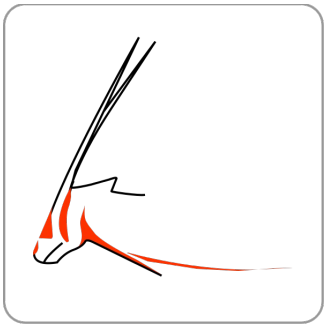
Introduction to Rexx and ooRexx

From Rexx to Open Object Rexx (ooRexx)

Rony G. Flatscher



”REXX was invented to make programming easier than ever before. Therefore the programming language was designed to be ‘human centric’ and, as a result, it can be learned and understood very quickly!”

“ooRexx adds all the object-oriented bells and whistles, yet keeps the language ‘human centric’.”

“This book gives you an incredibly useful, productive and durable ‘Swiss Army Knife’ (SAK) kind of programming tool. You can quickly create glue code, scripts, macros and even full blown applications for any operating system and deeply understand Rexx and ooRexx code easily and clearly.”

Rony G. Flatscher works as a professor for Information Systems (German: “Wirtschaftsinformatik”) at the *WU Wien*, Austria, a business university with approximately 25,000 students.

He has been experimenting over a decade teaching programming to end-users of information systems using various programming languages. In the course of time, a lecture of two consecutive classes was developed which succesfully introduces the students to programming, object-oriented programming and scripting/remote-controlling business applications such as Microsoft Office or (in an operating system independent manner) Apache OpenOffice.

One *key element* that allows for this to be done in a very short time is the choice of programming language: *“Open Object Rexx (ooRexx)”*, a human centric, easy to understand and easy to use programming language that originates in the IBM product “Object REXX” which was handed over to the open-source community.

This book introduces the principles of the programming language ooRexx in a very concise manner. It demonstrates all of the introduced concepts immediately with *“nutshell examples”* (very small programs) showing their output, which allows anyone to study and learn the language *by just reading* this introductionary book.

Author's ooRexx related URL: `http://www.RonyRexx.net`

Howard Fosdick (USA), the author of “Rexx – Programmer's Reference” says about this book:

> *“Excellent work! This is the book we all have been waiting for. I only wish it had been available when I was learning ooRexx!*
>
> *...*
>
> *I'd like to conclude by emphasizing that this just is a really fantastic book, Rony, written by the one person who is so well qualified to write it. Congratulations!”*

Rony G. Flatscher

Introduction to Rexx and ooRexx

From Rexx to Open Object Rexx (ooRexx)

First Edition: 1.00, 2013-02-15 (Version 100.20130215)

http://www.RonyRexx.net

c/o WU Wien, Augasse 2-6, A-1090 Wien

Printed by: Facultas Verlags- und Buchhandels AG, Austria, Europe

Acknowledgements

The author thanks the following persons for their feedback, proof reading and help in creating free Rexx-related art (in alphabetical order):

Gilbert *Barmwater* (U.S.A.): feedback, proof reading

Daniel A. *Flatscher* (Austria): proof reading

Howard *Fosdick* (U.S.A.): feedback, proof reading

René Vincent *Jansen* (The Netherlands): feedback, proof reading

Les *Koehler* (U.S.A.): feedback, proof reading

Gerald *Leitner* (Austria): feedback, proof reading

DI Walter *Pachl* (Austria): feedback, proof reading

Graham *Wilson* (South Africa): art including icons for BSF4ooRexx

Jon “Sahananda” *Wolfers* (United Kingdom): feedback, proof reading

Foreword

A Brief History of the `Rexx programming language`. In 1979 *Mike F. Cowlishaw* (MFC), an English gentleman working for IBM, devised a "human centric" programming language for the IBM mainframes that was easier to understand and to program than the arcane mainframe batch language named `Exec 2`. The design work was carried out at the IBM Research facilities in Hursley under the management of *Dr. Brian Marks*. It was probably the first time in the history of programming language design that IBMers interconnected worldwide via the IBM internal network were able to influence the design by studying the distributed specifications and giving feedback, like *Les Koehler* from IBM USA.

IBM later defined the `REXX` programming language to be the strategic batch/scripting language on all of its operating systems via IBM's `SAA` (System Application Architecture) standard. Another outstanding IBM employee who has probably been the only person to create Rexx interpreters multiple times for multiple operating systems is *Rick McGuire,* who led the development and maintenance of the `IBM SAA REXX` interpreters.

The IBM lab in Vienna created a Rexx compiler for its mainframe REXX (*Klaus Hansjakob*, *Walter Pachl*), which is being sold and maintained by IBM to this very day.

Mike F. Cowlishaw documented the REXX language in a book named "The Rexx Language" also known as "TRL" and he later became one of the few IBM Fellows[1] due to his continuing innovative and influential work (he is also attributed to be the person who made Java a strategic language and platform within IBM, having ported Java to the IBM OS/2 PC operating system in the 90'ies).

The high impact of the Rexx language can be witnessed by the appearance of numerous non-IBM implementations of the Rexx language, e.g. `Regina` (open source, *Anders Christensen*, *Mark Hessling*), `Rexx/imc` (open source, *Ian M. Collier*) or `BREXX` (open source, *Vassilis N. Vlachoudis*), but also proprietary and commercial Rexx interpreters like `ARexx` (part of the Amiga operating system), Novell Netware's Rexx (in the 90'ies), Workstation Unix

[1] An IBM Fellow is free to research and to work, very much like professors at Universities, who have the freedom to freely determine what they research and what they teach.

Rexx. Some pointers to various Rexx interpreters can be found at http://www.rexxla.org/rexxlang/mfc/rexxplat.html.

ANSI/INCITS `REXX` Standard. In 1996 the American National Standards Institute (ANSI) working group X3J18 finalized the "American National Standard for Information Systems – Programming Language REXX". After ANSI got renamed to "INCITS (InterNational Committee for Information Technology Standards, http://www.incits.org/)" the respective standard was named "INCITS 274 :1996 [R2001]". In 2007 the INCITS 274 REXX standard was extended for another period of ten years, reflecting the importance of the Rexx language in the industry.

The `INCITS 274 REXX` standards on decimal arithmetic served as the basis for defining decimal formats in `IEE 754-2008` and `ISO/IEC/IEEE 60559:2011`. An ANSI C implementation has been created by *Mike F. Cowlishaw* (http://www.speleotrove.com/mfc/), who on behalf of IBM has been a driving force behind standardizing decimal arithmetics in the context of IEEE, Java `JSR-13` and who also implemented an open source `decNumber` package in ANSI C.

A Brief History of the ooRexx Language. At the end of the 80'ies Dr. Brian Marks oversaw another interesting project, named "Oryx" with the technical lead of *Simon Nash*. The aim of this project was to experiment with a Rexx interpreter that extends the Rexx language with object-oriented features. This work would later lead, under the auspices of *Rick McGuire,* to the IBM product "Object REXX". It was first distributed with `OS/2 Warp 4` in 1997, versions for IBM AIX and MS Windows were created and sold as well by IBM.

In 2004 IBM handed the source code of "Object REXX" over to the non-profit special interest group (SIG) "Rexx Language Association (RexxLA, http://www.RexxLA.org)". RexxLA published the first open source version of IBM's "Object REXX" as "Open Object Rexx 3.0 (http://www.ooRexx.org)" in 2005. The lead architect of this now open-source project has been *Rick McGuire*, who has been working in his own time on the open source version of ooRexx ever since.

What Is ooRexx?

- *A "classic Rexx" interpreter.* ooRexx runs any "classic Rexx" program and can be used to write "classic Rexx" programs. There is no need to use any of its new features that extend the Rexx language.

- *An object-oriented Rexx language, hence "ooRexx":* ooRexx comes with many useful classes (data types) and offers state-of-the art object-oriented features, devised in a "human-centric" way. Among other great things, ooRexx makes it easy to create multithreaded Rexx programs!
- *Fast and powerful:* ooRexx is a fast Rexx interpreter. It is a very powerful interpreter, which can be invoked from C++ or Java (via BSF4ooRexx) to allow Rexx and ooRexx macros/programs to run with C++ and Java applications. For C++ and Java applications it is possible to run multiple ooRexx interpreter instances in the same process space, each of which may execute even multithreaded Rexx code!
- *Great documentation:* IBM not only donated the source code for open sourcing to RexxLA, but also the excellent and professional technical documentation, which has been kept up-to-date. All the ooRexx documentation is available in the form of HTML and PDF-files, which can be nicely printed as books. The documentation is also directly available via the Internet: http://www.oorexx.org/docs/.
- *Free and open source:* originally created by IBM and marketed as "Object REXX", RexxLA received the sources for publishing, maintaining, and enhancing this powerful Rexx interpreter. RexxLA distributes ooRexx with source code for free: http://www.ooRexx.org.
- *Multiplatform:* ooRexx is available in 32 and 64 bit versions for the operating systems AIX, Linux, MacOSX, Windows, and can be built for any Unix implementation. Rexx programs written in one operating system environment can execute in any other operating system environment.
- *Extensible:* ooRexx comes with a powerful and easy to use C++ API which is documented in one of the accompanying ooRexx documentation PDFs (cf. rexxpg.pdf). This allows you to extend ooRexx with functions and methods implemented in C++, but also to bridge Rexx with other programming infrastructures like Java (cf. "BSF4ooRexx"). In addition it allows C++ applications to create Rexx interpreter instances which execute Rexx programs. This way it is fairly easy/simple to employ ooRexx as a macro language for any C++

applications. The BSF4ooRexx extension package provides the same functionality for Java applications.

The author wishes to acknowledge the following persons important to the Rexx world in the context of RexxLA in alphabetic order: *Gil Barmwater* (vice president), *Mike F. Cowlishaw* (honorary board member), *Chip Davis* (past president), *Mark Hessling* (board member), *René Vincent Jansen* (current president), *Les Koehler* (secretary/treasurer), *Lee Peedin* (past president), *Pam Taylor* (board member), *Jon "Sahananda" Wolfers* (board member).

Of course all the developers of ooRexx (including past) are acknowledged hereby (in alphabetic order): *David Ashley*, *Jean-Louis Faucher*, *Mark Hessling*, *Moritz Hoffmann*, *Rick McGuire*, *Mark Miesfeld*, *Lee Peedin*, *David Ruggles*, *Bruce Skelly*, *Rainer Tammer*, *Jon Wolfers*.

About this Book

This book introduces the programming language `Open Object Rexx`, also known as "`ooRexx`" in two steps:

1. Chapter 1 'The Rexx Language (“Everything Is a String”)' introduces the `Rexx` programming language that was created in 1979 by the IBM employee Mike F. Cowlishaw who later became an IBM Fellow due to his work on `Rexx`. The most important design philosophy for the language was the principle of "human-orientation", making it easy for programmers to create programs in the `Rexx` language compared to the arcane IBM mainframe batch language `Exec 2` which `Rexx` successfully replaced. One key success factor of the `Rexx` language has been its easy English-like syntax that makes it easy to learn, fast to comprehend, easy to apply and inexpensive to maintain. `Rexx` programs can be read almost like prose. As `ooRexx` is backwardly compatible to `Rexx` it can be used to learn `Rexx` and thereby the fundamentals of programming. The concepts in this chapter apply generally to all existing Rexx interpreters, which sometimes are called “classic Rexx” interpreters (as opposed to `ooRexx`, which is a leading edge `Rexx` interpreter that extends classic `Rexx` nicely into the object-oriented world). `ooRexx`-only features are highlighted in the text.

2. Chapter 2 'Extensions to the Rexx Language by ooRexx' documents the `ooRexx`-only keyword instructions (`LOOP`, `RAISE`, `USE`) and enhancements to the `Rexx` language like short hand assignment operators (e.g. “`+=`”) and the directives `::routine` and `::requires` that may prove quite helpful to “classic Rexx” programmers.

3. Chapter 3 'The ooRexx Language (“Everything Is an Object”)' builds upon the previous chapters and introduces the fundamental concepts of what is known as the "object-oriented (OO) paradigm". This is followed by an overview of the numerous new classes (data types) that come with `ooRexx` and which could be exploited by `Rexx` programmers to ease their programming life considerably in most cases. At the end of this chapter the reader should understand the OO-concepts and be able to take advantage of these new, powerful features!

4. Chapter '4 Reaching Out with ooRexx' opens with useful information about the ooRexx runtime system, followed by a categorized overview

of the ooRexx classes (data types, types) that are installed with the interpreter. The ooRexx programmer can directly use these ooRexx classes and take advantage of the features they implement.

5. Chapter 5 'Advanced Topics' introduces the interested reader to defining and implementing Rexx classes (data types), which is very easy and straight-forward. For those programmers who need the ability to create `Rexx` programs in which parts are executed concurrently, there is a concluding section which explains and demonstrates how easy it is to do that with `ooRexx`.

The structure and contents of the book are aimed at people who are interested in learning programming in `Rexx` and afterwards `ooRexx`. Still, it aims to introduce and demonstrate the concepts in a very concise, yet understandable manner. The reader is advised to consult the excellent `ooRexx` reference documentation, which completely documents `ooRexx` and is available as a nicely formatted PDF-book, named `rexxref.pdf` (`http://www.oorexx.org/docs/rexxref/rexxref.pdf`).

The way this book is written should also allow professional programmers to skim the book and learn about the fundamentals of `Rexx` and `ooRexx` by looking out for the definition boxes that are formatted like this:

> This is how a definition box is formatted. Definition boxes allow you to quickly get (re-)acquainted with the fundamental concepts that are taught in a chapter. This book will sometimes directly use the definitions of the ANSI/INCITS Rexx standard if possible in this book's context.

In addition, numerous little "nutshell programs" or "code snippets" demonstrate how to apply the introduced concepts. These programs, as short as they may seem, are full programs that can be executed as is by the `ooRexx` interpreter, yielding the output that is sometimes depicted alongside the program as well. "Nutshell programs" are formatted like this:

```
say "Hello world, this is Rexx speaking"
```

The above program will output the string *`Hello world, this is Rexx speaking`*.

Alternatively, ooRexx for Windows comes with a GUI program (menu entry named “`Try Rexx (GUI)`”) which allows you to enter Rexx code and execute it with the push of a button. ooRexx users on Linux or MacOSX might want to

install BSF4ooRexx[2] (https://sourceforge.net/projects/bsf4oorexx/files/GA/) which comes with a comparable GUI program (menu entry named "GUI RexxTry Program (ooRexxTry.rxj)").

Finally, ooRexx can be downloaded for free from one of the following locations:

- http://www.oorexx.org/download.html
- https://sourceforge.net/projects/oorexx/files/

There are editors that support Rexx syntax highlighting, for example the following two free and open source editors:

- "The Hessling Editor (THE)", which uses Rexx as its macro language, URL: http://hessling-editor.sourceforge.net/
- "vim (vi improved)", a part of many Linux distributions, is generally available for all operating systems, URL: http://www.vim.org/

[2] This GUI program is also available for Windows, if "BSF4ooRexx" gets installed there. "BSF4ooRexx" is an ooRexx external function package that allows ooRexx programs to interact directly with Java, which gets camouflaged as ooRexx. At the time of writing this external function package is available for Linux, MacOSX and Windows. Cf. 4.1 Exploiting Java on All Platforms, p. 159 below.

Table of Contents

List of Figures

List of Tables

List of Codes (Nutshell Examples)

List of Codes (Nutshell Examples)

List of Codes (Nutshell Examples)

W

:

.

~

Some oo|Rexx-Related World-Wide-Web Links

Author's ooRexx related homepage

Latest information on ooRexx books, projects, updates, seminars, news: `http://www.RonyRexx.net`

Rexx Language Association (RexxLA)

Not-for-profit, special interest group (SIG), owner of *BSF4ooRexx*, *NetRexx* and *ooRexx*, organizer of the annual “International Rexx Symposium”: `http://www.RexxLA.org`, `http://www.rexxla.org/events/`

ooRexx

Homepage: `http://www.ooRexx.org`

Download: `http://sourceforge.net/projects/oorexx/files/oorexx/`

`rexxref.pdf`: `http://www.oorexx.org/docs/rexxref/rexxref.pdf`, ooRexx PDF reference documentation

HTML reference : `http://www.oorexx.org/docs/rexxref/book1.htm`

BSF4ooRexx (Camouflages Java as ooRexx)

Homepage: `http://sourceforge.net/projects/bsf4oorexx/`

Download: `http://sourceforge.net/projects/bsf4oorexx/files/GA/`

Further information about the Rexx family:

Mark Hessling's Rexx Function Packages and the Regina Rexx Interpreter: `http://www.Rexx.org`

Howard Fosdick's Rexx Information: `http://www.RexxInfo.org`

Some *Rexx* Aware *Editors*:

THE (*The Hessling Editor*), uses Rexx and ooRexx as its macro language: `http://hessling-editor.sourceforge.net/`

vim (*vi improved*, available for practically all operating systems), includes Rexx and ooRexx syntax highlighting: `http://www.vim.org`

Student's Work Related to Rexx and BSF4[oo]Rexx including AOO/OOo/LO:

`http://wi.wu.ac.at/rgf/diplomarbeiten/`

Rony G. Flatscher works as a professor for Information Systems (German: “Wirtschaftsinformatik”) at the *WU Wien*, Austria, a business university with approximately 25,000 students.

He has been experimenting over a decade teaching programming to end-users of information systems using various programming languages. In the course of time, a lecture of two consecutive classes was developed which succesfully introduces the students to programming, object-oriented programming and scripting/remote-controlling business applications such as Microsoft Office or (in an operating system independent manner) Apache OpenOffice.

One *key element* that allows for this to be done in a very short time is the choice of programming language: *“Open Object Rexx (ooRexx)”*, a human centric, easy to understand and easy to use programming language that originates in the IBM product “Object REXX” which was handed over to the open-source community.

This book introduces the principles of the programming language ooRexx in a very concise manner. It demonstrates all of the introduced concepts immediately with *“nutshell examples”* (very small programs) showing their output, which allows anyone to study and learn the language *by just reading* this introductionary book.

Author's ooRexx related URL: `http://www.RonyRexx.net`

Howard Fosdick (USA), the author of “Rexx – Programmer's Reference” says about this book:

> *“Excellent work! This is the book we all have been waiting for. I only wish it had been available when I was learning ooRexx!*
>
> *...*
>
> *I'd like to conclude by emphasizing that this just is a really fantastic book, Rony, written by the one person who is so well qualified to write it. Congratulations!”*