,icon])
  - Returns a number, that indicates which button was pressed by the user
    - 1 (OK button)
    - 2 (CANCEL button): also, if ESC key got pressed instead, while the CANCEL button is displayed
    - 3 (ABORT button)
    - 4 (RETRY button)
    - 5 (IGNORE button)
    - 6 (YES button)
    - 7 (NO button)
    - Note: OS/2 defines the return value 8 if the ENTER button was pressed

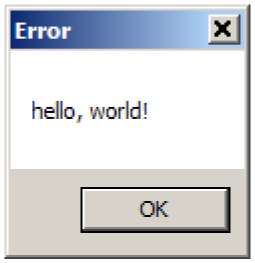
RxMessageBox(), 3
Example

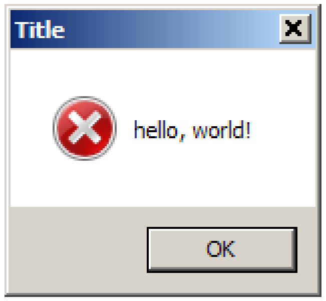
- Example

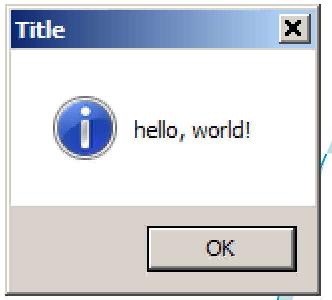
```
say "example #1:" rxMessageBox("hello, world!")
say "example #2:" rxMessageBox("hello, world!","Title")
say "example #3:" rxMessageBox("hello, world!","Title","ok")
say "example #4:" rxMessageBox("hello, world!","Title","ok","error")
say "example #5:" rxMessageBox("hello, world!","Title","ok","information")
say "example #6:" rxMessageBox("hello, world!","Title","ok","question")
say "example #7:" rxMessageBox("hello, world!","Title","ok","warning")
```

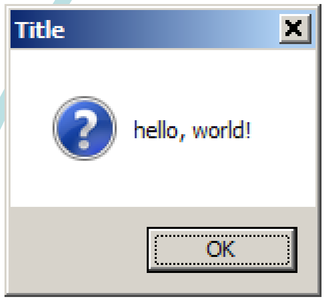
- Output

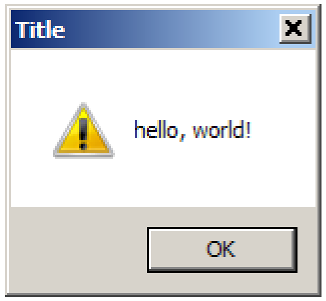
```
example #1: 1
example #2: 1
example #3: 1
example #4: 1
example #5: 1
example #6: 1
example #7: 1
```









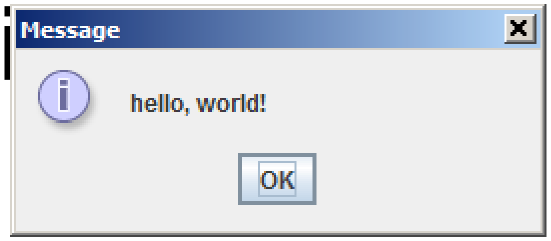


BSF4ooRexx, 1

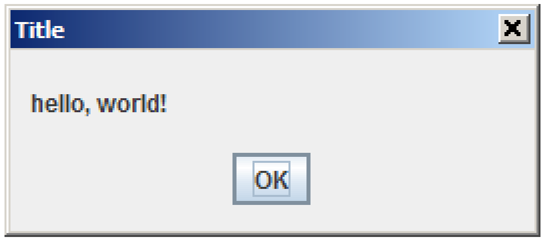
BSF.Dialog's messageBox()

- Importance of RxMessageBox() clear from day 1! :)
- BSF.CLS package (program) defines a public class BSF.Dialog with a method messageBox()
  - Simpler syntax
    - messageBox(text[,title][,type])
      - text: the string to be displayed to the user
      - title: optional, defaults to "Message"
      - type: one of "Information" (default, if neither title nor type supplied), "Warning", "Error", "Question"
        - Note: only first character needs to be supplied! :)
  - Always returns .nil

BSF4ooRexx, 2
Example 1, Wi



hello, world!



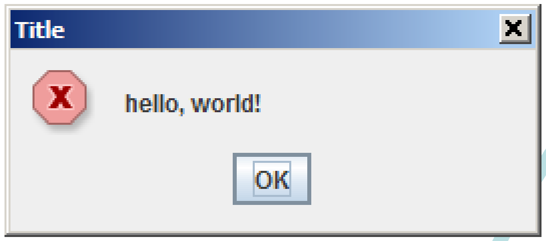
hello, world!

- Example

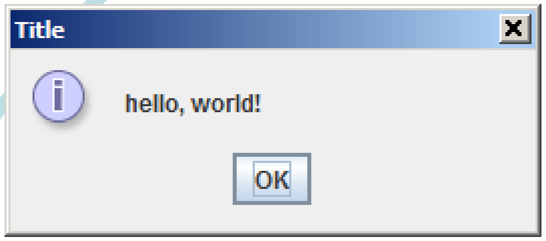
```
say "example #1:" .bsf.dialog~messageBox("hello, world!")
say "example #2:" .bsf.dialog~messageBox("hello, world!","Title")

say "example #3:" .bsf.dialog~messageBox("hello, world!","Title","error")
say "example #4:" .bsf.dialog~messageBox("hello, world!","Title","information")
say "example #5:" .bsf.dialog~messageBox("hello, world!","Title","question")
say "example #6:" .bsf.dialog~messageBox("hello, world!","Title","warning")

   -- place this directive at the end of your program
::requires "BSF.CLS" -- get the Java bridge, camouflage Java as ooRexx
```

- Output

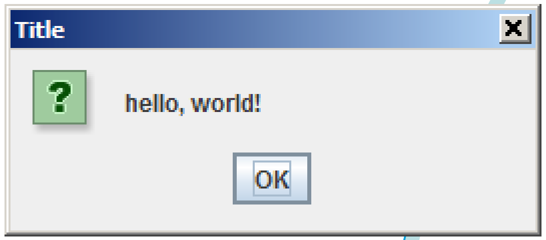
```
example #1: The NIL object
example #2: The NIL object
example #3: The NIL object
example #4: The NIL object
example #5: The NIL object
example #6: The NIL object
```



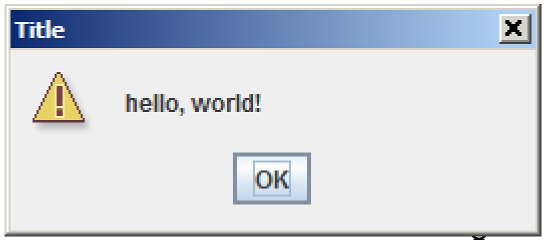
hello, world!



hello, world!

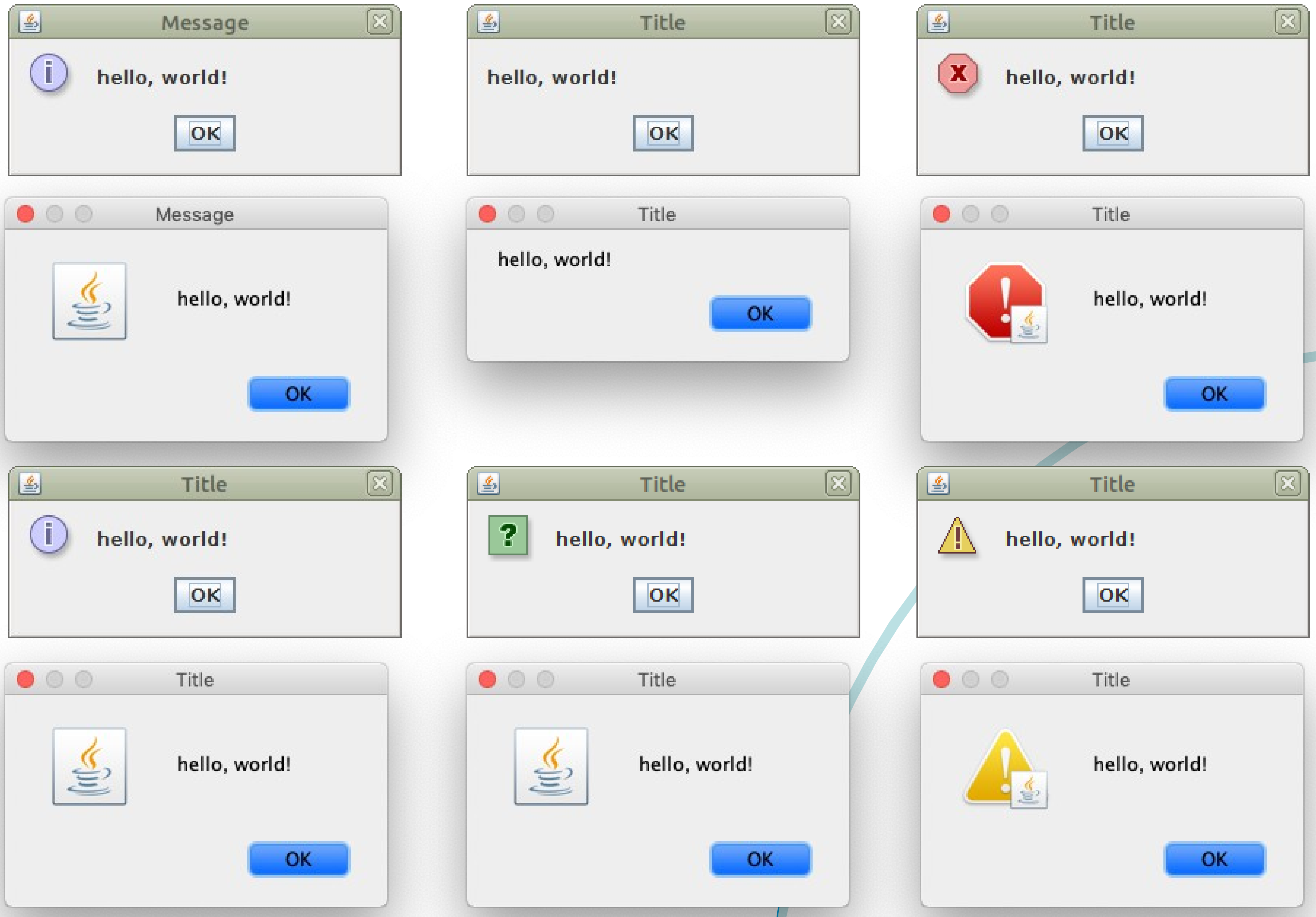


hello, world!



hello, world!

BSF4ooRexx, 3
Example 1, Linux and MacOS



Message

hello, world!

Title

hello, world!

Title

hello, world!

Message

hello, world!

Title

hello, world!

Title

hello, world!

Title

hello, world!

Title

hello, world!

Title

hello, world!

Title

hello, world!

Title

hello, world!

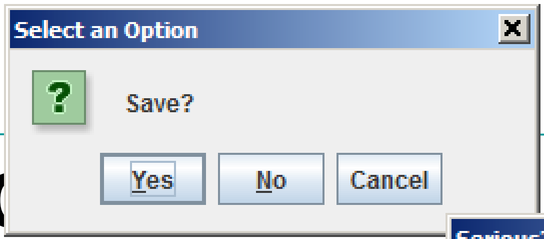
Title

hello, world!

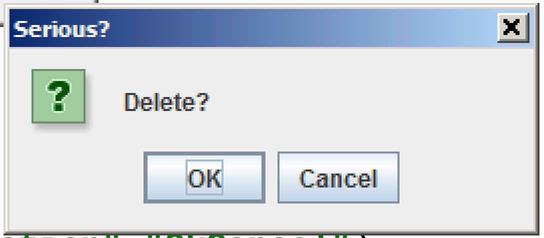
BSF4ooRexx, 4
BSF.Dialog's dialogBox()

- Wait, there is more …
  - dialogBox(text[,title][,type][,optionType][,icon][,buttons][,defButton])
    - text: the string to be displayed to the user
    - title: optional, defaults to "Select an Option"
    - type: one of "Information" (default, if neither title nor type supplied), "Warning", "Error", "Question"
      - Note: only first character needs to be supplied! :)
    - optionType: optional, or one of "default", "OkCancel" (default), "YesNo", "YesNoCancel"
    - icon: optional, e.g. a java.swing.ImageIcon
    - buttons: optional, collection of button names or blank delimited button names
    - defButton: optional, one of the button names that should be the default push button
    - Returns 0-based number of pressed button, counted from left to right
      - Returns -1 if ESC key or the X icon was pressed

BSF4ooRexx, 5
Example 2, Windo


Select an Option

Save?


Serious?

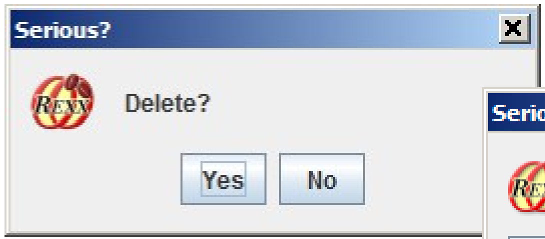
Delete?

- Example

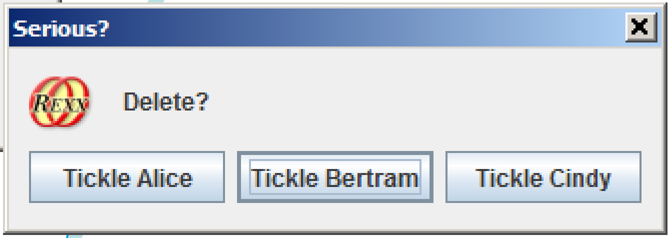
```
say "#1:" .bsf.dialog~dialogBox("Save?")
say "#2:" .bsf.dialog~dialogBox("Delete?","Serious?","question","OkCancel")

icon1=.bsf~new("javax.swing.ImageIcon", "bsf4oorexx_032.png")
say "#3:" .bsf.dialog~dialogBox("Delete?","Serious?","question","YesNo",icon1)

buttons=("Tickle Alice", "Tickle Bertram", "Tickle Cindy")
defButton=buttons[2]
icon2=.bsf~new("javax.swing.ImageIcon", "oorexx_032.png")
say "#4:" .bsf.dialog~dialogBox("Delete?","Serious?","question",,icon2, -
                                buttons,defButton)

   -- place this directive at the end of your program
::requires "BSF.CLS" -- get the Java bridge, camouflage Java as ooRexx
```

- Output, e.g.,

```
#1: 2
#2: 0
#3: 1
#4: -1
```

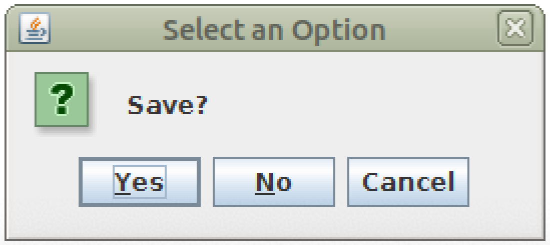

Serious?

Delete?

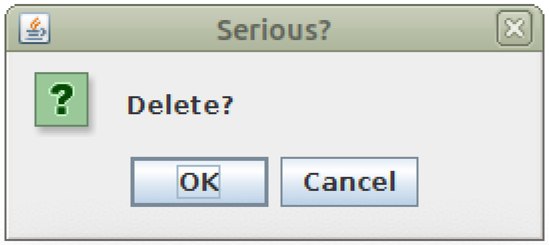

Serious?

Delete?

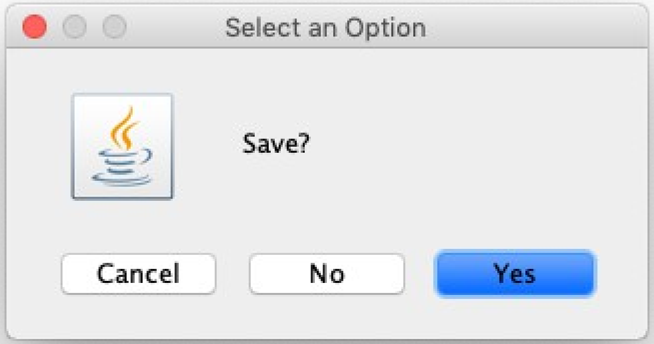
BSF4ooRexx, 6
Example 2, Linux and MacOS



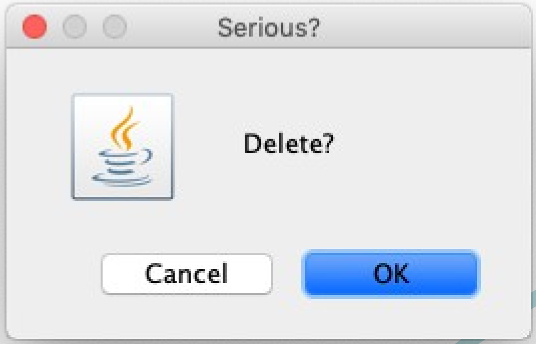
Save?



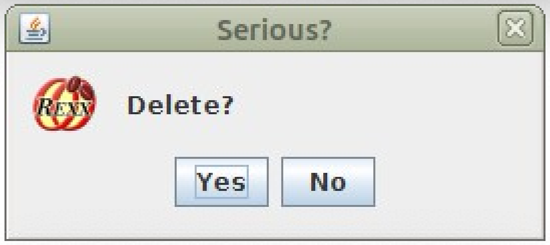
Delete?



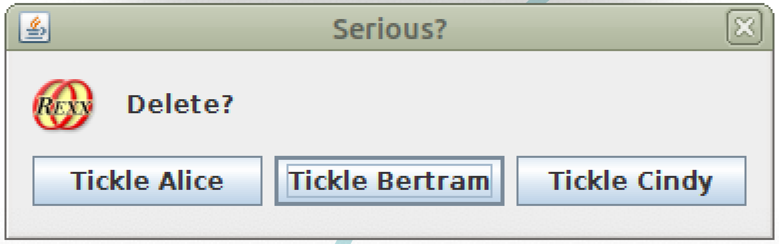
Save?



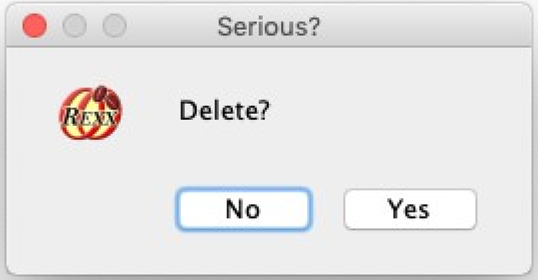
Delete?



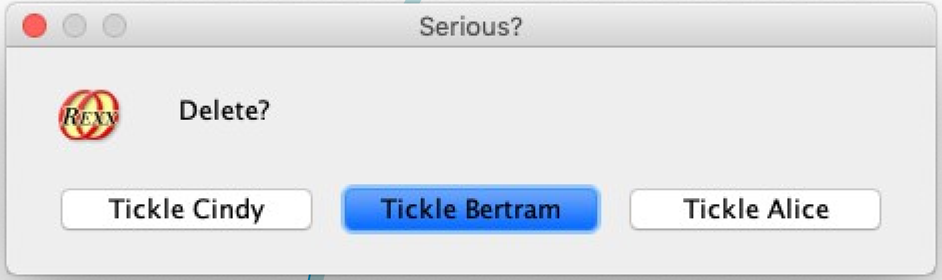
Delete?



Delete?



Delete?



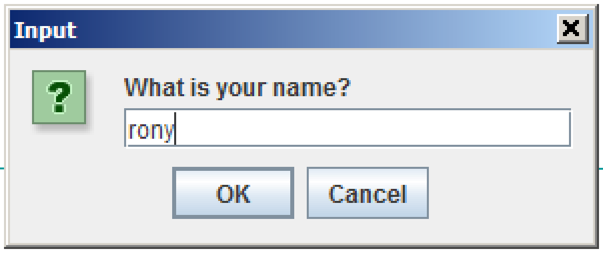
Delete?

BSF4ooRexx, 7

BSF.Dialog's inputBox()

- Wait, there is more …
  - inputBox(text[,defaultText])
  - inputBox(text[, ,type])
  - inputBox(text[,title][,type][,icon][,options][,defOption])
    - text: the prompt string to be displayed to the user
    - title: optional, defaults to "Input"
    - type: one of "Information" (default, if neither title nor type supplied), "Warning", "Error", "Question"
      - Note: only first character needs to be supplied! :)
    - icon: optional, e.g. a java.swing.ImageIcon
    - options: optional, collection of option names or blank delimited button names
    - defOption: optional, one of the option names that should be the default option
    - Returns input/option text
      - Returns .nil if ESC key or the X icon get pressed

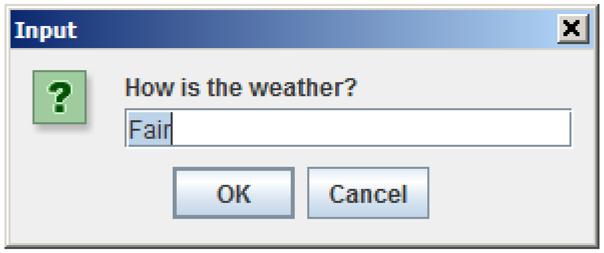
BSF4ooRexx, 8 Example 3, Windows



Input

What is your name?

rony



Input

How is the weather?

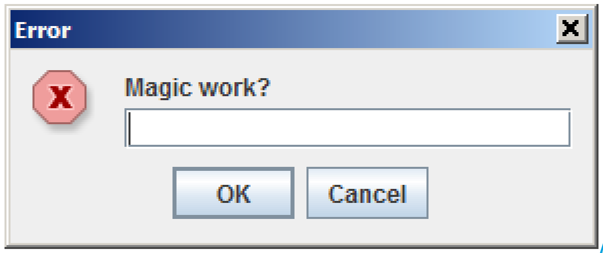
Fair

- Example

```
say "#1:" .bsf.dialog~inputBox("What is your name?")
say "#2:" .bsf.dialog~inputBox("How is the weather?", "Fair")
say "#3:" .bsf.dialog~inputBox("Magic work?", , "error")

icon1=.bsf~new("javax.swing.ImageIcon", "bsf4oorexx_032.png")
options=("Ask", "Just do it!", "Do nothing!")
defOption=options[2]
say "#4:" .bsf.dialog~inputBox("Please pick one!","Options","warning", -
                               icon1,options,defOption)

   -- place this directive at the end of your program
::requires "BSF.CLS" -- get the Java bridge, camouflage Java as ooRexx
```

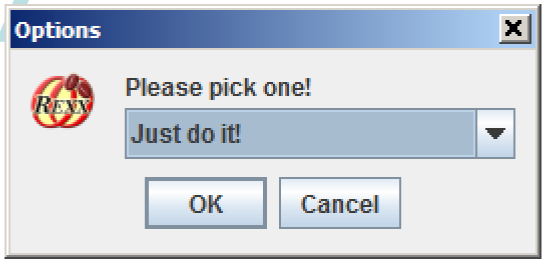
- Output, e.g.,

```
#1: rony
#2: Fair
#3: The NIL object
#4: Just do it!
```



Error

Magic work?

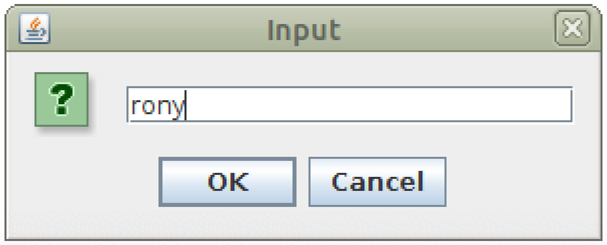


Options

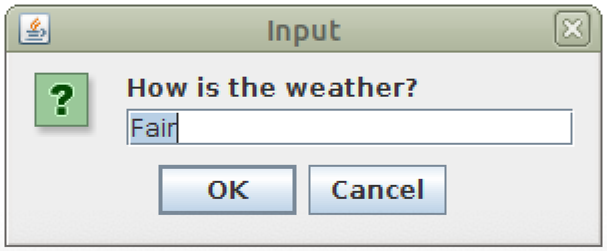
Please pick one!

Just do it!

BSF4ooRexx, 9
Example 3, Linux and MacOS

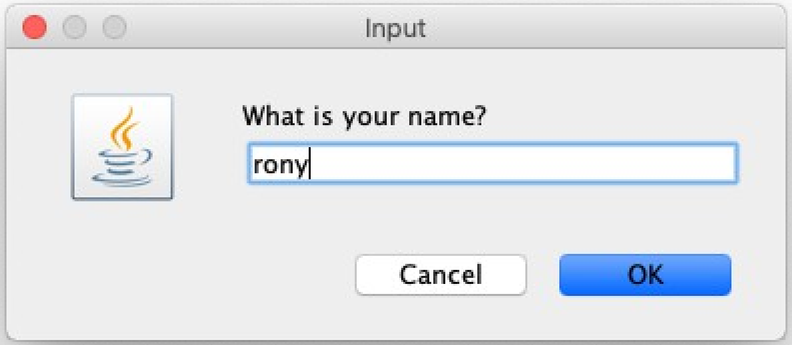


rony



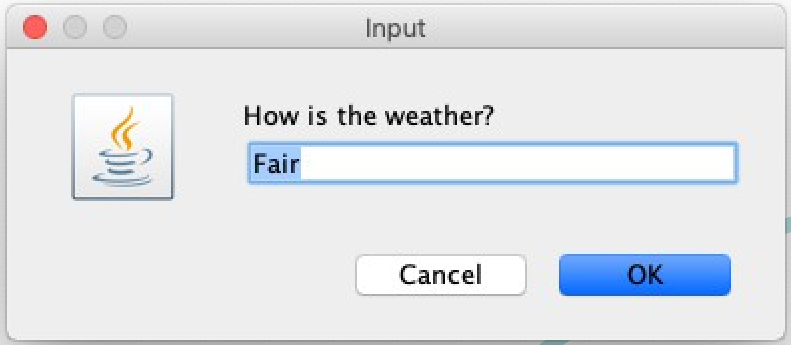
How is the weather?

Fair



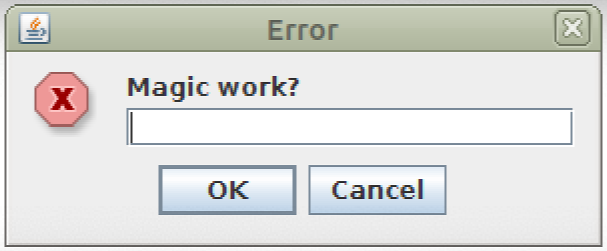
What is your name?

rony

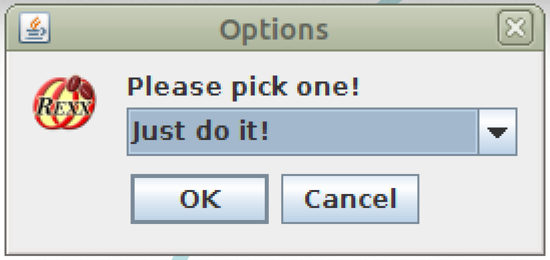


How is the weather?

Fair

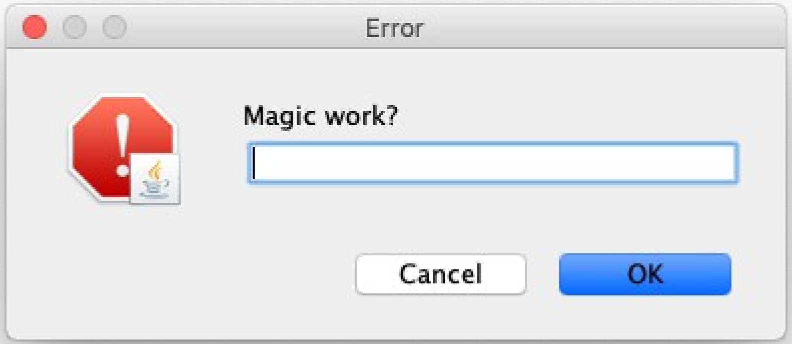


Magic work?

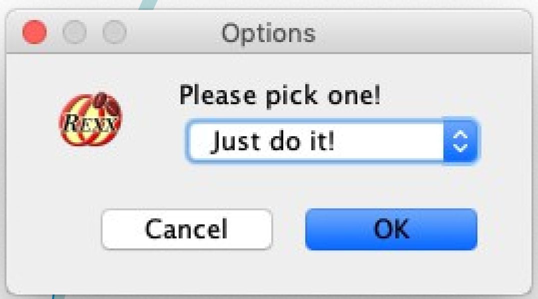


Please pick one!

Just do it!



Magic work?



Please pick one!

Just do it!

BSF4ooRexx, 10

BSF.Dialog

- Wait, there is more …
  - One can control where the dialog gets displayed
  - How?
  - Create an instance of the BSF.Dialog class
    - Supply the java.awt.Component component to which the dialog is related
      - If .nil then the "default frame" (the screen) is the parent
        - The dialog will be centered on the screen
      - The Component's java.awt.Frame will be located and its position used
    - Send the messageBox, dialogBox and inputBox message to that instance

BSF4ooRexx, 11
Example Positioning Dialog, 1

- Example

```
reh=.RexxHandler~new        -- Rexx event handler for Java events
jeh=BsfCreateRexxProxy(reh, ,                                  -
                        "java.awt.event.WindowListener", -
                        "java.awt.event.ActionListener")
frame=.bsf~new("java.awt.Frame", "Testing the .BSF.Dialog Class")

frame~addWindowListener(jeh)
frame~setLayout( .bsf~new("java.awt.FlowLayout") ) -- set layout manager
frame~add(.bsf~new('java.awt.Button', 'BSF.Dialog without parent')~~addActionListener(jeh))
btn=.bsf~new('java.awt.Button', 'BSF.Dialog with this frame as parent')~~addActionListener(jeh)
frame~add(btn)
frame ~~pack ~~setVisible(.true)~~toFront -- layout the Frame object, show it

::requires BSF.cls    -- load Object Rexx BSF support


 /* Rexx event handler which handles Window and Action events   */
::class RexxHandler
... continued on next page ...
```

BSF4ooRexx, 12
Example Positioning Dialog, 2

```
... continued ...
::class RexxHandler
::attribute closeApp            -- allow to get and set the control variable's value
::method init                   /* constructor */
  expose closeApp count         -- used as control variable
  closeApp  = .false
  count     = 0                 -- dialog counter

::method unknown                -- intercept unhandled events, do nothing

::method waitForExit            -- blocking (waiting) method
  expose closeApp
  guard on when closeApp=.true-- blocks (waits) until control variable is set to .true

::method windowClosing          -- event method (from WindowListener)
  expose closeApp
  closeApp=.true                -- change control variable to unblock

::method actionPerformed        -- button Action event
  expose count
  use arg eventObject

  button=eventObject~source     -- get the button object
  say "button="pp(button~toString)
  count+=1
  if button~label="BSF.Dialog without parent" then    -- a non-modal dialog
     .bsf.dialog~messageBox("BSF.Dialog #" count "(centered on screen)")
  else       -- create a modal dialog for the frame itself
  do
     bd=.bsf.dialog~new(button) -- or directly the Frame object of the button
     bd~messageBox("BSF.Dialog #" count "(dialog with the frame as parent)")
  end
```

BSF4ooRexx, 13

BSF.Dialog's

- Nutshell sample
  - samples/1-020_demo.BSF.dialog.rxj
  - Hint: point your browser to the index.html file there
- Implementation
  - BSF.CLS
  - Employing the Java class javax.swing.JOptionPane
    - Documentation search with "*javadoc JOptionPane*"
    - E.g. Java 6 version (as of 2019-09-11)

      https://docs.oracle.com/javase/6/docs/api/javax/swing/JOptionPane.html

BSF4ooRexx, 14

JavaFX Alert class

- Nutshell sample
  - samples/JavaFX/javafx_dialog_demo.rxj
  - Hint: point your browser to the index.html file there
- Implementation
  - Employing the Java class javafx.scene.control.Alert
    - Documentation search with "*javadoc javafx alert*"
    - E.g. Java 8 version (as of 2019-09-11)

      https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Alert.html

BSF4ooRexx, 15

- Not happy?
- You want a snap-in solution??
- O.K., do it yourself ! :)
  - And if you do, please share it with the Rexx community!

BSF4ooRexx, 16
A Snap-in Solution? Do it yourself !

- Possible to create your own snap-in RxMessageBox() function! :)
- Create a package (program), e.g. "rxfuncs.rex" that defines a public routine "RxMessageBox"
  - Use e.g. s javax.swing.JOptionPane or javafx.scene.control.Alert
  - Process all RxMessageBox() arguments accordingly
  - Return the number of the pressed key
- Add the directive ::requires "rxfuncs.rex" at the very end of your Rexx program
- That's it! :)

BSF4ooRexx, 17 Package/Program "rxfuncs.rex"

- Add your own Rexx code after the routine directive that implements the RxMessageBox() functionality
- Ask questions on the BSF4ooRexx support mailing list!
- Package/program "rxfuncs.rex" initially may look like

```
/* a package (program) "rxfuncs.rex" with useful public routines */

   /* this directive makes all of Java available to us   */
::requires "BSF.CLS" -- get the Java bridge, camouflage Java as ooRexx

/* this is YOUR implementation of the snap-in rxMessageBox-function   */
::routine RxMessageBox public
   -- ... your Rexx implementation goes here ...
   -- ... your Rexx implementation goes here ...
   -- ... your Rexx implementation goes here ...
  return keyNumber   -- you would return a number between 1 and 7
```

Roundup

- The BSF4ooRexx BSF.Dialog class
  - A utility class defined in BSF.CLS
  - Allows platform independent dialogs that are easy to use
  - May be used to replace RxMessageBox() calls
    - BSF.Dialog not compatible with RxMessageBox() but very similar
    - BSF.Dialog adds additional features that are typically needed in dialogs
  - Everyone could create a compatible snap-in implementation by studying the BSF.Dialog class' usage of the java class javax.swing.JOptionPane which gets used to realize the dialogs
    - Alternatively one may use the modern javafx.scene.control.Alert Java class to implement the RxMessageBox() features

URLs

- RexxLA-Homepage (non-profit SIG, owner of ooRexx, BSF4ooRexx)

  <http://www.rexxla.org/>
- ooRexx 5.0 beta on Sourceforge

  <https://sourceforge.net/projects/oorexx/files/oorexx/5.0.0beta/>
- BSF4ooRexx on Sourceforge (ooRexx-Java bridge)

  <https://sourceforge.net/projects/bsf4oorexx/>
- Introduction to ooRexx (254 pages)

  <https://www.facultas.at/Flatscher>
- JetBrains "IntelliJ IDEA", powerful IDE for all operating systems
  - <https://www.jetbrains.com/idea/download>, free "Community-Edition"
  - Alexander Seik's ooRexx-Plugin with readme (as of: 2019-08-27)
    - <https://sourceforge.net/projects/bsf4oorexx/files/Sandbox/aseik/ooRexxIDEA/beta/1.0.5>