История вопроса

Сайт создан в мае 2009 года, после выхода для тестирования, новой версии REBOL 3. Его цель - зафиксировать в интернете редкие факты интереса к этой технологии в России до настоящего времени. Есть основания считать, что проект рано или поздно займет свое особое место в образовании и использовании информационных технологий.

Как видим из ссылок справа "реболюционеров" в России пока нет. Есть с десяток сочувствующих и то, самый активный rebolist с Украины, открывший журнал в ЖЖ RU_REBOL Были ребята из Твери, откуда родилась идея сообщества REBOL.RU, но проект заглох.

Есть ответ на вопрос - почему все так медленно. Хорошая русская пословица гласит: "Тише едешь - дальше будешь!". Принципы, заложенные в проекте требуют "семь раз отмерять, прежде чем, резать." Представьте себе, что вы вдруг проснулись волшебником. Осознание самого факта, что вы теперь можете все, неожиданно заставляет вас задуматься, а что же сделать в первую очередь? Что важнее всего? То есть вы становитесь самодостаточны, мир приобретает целостность и суета вокруг вас, вас больше уже не волнует. Вы ищете гармонию, вы становитесь художником, то есть больше, чем только ученым или мастером кодирования. (Продолжение ...)

Старые блоги Карла Сассенрата (новые активны и ссылки на них есть везде, а вот это уже история ...) - http://www.rebol.blogspot.com/, http://www.rebol3.blogspot.com/
Также существуют странные домены второго уровня ... на будущее? - http://www.rebolpress.com/, http://www.reboltech.com/ ...
И ещё ... неудачные эксперименты (http://ru.rebol.wikia.com/wiki/REBOLution_in_RUSSIA всё бы ничего, но реклама ...) , а также попытка создать тексты с российским проектом "Приложение на REBOL".

Дублирующие ресурсы:
http://ru.rebol.wikia.com/wiki/REBOLution_in_RUSSIA
https://ru.wikibooks.org/wiki/REBOL_3

Январь 1999

Август 1999

Август 2000

Декабрь 2000

Апрель 2002

Лето 2002

Сентябрь 2002

Октябрь 2002
Июль 2003
Декабрь 2004
Май 2005
Сентябрь 2005
Декабрь 2005
Март 2006
Июль 2006
Август 2006
Февраль 2007
Апрель 2007
Июнь 2007
Август 2007
Октябрь 2007
Февраль 2008
Март 2008
Ноябрь 201

rebol3 - glossary

A: about, abs, absolute, action!, action?, add, ajoin, alias, all, also, alter, and, and~, any-block?,any-function?, any-path?, any-string?, any-word?, any, append, apply, arccosine, arcsine, arctangent,array, as-binary, as-pair, as-string, ascii?, ask, assert, at, attempt

B: binary!, bitset!, back, binary?, bind?, bind, bitset?, block, block?, block!, body-of, break, browse,bugs

C: call, concept, caret-to-offset, case, catch, cause-error, cd, change-dir, change, char?, char!, chat,checksum, clean-path, clear, close, closure?, closure, closure!, collect, comment, complement,compose, compress, confirm, construct, context, continue, copy, cosine, create, cursor

D: datatype?, datatype!, date?, date!, debase, decimal?, decimal!, decloak, decode-url, decode,decompress, dehex, delect, delete, deline, delta-profile, delta-time, demo, detab, difference, dir?, dirize,disarm, divide, do-codec, do-events, do, docs, does, dp, draw, dt, dump-obj, dump

E: echo, effect, eighth, either, else, email?, email!, empty?, enbase, encloak, encode, end!, enline,entab, equal?, error, error?, error!, eval, even?, event?, event!, evoke, exclude, exit,,exists?, exp, expression, extend, extract

F: fifth, file, file?, file!, find-module, find-script, find, first+, first, for, forall, foreach, forever, form,format, forskip, found?, fourth, frame?, frame!, func, funco, funct, function, function?, function!, functor

G： get-env, get-path?, get-path!, get-word?, get-word!, get, gob?, gob!, graphics, greater-or-equal?,greater?, GUI

H： halt, handle-events, handle?, handle!, has, head?, hash!, head, help

I： if, image?, image!, integer!, import, in-dir, in, index?, info?, input, insert, install, integer?,intersect, issue?, issuse!

J： join

L： last, latin1?, launch, layout, length?, lesser-or-equal?, lesser?, library?, library!, license, list!, list-dir, lit-path?, lit-path!, lit-word?, lit-word!, load-gui, load, log-10, log-2, log-e, logic?, logic!, loop,lowercase, ls

M： make-dir, make-scheme, make, map-event, map-gob-offset, map, map?, map!, math, max, maximum-of, maximum, min, minimum-of, minimum, mkdir, mod, modified?, module, module?, module!,modulo, mold, money?, money!, more, move, multiply

N： native, native?, native!, negate, negative?, new-line?, new-line, next, ninth, none?, none!, not-equal?, not, now, number?

O： object, object?, object!, odd?, offset?, offset-to-caret, op?, op!, open?, open, or, or~

P： pair?, pair!, paren?, paren!, parsing, parse, past?, path?, path!, percent?, percent!, pick, poke, port,port?, port!, positive?, power, prin, print, printf, probe, protect, protocol, pwd

Q： q, query, quit

R： random, read, rebcode?, rebcode!, recycle, reduce, refinement?, refiniment!, reflect, reform, rejoin,remainder, remold, remove-each, remove, rename, repeat, repend, replace, request-file, resolve, return,reverse, reword, rm, round, routine?, routine!

S： same?, save, scalar?, script, script?, second, select, series, series?, set-path?, set-path!, set-word?, set-word!, set, seventh, show, sing?, sine, sixth, size?, size-text, skip, sort, source, spec-of, split-path,split, square-root, stack, stats, strict-equal?, atrict-not-equal?, string?, string!, struct?, struct!, subtract,suffix?, swap, switch

T： tag?, tag!, tail?, tail, take, tangent, task, task?, task!, tenth, third, throw, time?, time!, title-of, to-action, to-binary, to-bitset, to-block, to-char, to-closure, to-datatype, to-date, to-decimal, to-email, to-end,to-error, to-event, to-file, to-frame, to-function, to-get-path, to-get-word, to-gob, to-handle, to-hex, to-image, to-integer, to-issue, to-library, to-lit-path, to-lit-word, to-local-file, to-logic, to-map, to-module, to-money, to-native, to-none, to-object, to-op, to-pair, to-paren, to-path, to-percent, to-port, to-rebcode, to-rebol-file, to-refinement, to-relative-file, to-routine, to-set-path, to-set-word, to-string, to-struct, to-tag, to-task, to-time, to-tuple, to-typeset, to-unset, to-url, to-utype, to-vector, to-word, to, trace, transcode, trim,true?, try, tuple?, tuple!, type?, types-of, typeset?, typeset!

U： unhandle-events, union, unique, unless, unprotect, until, unview, update, upgrade, uppercase,url?, url!, usage, use, unset, unset?, unset!, utf?, utype?, utype!

V： value?, values-of, vector?, vector!, view

W: wait, wake-up, what-dir, what, while, why?, word?, word!, word-of, write

X: xor, xor~

Z: zero?, -- , -, //, /, ==, =?, =, !=, !, <=, <>, <, >=, >, **, *, ++, +, ??, ?

r3 - specification

Зеркало проекта

Концепты : скрипты , выражения, серии, блоки, файлы, функции, объекты, модули, порты, протоколы, парсинг, математика, ошибки, графика , системного представления , Draw диалект, VID диалект, системных событий , графический пользовательский интерфейс (GUI диалект), Richtext , эффекты, DevBase Server, RebDev (Smart Client), DocBase

Типы данных (58) : ! действия , ! бинарный !, BitSet, блок! , символ! , закрытие! , тип данных! дата! ,десятичной! , электронной почте! , конец! , ошибка, событие! файл! , кадр! , функция! , получить-пути!, получить слово!, плевок!, ручка!, хэш! , изображение!, целое! , issuse! , библиотека!, списка! , лит-пути!, лит-слово!,логика! , карты!, модуль! , деньги! , родной!, нет! , объект!, оп!, пара! ,парень! , пути! , процентов!, порт!, rebcode!, refiniment! , рутина!, ТВ-пути! , набор слов!, строка! , структуры!, тег! , задача!, время!, кортеж! , набирается!, отменяется!, гиперссылка! , utype!, вектор!, слова!

Функции (460): about, abs, absolute, action?, add, ajoin, alias, all, also, alter, and, and~, any-block?, any-function?, any-path?, any-string?, any-word?, any, append, apply, arccosine, arcsine, arctangent, array, as-binary,as-pair, as-string, ascii?, ask, assert, at, attempt,back, binary?, bind?, bind, bitset?, block?, body-of, break,browse, bugs, call, caret-to-offset, case, catch, cause-error, cd, change-dir,change, char?, charset, chat,checksum, clean-path, clear, close, closure?, closure, collect, comment, complement, compose, compress,confirm,construct, context, continue, copy, cosine, create, cursor, datatype?, date?, debase, decimal?, decloak, decode-url,decode,decompress, dehex, delect, delete, deline, delta-profile, delta-time, demo, detab, difference, dir?, dirize,disarm, divide, do-codec, do-events, do, docs, does, dp, draw, dt, dump-obj, dump, echo, effect, eighth, either,else, email?, empty?, enbase, encloak, encode, enline,entab, equal?, error?, eval, even?, event?, evoke, exclude,exists?, exit, exp, extend, extract, fifth, file?, find-module, find-script, find, first+,first, for, forall,foreach, forever, form, format, forskip, found?, fourth, frame?, func, funco, funct, function?, function, functor,get-env, get-path?, get-word?, get, gob?, greater-or-equal?, greater?, halt, handle-events, handle?, has, head?,head, help, if, image?, import, in-dir, in,index?, info?, input, insert, install, integer?, intersect, issue?, join,last, latin1?, launch, layout, length?, lesser-or-equal?, lesser?, library?,license, list-dir, lit-path?, lit-word?,load-gui, load, log-10, log-2, log-e, logic?, loop, lowercase, ls, make-dir, make-scheme, make, map-event, map-gob-offset, map?, map, max, maximum-of, maximum, min, minimum-of, minimum, mkdir, mod, modified?, module?,module,modulo, mold, money?, more, move, multiply, native?, native, negate, negative?, new-line?, new-line, next, ninth,none?, not-equal?, not, now, number?, object?, object, odd?, offset?, offset-to-caret, op?, open?, open, or, or~,pair?, paren?, parse, past?, path?,percent?, pick, poke, port?, positive?, power, prin, print, printf, probe,protect, pwd, q, query, quit, random, read, rebcode?, recycle,reduce, refinement?, reflect, reform, rejoin,remainder, remold, remove-each, remove, rename, repeat, repend, replace, request-file, resolve,return, reverse,reword, rm, round, routine?, same?, save, scalar?, script?, second, select, series?, set-path?, set-word?, set,seventh, show,sing?, sine, sixth, size?, size-text, skip, sort, source, spec-of, split-path, split, square-root,stack, stats, strict-equal?, atrict-not-equal?,string?, struct?, subtract, suffix?, swap, switch, tag?, tail?, tail,take, tangent, task?, task, tenth, third, throw, time?, title-of, to-action, to-binary, to-bitset, to-block, to-char,to-closure, to-datatype, to-date, to-decimal, to-email, to-end, to-error, to-event, to-file, to-frame, to-function,to-get-path, to-get-word, to-gob, to-handle, to-hex, to-image, to-integer, to-issue, to-library, to-lit-path, to-lit-word, to-local-file, to-logic, to-map, to-module, to-money, to-native,

to-none, to-object, to-op, to-pair, to-paren,to-path, to-percent, to-port, to-rebcode,to-rebol-file, to-refinement, to-relative-file, to-routine, to-set-path,to-set-word, to-string, to-struct, to-tag, to-task, to-time, to-tuple, to-typeset, to-unset, to-url, to-utype, to-vector, to-word, to, trace, transcode, trim, true?, try, tuple?, type?, types-of, typeset?, unhandle-events, union,unique, unless, unprotect, unset?, unset, until, unview, update, upgrade, uppercase, url?, usage, use, utf?, utype?,value?,values-of, vector?, view, wait, wake-up, what-dir, what, while, why?, word?, word-of, write, xor, xor~, --,-, //, /, ==, =?, =, !=, !, <=, <>,<, >=, >, **, *, ++, +, ??, ?, zero?

Ссылки

http://ru.rebol.wikia.com/wiki/REBOLution_in_RUSSIA
http://ru.rebol.wikia.com/wiki/RU-Dialect

https://sites.google.com/site/rebolinrussia
https://sites.google.com/site/rurebolcom - официальный сайт (попытки перевода на русский язык http://rebol.com

http://ru.wikipedia.org/wiki/REBOL
http://ru-rebol.livejournal.com

http://rebol.informe.com/portal.html
http://www.similarsites.com/goto/www.vanillasite.at
http://ru.syllable.org
http://rebol2.blogspot.com
https://github.com/hostilefork/rebol/wiki
http://re-bol.com/rebol.html
http://www.altme.com

http://www.red-lang.org
http://tryrebol.esperconsultancy.nl

10 причин еще раз взглянуть на REBOL

- Разнообразие символьных типов данных
- Мета программирование сделано удобнее. "Диалекты" (DSL - языки предметной области) продуктивны.
- Тише едешь -- дальше будешь.
- Включен графический диалект (VID - Visual Interface Dialect) (Не больше ТК или что-то около того, пожалуйста ...)
- Делается больше с меньшим.

- Улучшенный парсинг (и нет никакого оправдания для использования XML)

- Для скорости, я просто люблю Rebcode

- Большинство языков не поддерживают Unicode, а мы делаем (имеется ввиду, мы будем)

- Не отказываться от SOA. SOA может быть легковесным и гибким (Rebservices).

- WORA (Windows, Linux, MacOS, ...)

- Мы будем поддерживать многофункциональные интернет приложения (Browser-Plugin)

Открытость. REBOL Technologies (RT) открыли REBOL еще больше, чем прежде. Хотя само ядро языка до сих пор закрыто, компания открыла все остальное вокруг него, так что вы можете помочь в разработках. Все подсистемы, высокоуровневые стандартные функции, документация. Все.

Мощное устройство и решение независимого графического пользовательского интерфейса. Новый визуальный интерфейс диалект третьей версии - очень сильный, масштабируемый и полностью заменяет оригинальный VID, являвшимся пионером графического пользовательского интерфейса кода в одну строку. В этот раз акцент на упрощение конструирования сложных пользовательских интерфейсов.

Скорость. REBOL быстрее вдвое за счет новой техники кодирования, переписано ядро снизу вверх с устранением узких мест в графике и портах. Для задач появилась возможность ассинхронного выполнения, что значительно повышает скорость выполнения операций по сети, без необходимости создавать собственный движок. Некоторые из общих высокоуровневых стандартных функций в старой версии REBOL построены как родные низкоуровневые для ускорения.

Размер. REBOL традиционно маленький, но скрипты REBOL 3 предполагаются еще меньшим размером. С новыми функциями для общих операций, особенно для серий, многие скрипты могут быть сокращены в размерах.

Усиление проекта. Более двух лет разработки, основанной на 8 годах опыта предыдущих версий REBOL и тысячи девелоперовских дискуссий, изучение всех недостатков и преимуществ REBOL 2 оказали глубокое влияние на проектирование каждой отдельной части REBOL 3.

Расширяемость. С новыми и открытыми подсистемами, это возможно для вас, разработчиков, у которых есть желание портировать REBOL 3 на различные устройства. Благодаря почти полной ОС и аппаратной независимости, можно адаптировать REBOL для использования, например, Open GL или DirectX для аппаратных ускорителей 3D графики. Вы можете это выполнит на вышей установленной аппаратуре или альтернативной операционной системе на выбор. Имеется масса покрытий. REBOL 3 может это все.

Больше, чем необходимо. Модули дают вом возможность конструировать большие программы (в смысле возможностей, а не размера кода). Вы знаете, что REBOL это способ упорядочивания большого количества маленьких компонент.

Поддержка юникода поможет интернационализации. Добавление новых форматов, таких как более точный money! тип данных, придают REBOL 3 индустриальную силу с математическими возможностями, поэтому мы уверены, REBOL 3 не может не оправдать ваших ожиданий в этом направлении.

Разработка. Новые инструменты для развития имеющихся возможностей: DocBase и DevBase. Предстоит разработатать: ссылочное руководство, книги, инструменты проектирование графического пользовательского интерфейса, интегрированную среду разработки (IDE) и SDK. (это рискованное заявление ...) И что каждый из вас решит, что может быть в REBOL 3.

Свободный. REBOL 3 теперь совершенно бесплатный (как с пивом) со всеми будущими возможностями.

Добро пожаловать! Добро пожаловать в REBOL 3. Добро пожаловать в новую концепцию программирования. Добро пожаловать в будущее на 20 лет. Если вы видели REBOL 2 и отвернулись, для вас может быть сюрприз!

Ребраузер

Этот документ описывает **РеБраузер**, название проекта кодирования для **R3 GUI** оболочки по умолчанию.

Определение

РеБраузер это замена **ViewTop** в **R2**, устанавливаемый по умолчанию графический пользовательский интерфейс, который вы видите, когда вызываете стандартную **R3** программу и он выглядит больше похожим на веб-браузер, чем на рабочий стол иконок. Одно из главных преимуществ такого подхода, что любой пользователь умеет работать с веб-браузером и никому ничего не надо объяснять. Тем не менее, существуют и другие преимущества в этом, а главное, что содержимое окна будет показываться текущим реблетом, который может принимать любую форму по требованию.
Следует отметить несколько вещей:

- Это не веб-браузер, отображающий только веб-контент и использующий веб-языки разметки. Такое может быть сделано традиционным браузером и было бы глупо пытаться с ними конкурировать.
- Вы можете представить реблеты как особый формат для контента, что-то типа между **Flash** и **HTML**.**Реблеты** абсолютно динамичны и их основной формат предназначен для декларации текстов диалектов (читать исходный код).
- Для удобства навигации старый, основанный на иконках, **ViewTop**, не подходит. Рабочий стол иконок может быть легко создан любым реблетом при необходимости. Например, если вы идете по **URL** в библиотеку **REBOL.ORG**, то это опция, показывающая ее как лист скриптов или как панель иконок. Как показывают большинство операционных систем, каждый режим отображения имеет свои преимущества (и недостатки).
- Как прежде с **ViewTop**, вы сможете выполнить **REBOL** скрипт непосредственно и оболочка браузера не появится.
- Содержимое реблета может быть адресовано **URL**, таким же способом, как веб-контент, используя похожие выходные буферы (веб-сервисы).

Надеюсь, вы поймете, почему это изменение настолько важно. Это поможет пользователям представлять **REBOL** приложения в рамках общей понятной структуры - адрес (**URL**) и содержание страницы.
Наконец, стоит отметить, что такой подход для нас не нов. Первый прототип **ViewTop** 2001 года работал таким способом, но другие изменения в **RT** повлияли на движение в другую сторону. И это уже много раз предлагалось и сотрудниками фирмы и сообществом.

Задачи

Важнейшая задача в рамках этого подхода безопасность реблета. Пользователи могут запускать множество реблетов в рамках одного и того же **REBOL** процесса. Чтобы сделать

такое безопасным, требуются модули, задачи, песочницы и брандмауэры для изоляции каждого реблета друг от друга.
Как вы понимаете, это не тривиально, если мы хотим, чтобы реблеты выполняли свои требуемые операции ввода-вывода, запись файлов и сетевой доступ.
Нам придется пойти на ряд компромиссов, чтобы избежать задержек с релизом R3. Однако продукт от этого будет только сильнее.

Общий план

Общее представление окна пользовательского графического интерфейса ребраузера подобно простому веб-браузеру.
Требование **REBOL** сохранить простоту проекта. Меньше - лучше.
Запуск GUI - это отделенные по горизонтали:

- Топ-бар для навигации, кнопок контролов и URL поля.
- Панель закладок
- Панель контента
- Информация о статусе полей (прогресс-бар)

Топ-бар может быть:
[back][forward] [reload][home][stop] [url][history][go] [opts]

Большая часть этого очевидна для каждого, однако, выбор именно этих кнопок надо пояснить.
В общем, Я (Карл) хочу обычный метод переходов мета-представлений браузера. Опции это панель, предоставляющая две главные вещи: информацию и ссылки.

- Информационные опции включают панели о браузере и текущем реблете, включая **REBOL** заголовок, информирующий о цели, авторе, дате, версии, размере, сигнатуре. ссылки на документацию и т.п.
- Опции ссылок включают секцию для обычных опций браузера, но также все специальные требования для текущего реблета.

Критерий выбора опции - предоставление требуемой информации методом одного клика за раз. (И может быть уверены, что увидите это будет добавлено в будущие версии iphone и т.п.)

Документация - требования и структура

Эта страница содержит краткие сведения о том, как организована документация.

Заметим, однако, что это лишь цель-максимум, и для её достижения потребуется отредактировать сотни страниц (740, если быть точным). Так что это займёт какое-то время.

Паттерны использования

Этот документ создаётся на основе уже устоявшихся пользовательских паттернов использования. Мы выделяем следующие основные паттерны:

новые пользователи — пользователи, которые только начали и хотят быстро познакомиться с основами, но со ссылками на более детальные знания. Также мы заметили, что многих новых пользователей не заинтересованы читать длинные главы и описания.
глубокие знатоки — пользователи, которые хотят (или нуждаются) понять более сложные специфические сильные вещи или концепции языка. Это то, что вы читаете, если не получаете необходимое из текстов для новых пользователей.
делающие код — пользователи, которым требуются специальные детали о функциях, типах данных, ошибках и системном объекте.

Нам известна также другая важныя модель: пользователи, часто желающие вносить свой вклад в документацию. Они могут:

- усиливать небольшие статьи, которые найдут при чтении разделов
- добавлять и редактировать параграфы, чтобы сделать их смысл яснее и нагляднее
- добавлять или поправлять примеры

Для нас важно, чтобы был механизм обратной связи (для случайных пользователей) и также прямое редактирование содержания для пользователей экспертов.

Итоговые вгляды

Суммарно поддерживая все выше описанные модели, мы также хотим:

- Разделение «пространство» (домены или контексты, предметы или содержание) информации. Учебник не руководство, а руководство не справка. Структурно мы делаем такие секции очевидно отдельными (но связанными внешними ссылками).
- Ускоренную навигацию основных секций. Это топ-кнопки на каждой странице.
- Меню-ориентированная sidebar навигацию каждой секции. (Еще не обеспечена, но это цель)
- Простую логичную навигация страниц. Хотя это менее важно, но мы хотели бы представить информацию похожей на последовательную информацию в книгах. Это основание для вперед и назад ссылок.
- Эффективное использование поиска для нахождения контента. По ряду причин, наши ранние руководства не использовали хорошо поиск. Мы находим, что если вы «гуглите» «REBOL protocols», то вам не легко найти основной, главный топик. Мы это имеем ввиду.
- Богатое использование гипер-ссылок для всего содержания. Пользователи, читающие учебник, могут находить концептуальные ссылки и ссылки, отсылающие к деталям.
- Использование разумных названий гипер-ссылок для всего содержания, включая путеводитель, концепции, функции, типы данных, ошибки и секции системного объекта. Такие ссылки типа: concepts/series.html, functions/insert.html
- Поддержка многоязычных (локализация) документов на основе per-page.

Отношение с DocBase

Этот документ освещает REBOL как язык и не заменяет основные базовые знания, поддерживаемыеDocBase.
обеспечивает более детализированными статьями и замечаниями. Там могут быть части этого руководства и ссылки туда могут быть здесь. Для примера, описание графического объекта типа данных gob! Приведенного тут ниже.

Наконец, DocBase включает многие статьи и предложения, описывающие детали реализации, нижнеуровневое проектирование, специальные интерфейсы и еще много чего.

Организация

Основные секции. Документ делится на две главные категории: руководство и справки. Секция руководство это путеводитель и концепции.
Путеводитель — замысел дать пользователям быстрое, но на верхнем уровне, введение в язык. Введение в основные идеи, но без деталей.
Концепции — замысел обеспечить деталями, глубже описать все REBOL концепции, также включая графику, звук и другие подобные модули.

Справочные секции.
Функции — описание каждой встроенной в язык функции. Объединение каждой функции встроенной спцификацией с описанием деталей.
Типы данных — описание каждого типа данных в языке.
Системный объект — описание каждого поля системного объекта, цель, разрешенные варианты и т.д.
Ошибки — объснение в деталях каждого сообщения об ошибках, как это возникает и что это значит.
Могут быть добавлены другие справочные разделы, например, плагин-модули и стандартные интернет-протоколы.

Навигация

Имеются основные способы навигации по документам, которые поддерживаются:

Top level – обеспечивает доступ к документом посредством заголовочной панели наверху каждой страницы. Это основные разделы документа.
TOC – (таблица контента) страницы, предаставляемые каждым вышеуровневым разделом.
Next and back – кнопки вашего последовательного движения между страницами в некотором разделе.
Sidebar menu – позволяет вам перескакивать в различные части некоторого раздела (Pending).
Hiper-links – из текста вы берете прямо любую страницу
Search – позволяет вам сканировать внутренний документ по специальным ключам.

Локализация (интернационализация)

Мы хотим поддерживать перевод этой документации и обеспечить легкий способ синхронизации этого содержания с мастером контента.

В общем, мы будем обеспечивать альтернативные URL на других языках, подобно тому, что вы видите в других системах документации. Для примера **r3/docs/fr/functions/absolute.html** будет URL переведенной на французский **absolute** функция.

Однако, прежде чем начать перевод, нам реально требуется закончить переработку, реструктуризацию (и закончить новые разделы) этого содержания для REBOL 3.0 изменений.

Поддержание качества

REBOL язык это мастерский проект, построенный на принципах компьютерной архитектуры. Это не стройка типа базара с беспорядочными маленькими пассажами, где торчат гводи, забитые каждым пользователем.

Мы не думаем, что справочное руководство языка может быть написано начинающими или людьми, которые однажды вдруг немного подумали о концепциях. Хотя мы разрешаем редактирование этого руководства, напрямую модификация этих страниц позволена только квалифицированным REBOL разработчикам и экмертам по документации. Это делается, прежде всего, для качества документации.

Тем не менее любой пользователь может подкорректировать топики и добавить новое содержание через механизм обратной связи, редактируя страницы, потребовав пользовательский аккаунт на DevBase (chat messaging) со специальным рангом для определения того, что вы действительно знаете, что делаете. Таким способом мы можем содержать руководство в рамках истинных принципов, делающим язык специфичным.

rebol3-componentarchitecture

Документ, описывающий компоненты высоко уровневой архитектуры REBOL 3

Цель

Этот документ описывает REBOL 3.0 компоненты архитектуры верхнего уровня проекта. Имеется ввиду, представить описание системы в целом и мы не открываем здесь глубоких технических деталей (эта тема для отдельной документации).

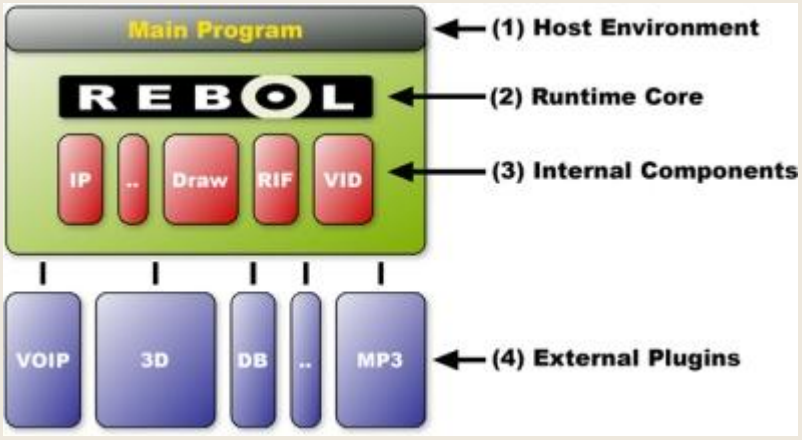
Структура

Главная цель REBOL 3.0 это построить легковесную, расширяемую архитектурно платформу, которая работает над широким кругом различных операционных систем и устройств.

Исходный код REBOL 2.0 был, в основном, закрытым (исключение составляли верхнеуровневые функции и схемы протоколов). Как результат, REBOL Technology несла ответствнность за портацию всех компонент во всех операционных системах (больше, чем 40 систем и свыше 150 выполненных продуктов). Среди последних, хорошо известные, конструирующие и поддерживающие MySQL, Oracle, ODBC, аудиовыход, криптографию и т.д. Это тяжелый груз на плечах компании — осуществлять актуальную и полную поддержку.

Чтобы достичь цели REBOL 3.0, мы вводим гибридную модель открытого исходного кода. Преимущество такого подхода заключается в том, что он позволяет сообществу развивать REBOL, поможет добавить и сохранить определенный характер языка и в будущем развивать его в качестве стандарта, простого и одинаково работающего в отношении всех систем и устройств.

Это четырех уровневая модель компонентов, как показано на диаграмме ниже:



Ниже объясняется каждая из этих компонент.

(1) Хост окружение — открыто

Хост окружение — это главная программа, которая инициализирует REBOL. Это может быть единственная установленная программа, такая как REBOL/View, плагин для web-браузера, web-сервер, интерактивная среда разработки (IDE) или даже что-то внедренное в пользовательские приложения.

Окружение (среда) определяется операционной системой абстрактным слоем (OSAL – abstaction layer operation system), который может быть перенесен на широкий спектр систем, включая настольные системы, коробочные установки, мобильные устройства и сотовые телефоны. Такой подход дает разработчикам возможность запускать и развивать REBOL на их специальных целевых платформах.

Например, REBOL может быть частью плагина web-браузера, который получается загрузкой с web-сайта, когда это необходимо. Мы сами сможем поддерживать Firefox или IE браузеры. Другие браузеры могут быть поддержаны сообществом разработчиков и поставляться сайтом REBOL Technology.

На сервере REBOL может быть встроен в веб-сервер Apache модулем или стартовать как привелегированная служба Windows или OSX прослойкой. Опять же ряд возможностей ограничивается только потребностями и талантом разработчиков.

REBOL Technology будет обеспечивать доступ к локализованному архиву исходных кодов для разработчиков, развивающих среды окружения.

(2)Ядро среды исполнения — закрыто

Это ОС зависимое ядро, которое обеспечивает стандартные возможности REBOL для всех систем и устройств. Это компонента, которая делает REBOL операции идентично на Windows, OSX, Linux, BSD, Sony, Nokia, Nintendo и на других системах.

Следует отметить, что эта компонента не является открытым исходным кодом, и является собственно, ядром. Однако, депозитная лицензия доступна для REBOL клиентов, которые нуждается в защите своих инвестиций в программное обеспечение с гарантированным доступом, в тех ситуациях, когда это станет необходимо.

(3)Внутренние компоненты — открыты

Внутренние компоненты обеспечивают встроенные возможности REBOL. Например, диалект визуального интерфейса GUI,загружающий image для GIF или JPG, библиотеку скалярно-векторной графики (AGG), вызывает функции, исполняющие внутренние программы, TCP/IP сетевой интерфейс и большинство общих протоколов, которые все являются общими частями REBOL. Не требуется импорта этих модулей. Они часть стандартной системы.

Это справедливо определенным образом для mezzanine функций системы, котрые при развитии будут неотъемлемой частью будущих релизов. REBOL Technology будет обеспечивать доступ к локализованному архиву исходных кодов для разработчиков внутренних компонент.

Мы будем со временем добавлять внутренние компоненты как часть стандарта REBOL системы для распределения. Однако, чтобы избежать потенциального раздутия компонент REBOL, будут добавляться только компоненты, удовлетворяющие конкретным требованиям.

(4)Внешние плагины — открыты/закрыты

Внешние плагины обеспечивают расширения REBOL. Эти компоненты требуются для специфичных типов приложений, нетсандартных частей REBOL. Для примера, интерфейс для MySQL, Oracle, ODBC и других систем баз данных лучше обеспечивать внешними плагинами. Другие возможности, включая загрузку и хранение широкого круга различных медиа (image, sounds, music, movies), голоса над IP (VOIP), 3D библиотеки, такие как OpenGL, пользовательскую аппаратуру, пользовательские инструменты и многое другое.

Внешние плагины могут быть:

Открытым или закрытым исходным кодом — мы надеямся, что большинство плагинов будет открытым исходным кодом, но разработчики имеют право создавать закрытый код для распространения.

Свободным или коммерческим — разработчики свободны при выборе лицензионной модели, которая лучше подходит для их цели. Мы бы хотели поощрить модель свободного распространения типа BSD, но мы также хотим, чтобы была возможность построения коммерческих разработок и продажи их собственных продуктов.

REBOL и/или C код — плагин может быть только REBOL кодом, C кодом (с REBOL нативными декларациями) или комбинацией обоих. Например, плагин может добавлять несколько нативных функций и несколько mezzanine функций или схему порта.

Подписанные и/или зашифрованные — плагины могут иметь цифровую подпись и зашифрованы для предоставления конечному пользователю, как удобнее разработчику.

rebol3-motivation

Viva la Rebolution!

Перевод заявления Карла Сассенрата о мотивах создания REBOL 3.0

Почему REBOL 3?

REBOL 3.0 представляет из себя полностью новый проект с нуля, построенный на основании новой информации, полученной за последнее десятилетие использования и исследования REBOL. Кроме того, настало время, чтобы сделать REBOL более открытым, позволить сообществу вносить больший вклад в развитие проекта. Это единственный путь, как поддержать **REBOL** в будущем на большом числе систем.

Хотите это тестировать?

Мы предлагаем тестирование широкому кругу пользователей и особенно призываем опытных пользователей **REBOL** проверить **R3 альфа**.

История разработки

REBOL уникален. С первых дней, более десяти лет назад, проект языка был до того уникален, что до сих пор там десятки глубоких вопросов остаются без ответа.

Например, насколько хорошо в реальности механизм блоков работает для выражений совместного интепретирования данных и кода? Можно ли построить целый язык вокруг одной фундаментальной концепции серии? Будет ли в REBOL приоритет слева направо для чтения выражений более или менее полезен, чем level's-of-precedence модель большинства языков? В какой степени, концепция диалектов стала основным принципом языка? Что должны INSERT и другие такие же действия возвращать? Как глубоко можно в сети осуществить открытое сообщение?

Были сотни таких вопросов и ответов. Тогда я мог только догадываться на основании образования, теории и двадцатилетнего опыта проектирования и внедрения языков. На многие вопросы, даже после многонедельных дискуссий, не было окончательных, идеальных решений.

А ответы затрагивают не только общую конструкцию языка, но и внутреннюю реализацию. Как с любым системным программным обеспечением, независимо от того, насколько хорошо она продумана, новые идеи и глубокие изменения подрывают согласованность первоначальной структуры. Со временем конструкция становится хрупкой и ее все труднее обновлять и поддерживать.

Теперь, после многих лет глубокого, ежедневного опыта, я думаю, что мы знаем ответы на многие из этих вопросов. Да, в большей части, я думаю, что многое было сделано правильно.

Но нескольким частям REBOL, я думаю, необходимы некоторые фундаментальные улучшения.

Например, я считаю, что некоторые из основных принципов проектирования портов имели недостатки. Функция второго порядка (такая как автонаследование поля порта или даже концепт порт как серия) занимали места больше функций первого порядка (типа легкого ввода вывода метода доступа), а также многообразие внешних устройств были сосредоточены в простом определении порта, а как результат, раздутый объект порт. Некоторые из этих проблем также относятся к графическому объекту фэйс. Стандартный объект фэйс стал слишком большим для реализации приложений, которые могут включать в себя тысячи графических объектов (например, игры или очень сложные пользовательские интерфейсы).

С другой стороны, REBOL просто вырос. Многие приложения выросли из оригинального проекта REBOL по принципу «программирование по-маленьку» (PITS). Есть несколько довольно фантастических REBOL приложений, которые действительно показывают силу REBOL, но я знаю, фактически, такая модель затрудняет им создание и поддержку, что на самом деле, можно улучшить. REBOL необходимо добавить усиление программирования самых больших моделей. Наконец, настало время REBOL вырасти и повзрослеть. Замечу, что речь не идет о том, чтобы REBOL стал больше и тяжеловеснее. Мы не желаем, чтобы качалось по 10 Мб или усложнялась установка. Это не суть REBOL. REBOL всегда был маленьким и легким и REBOL 3.0 останется, именно, таким. Пора сделать то, что REBOL, действительно, может. И это такое время, когда вы можете помочь.

Рыночная цель

У вас есть цель. Вы должны обеспечить простую специальную задачу что позволяет вам идеально подойти определенной группе людей.

Быть «всем» и возможность «делать все», хорошо на первый взгляд, быть «швейцарским армейским ножом». Да, это трудно, выделить то, что кажется важным и сказать: «Мы не хотим делать все» или «REBOL не для всех».

Однако, я создавал прежде продукт для каждого. Технология Amiga на годы опережала другие персональные компьютеры, и пострадала, как компьютер общего назначения. И это удивительный синдром. Она была так хороша во многом, что мы все посчитали, что она должна делать все для всех. Но это не то, что люди покупают. Люди покупают решения. Люди покупают приложения. Очень немногие из нас покупают технологии, чтобы владеть еще большей технологией. (Я имею ввиду «купить» как «признать и использовать», а не просто «ради оплаты»).

Те из нас, кто прошел опыт нецеленаправленного маркетнига Amiga, пострадали от «сделать все» и прекратили все отношения.
Спросите любого пользователя Amiga и увидите следы больных шрамов. И я знаю, что эти шрамы никогда не исчезнут.

В конце концов, Commodore, решившая что Amiga БЫЛА, это не весь мир. В США Amiga известна, в основном, как недорогая система видеомонтажа, типа видео тостера. За пределами США Amiga была удивительной игрой, хакер машина с десятками тысяч игр,

связанных общим названием. Но чтобы было сделано там «подход - сделать все», не хотелось бы снова повторять.

Пришло время определить цель для REBOL и я думаю, что многие из нас ее понимают. REBOL идеально подходит для легковесных распределенных приложений. Да, это наша революция, и скорее всего, именно поэтому, вы выбрали REBOL и время для чтения этого длинного общего текста.

Да, я знаю ... Это все выглядит, как маркетинговый ход.
Я не люблю маркетинговые слова, так что не будем здесь распинаться. Давайте сконцентрируем наше внимание на следующем:

Легковесность — REBOL самый крошечный и это приложение микроскопическое. Не только сам REBOL может быть загружен и установлен меньше, чем за минуту, но и программы на нем имеют размеры, как правило меньше, веб-страниц, но обладают при этом большими возможностями. В моей книге, Macromedia Flash это легковесная система, а дотнет — нет. Вы можете однажды создать и постоянно пересматривать свою программу с точки зрения совершенства в рамках своей предметной области. Легковесность позволяет больше творчества, позволяет больше экспериментировать и так далее.
Распределенность — это работает на клиенте и это работает на сервере. Оба независимы, но оба объединены языком, семантической основой REBOL. Распределенность не требует сложности и не означает тяжести или чего-то подобного, что стало считаться нормальным в последние годы. REBOL создан для решения этой проблемы. Какие еще технологии могут так облегчать клиент-сервер соединения, а также обеспечивать хранение данных?
Приложения — под этим я подразумевая делать что-то полезное. Решать проблемы. Я не имею ввиду крупные традиционные приложения «сделаем все». Я, фактически, имею ввиду, в основном, reblet (REBOL аплет). Маленькое приложение, делающее только то, что нужно. А большие приложения (типа AltME) - всего лишь набор мальньких приложений.

Да, я знаю, эта маркетинговая стратегия не нова. Я говорил это прежде. Я также видел недоумение на лицах, у которых нет цели совсем. Поэтому я повторяюсь. Помниться также удивление на лицах в 1983 году. Когда я предложил перенести многозадачность на персональный компьютеры. «Зачем? Кому это надо?» - спрашивали они.

Я думаю, что в это кризисное время, мы в той точке истории, типа края следующей волны. С совершенствованием методов веб-браузеров (например, AJAX), думаю, что каждый начинает, наконец, понимать , что дальше гонки нет. Следующий шаг за распределенными вычислительными платформами. И все знают, что здесь — золотое дно.

Пришло время переходить на огромные скорости. Пришло время объединить усилия сообщества мощью третей версии REBOL. Лазером жечь дыры слабых подходов. Пришло время расчистить дорогу REBOL. **Viva la Rebolution!**

Главные моменты

REBOL 3.0 — гигантский проект. Однако, это целенаправленный проект. Мы не будем пытаться в каждую функцию встраивать все возможные проблемы и желания разработчиков. Вместо этого, мы сделаем систему более надежной и мощной, позволяющей членам

сообщества сосредоточиться на создании специализированных модулей, необходимые для их нужд.

Здесь письмо-лист главных моментов развития проекта REBOL 3.0 :

Развитие легковесности — мы принципиально отказываемся раздувать REBOL 3. Конечно, в систему будут добавлены многие важные функции, но вместе с тем, некоторые вещи будут урезаны. Предел системы — 1 Мб, но с более широкими возможностями, чем когда либо прежде.

Масштабируемость — более мощные приложения должны быть проще построены. Для этого строятся модули (с контролем масштабов и пространства имен), мультизадачность, более эффективная чистка памяти и отладка «крючочков и инструментов»

Гибридная открытая модель и открытые интерфейсы — развитие REBOL должны происходить быстрее и получить более широкую поддержку. Для этого реорганизовано ядро таким образом, чтобы максимально открыть внутренние и внешние (подключаемые модули) интерфейсы. Это означает новую модель для внешних портов (для примера можно назвать MySQL, инструменты загрузки-хранения медиа, JPEG вставки, общие модули типа плагинов OpenGL).

REBOL в каждом браузере — уже стало простым фактом, что если вы не в браузере, то вы не в игре. То есть такую функцию следует иметь и она будет на уровне исходного кода, как открытый интерфейс, чтобы получить помощь сообщества и заставить систему работать во всех браузерах и на всех операционных системах.

Встраиваемые REBOL сервисы — все варианты REBOL смогут безопасно подключаться к другим узлам REBOL и службам, эффективно обмениваясь информацией. Создание сети таких услуг должно быть таким же легким, как в строительстве и торговле.

Оптимизированная графика — мы улучшам существующую графическую подсистему, заменяя лучшими элементами для более быстрого и эффективного проектирования. Кроме этого (мы надеемся на поддержку сообщества) будут слои стандартного графического интерфейса, более функционального и простого в использовании.

Создание поддержки локализации — пришло время улучшить поддержку родных языков и набора символов. Цель расширить юникодовсие аксессуары. Обязательным стандартом будет французский и итальянский и, возможно, один или два других языка.

Метод резидентного хранения - REBOL плоский текстовый файловый метод хранения служит нам удивительно хорошо. Например, сообщения в AltME сохраняются и доступны десятки тысяч REBOL форматированных записей одновременно. Но пришло время для более встроенного метода для такого рода данных.

Скорость работы будет зависеть и от помощи сообщества. Мы обращаемся за этой помощью. В отличии от предыдущих релизов, мы не планируем делать все сами. На веб-сайте мы будем рассказывать о задачах, которые нам требуются. Приоритетные задачи — это предыдущая функциональность системы.

rebol - concepts

Концепты: скрипты: внедрение

REBOL обеспечивает для вас возможность внедрить скрипт в другие тексты, такие, например, как, как**web** -страницы. Ваш скрипт найдет их и выполнится.
Текст прежде заголовка. Текст скрипта не требуется начинать с заголовка. Его можно внедрять в любом месте текста, позволяющем вставлять сообщения почты, **web** -страницы и

другие файлы. Заголовок маркирует начало скрипта и его тело следующий текст. Текст возникает прежде заголовка вызывается preface функцией и игнорируется в течении выполнения.
Как Unix shell command - другой общий случай.
Текст прежде и после. Когда скрипт следует за текстом, не имеющем отношения к скрипту, скрипт должен быть заключен в квадратные скобки.
Смотрите примеры описанных ситуаций здесь.

Концепты: скрипты: рекомендации по стилю

Структурируйте для ясности содержание. Квадратных скобок блоков для ясности содержание не достаточно. Априори они связаны с синтаксисом, который определяет блок, но не содержание. Кроме того существует пространство между блоками, где скобки отсутствуют. Там где есть возможность, открывайте скобки для строк, ассоциируемых с выражениями. закрывающая скобка должна выражать больше, чем просто уровень, но и глубину. Это в равной мере относится и к круглым и фигурным скобкам. Старайтесь выделять связные выражения и сгруппированные. Аккуратно используйте переносы.
Стандартная табуляция REBOL - четырех-пробельная. Существуют различные редакторы, где вы можете изменять предпочтительные настройки.
Де-табуляция прежде переноса - символ табуляции **ASCII 9** не идентичен четырем пробелам во многих обозревателях, поэтому используйте редактор или **REBOL** функции для де-табуляции перед опубликованием.
Предел длины строки в 80 символов легче прочитать в портативных редакторах и в той же электронной почте. Кроме того , длинные строки создают проблемы при загрузке.
Имена слов - демонстрируют ваш код. Относитесь к выбору имен очень критично. Имя должно быть кратким и ясным. Когда есть возможность используйте слова естественных языков. В простом прямой путь.
Используйте краткие слова, связанные со смыслом. Причем локальные слова могут быть короче простых слов, а длинные лучше использовать для глобальных слов.
Используйте целые слова, где только возможно. Не стремитесь сокращать слова. Слово - редкая ценность!
Имена сложных слов иностранного происхождения используются стандартным стилем, но не символами, а используя тире.
Начинайте имена функций с глаголов. Только потом местоимения, наречия и прилагательные. Некоторые существительные могут использоваться как глаголы.
Начинайте слова данных с существительных. Это слова для объектов и переменных, содержащих данные. Они могут включать модификаторы (прилагательные) при необходимости.
Используйте стандартные имена!
Заголовки скриптов используйте самым активным образом. Они позволяют другим скриптам использовать информацию, например, при каталогизации.
Заголовки функций, как правило обеспечиваюся специальными блоками для строк в пределах 70 символов и меньше. В них описываются типы значений функции и значения нормального возврата.
Имена файлов. Лучший способ - подумать о том, как вы найдете файл через несколько месяцев. Часто достаточно коротких и ясных имен. Не забывайте, что имена файлов участвуют в директориях. Связывайте их с общими словами.
Примеры внедрений. Показывайте в скрипте примеры, как его быстро верифицировать и корректно использовать в других системах.

Отладка внедрений. Часто используется для построения отладочных функций как порт скрипта. Особенно в сети и для файлов, содержащих скрипты, которые посылают и записывают файлы для тестов. Такие тесты могут быть возможны с контрольными переменными в голове скрипта.
Минимизация глобалов. В больших скриптах и, вообще, где только возможно избегайте использовать глобальные имена, которые доставляют внешние выражения другим моделям и функциям. Это всегда практично для коротких скриптов, но помните, что короткий скрипт может каждый раз вызывать скрипт, который длиннее. Когда используете коллекцию глобальных переменных, то закрывайте отношения с используемыми объектами, которые держат треки.

Для каждой рекомендации разработчики языка приводят примеры, которые можно посмотреть здесь.

Концепты: скрипты: оптимизация

Имеется ввиду процедура **cleanup**, форматирование скрипта с принятыми в сообществе рекомендациями. В библиотеке примеров имеется пример короткого скрипта, который для этого предназначен. Его можно посмотреть здесь. На тему оптимизации функций смотрите Rebcode.
В принципе, скрипт формируется свободно, вы можете предпочитать только командную строку, расщеплять его или соединять в одну длинную строку. Интерпретатору безразличны человеческие проблемы. Но из-за них, вопрос: следовать рекомендациям, более важная вещь, чем кажется на первый взгляд. Читабельность кода - это критерий его качества и основания для переиспользования. Такой код легче продуктировать, поддерживать и изменять.
В следующей секции подробнее о рекомендациях по форматированию. Смотрите примеры!

Концепты: скрипты: комментарии

Комментарии используются для пояснений назначений секций скрипта. **REBOL** предоставляет несколько типов комментариев:

- **in headers** - каждый скрипт начинается с заголовка, где имеется возможность описать общую информацию о скрипте
- **in functions** - это часть спецификации **function!** используемой **help**, рекомендуется всегда комментировать название функции и аргументы
- **embedded** - комментарии внедренные в код и могут быть там, когда код загружен
- **lexical** - лексические комментарии возникают в ресурсах кода и освобождаются от кода при загрузке

Лексические комментарии делаются точкой с запятой:

```
zertplex: 10 ; set to the highest quality
```

Внедренные комментарии в фигурных скобках:

```
{
This is long multined comment.
}
```

Такая техника работает, когда строки не интерпретируются как аргументы функций. Если вы хотите быть уверенными, чтобы сложный комментарий не интерпретировался как код, можно перед строкой указать слово **comment**:

```
comment {
This is a long multilined comment
}
```

Функция **comment** указывает игнорировать следующий блок или строку.
Внимание! Строка и блок актуальные части скрипта. Будьте аккуратны при использовании места в данных блока, потому что они становятся частью этих данных!

Концепты: скрипты: хранилище данных

В продолжение темы хранения данных мы видим, что файлы можно использовать как хранилище данных. Формат простой. Заголовок не требуется. Используются функции **load/all** и **save**.
Следует заметить, что **REBOL** может работать с файлами в сотни Гб. В дальнейшем будут продемонстрированы приемы сжатия, конвертации и криптографии. Планируется развитие данной темы.

Концепты: скрипты: сохранение

load функция загружает данные, которые могут быть записаны в ресурсном текст формате. Фантастическая возможность создания внутренней мини базы!
save функция имеет два аргумента: имя файла и либо блок либо сохраняемое значение.

```
data: [Buy 100 shares at $20.00 per share]
save %data.r data
```

Соотвественно позднее:

```
data: load %data.r
```

Так можно перегружать простые данные, например, **stamp**:

```
save %data.r now
stamp: load %data.r
```

Простые значения не требуют скобок. Можно сохранить файл скрипта с заголовком, тогда функция используется с параметром

```
header: [Title: "Example"]
save/header %data.r data header
```

Функция **save** имеет еще такие параметры: **/all**, **/flat** и **/only.**

Концепты: скрипты: загрузка

Файл скрипта может быть загружен как данные без выполнения. Это делается функцией **load,** которая

читает текст файла и транслирует переменные в значения, слова и блоки. Ее результатом будет блок, в котором просто значения и они возвращаются. Аргументы функции это имя файла, **URL** или строка.

```
load %script.r
load %datafile.txt
load http://www.rebol.org/script.r
load "print now"
```

Итак **load** функция:

- читает текст из файла, **URL** или строки
- ищет заголовок, если есть
- транслирует данные
- возвращает блок, содержащий значения

Например, если файл скрипта **buy.r** содержит текст:

```
Buy 100 shares at $20.00 per share
```

То он может быть загружен из командной строки:

```
data: load %buy.r
```

Результат загрузки в боке:

```
probe data
[Buy 100 shares at $20.00 per share]
```

Это пример диалекта **REBOL**, не прямо осуществляющего кода. Файлу не требуется заголовок при загрузке, который необходим при выполнении. Функция **load** имеет параметры:

- **/header** - включает заголовок, если он представлен
- **/next** - загружает только следующее значение (одно за раз)
- **/marup** - обрабатывает файл как текст разметки (**XML**, **HTML**), заключая теги в блоки

То есть можно возвратить и заголовок скрипта, который будет возвращен как первый аргумент и при использовании параметра **next** возвратит два значения.

Наконец, пример с использованием **load/markup:**

```
<title>Example</title>
probe data
[<title>Example</title>]
```

Концепты: скрипты: выполнение

Существует два способа выполнить скрипт: инициализировать его в работающем интерпретаторе или функцией **do**.

rebol script.r

Или:

do %script.r
do http://www.rebol.com/script.ru

do функция возвращает результат, когда завершит определение.
! Не забывайте о валидном **REBOL** заголовке в файле скрипта.

Концепты: скрипты: программные опции

Скрипты имеют доступ к опциям, предоставляемым интепретатором. Они находятся в**system/options** объекте. Объект содержит следующие поля:

- **home** - путь, определяемый как переменная окружения вашей операционной системы или (системного регистра для определенных типов ОС). Путь используется для нахождения файлов**rebol.r** и **user.r**
- **script** - имя файла, предоставляющего запущенному интепретатору скрипт
- **path** - путь к текущей директории
- **args -** аргументы, предоставляемые интепретатору в командной строке
- **do-arg** - строка как аргумент **--do** опции командной строки

Объект содержит и другие опции. Тип:

probe szstem/options

Примеры, содержащие опции объекта:

print system/options/script
probe system/options/args
print system/options/home/user.r

Концепты: скрипты: аргументы

Когда скрипт вычисляется (развертывается) он имеет доступ к информации о себе, которая находится в**system/script** объекте. Объект содержит список полей:

- **Header** - заголовок скрипта
- **Parent** - объект родительского скрипта, если скрипт развертывает другой скрипт
- **Path** - путь директорий файл или **URL** где осуществляется вычисление скрипта
- **Args** - аргументы скрипта, которые передаются командной строкой операционной системе или функции **do**, которая развертывает скрипт

Пример:

```
print system/script/title
print system/script/header/date
do system/script/args
do system/script/path/script.r
```

Последнем пример демонстрирует некоторый путь, где скрипт **script.r** выполняется в настоящее время.

Концепты: скрипты: заголовки

Каждый скрипт должен иметь заголовок, чтобы можно было определять назначение и другие атрибуты скрипта. Там содержится информация об имени скрипта, авторе, данных, версии, имени файла и тому подобное. В файлах данных, которые не связаны прямым "развертыванием" не требуют заголовка.
Итак:

- заголовки устанавливают скрипт, как существующий валидный ресурс для интерпретатора
- интерпретатор использует заголовок, чтобы определить название скрипта, ресурсы и версию перед развертыванием
- заголовок предоставляет стандартный путь для связывания деталей
- скрипт архивируется и сайты используют заголовок для генерации директорией скрипта и ссылок
- в некоторых редакторах доступны преобразования заголовка и с ним тоже можно работать

Сам скрипт может использовать информацию заголовка и его блочную структуру. Это расширяемый объект. Формально после слова **REBOL** содержание обязательных квадратных скобок может быть пустым, а насыщенный информацией заголовок может выглядеть примерно так:

```
REBOL [
 Title: "Full REBOL Header Example"
 Date: 8-Sep-1999
 Name: 'Full-Header ; For window title bar

 Version: 1.1.1
```

```
File: %headfull.r
Home: http://www.rebol.com/rebex/

Author: "Carl Sassenrath"
Owner: "REBOL Headquarters"
Rights: "Copyright (C) Carl Sassenrath 1999"

Needs: [2.0 ODBC]
Tabs: 4

Purpose: {
The purpose or general reason for the program
should go here.
}

Note: {
An important comment or notes about the program
can go here.
}

History: [
0.1.0 [5-Sep-1999 "Created this example" "Carl"]
0.1.1 [8-Sep-1999 {Moved the header up, changed
comment on extending the header, added
advanced user comment.} "Carl"]
]

Language: 'English
]
```

Концепты: скрипты: структура

Структура скрипта свободна. Разметка и форматирование может быть использованы для упрощения понимания структуры и содержания скрипта.
Поощряется использование стандартных стилей при конструировании скриптов, с большей универсальностью "читабельности".
Здесь еще будут примеры часто используемых конструкций -
header / body, including files, forward references, data creation, encoding Unicode, BOM.

Концепты: скрипты: файл-суффикс

Можно обойтись без расширения файла, но современные платформы так исторически устроены, что активно используют суффикс, например, для операций открытия и редактирования файлов. В этом случае, легче корректно интерпретировать их.

REBOL файлы имеют расширение **.r** и оно используется функциями инсталляции для ассоциации файлов с системой. Кроме того подразумевается, что файлы находятся в стандартном директории с именем **REBOL** .

Однако соглашение в отношении суффикса не принципиально и файлы могут иметь любое расширение. И, безусловно, что вам доступны файлы в любом другом месте, при определенных условиях.

Код "языка" **REBOL** не имеет официального **MIME** типа. Поскольку **MIME** тип используются веб-страницами и браузерами, то неофициально существует такой тип:

text/rebol

Вообще, это обосновано для медийных типов, работающих с бинарным кодом, но для интерпретации текстовых данных не существенно.

Тем не менее, в системе **REBOL** также возможен **GUI** код, презентации, базы данных и их комбинации, поэтому некоторый **MIME** класс имеет место, как в **XML** или **HTML**.

Концепты: скрипты

Скрипты **REBOL** можно называть текстами. Разработчики, в принципе, не различают код и данные. Скрипт может быть программой и данными.

Причем, под программой понимается "эволютный код" (**evoluted code**), что точнее толковать как "процесс развертывания" переменных в значения. Для кого-то привычнее будет понимать под этим вычисление, исполнение или даже определение.

Сам язык, составляющий основу технологии **REBOL**, контекстно-зависимый, как подобает быть "настоящему человеческому" языку. Это позволяет называть его метаязыком.

Все что над ядром - диалекты (**dialects**). Да и само ядро можно считать и языком и операционной системой внешнего типа (экзоядром). И все-таки, говорить о текстах, как о продуктах языка, точнее, чем просто об инструкциях. Тем более к ним относятся:

- файлы **REBOL**
- внедренные тексты в другие типы файлов (интернет страницы)
- источники ресурсов в сообщениях между системами
- фрагменты текстов типа документов

wipwiki

Что такое WIP Wiki?

"**WIP** устанавливается для работы в развитии, это вики-система, которую мы используем с 2004 года для управления и редактировании наших веб-сайтов.

WIP написан на **REBOL**. Он очень маленький, только 50 КБ исходного кода, с несколькими внешними HTML-страницами как образцами. Он также очень быстрый и эффективный, потому что только перестраивает веб-страницы, когда это необходимо, а не каждый раз, когда они просматриваются.

В феврале 2009 года WIP был существенно перестроен, чтобы была возможность использовать его как нашу главную систему управления документами, для нашего ссылочного руководства.

http://rebol.com/r3/docs/ - пример. Документы включает 740 страниц вики контента.

Использование WIP Wiki

Вам не требуется много знать, чтобы использовать **WIP wiki**. Эти страницы видны только как нормальные веб-страницы и только с протыми ссылками редактировать содержание страниц с помощью вашего веб-браузера.

Страница контенат написана в тексте, использующим MakeDoc , простой язык разметки (markup, markdown) , написанный в 2000 году для создания **REBOL** документации (больше, чем 1000 страниц).

Заметим, что **WIP** также работает один (установленный без веб-сервера), для генерации документов ставится просто прямо в директорий на диске.

Сожалеем, но ...

Мы не распространяем **WIP wiki**, потому что существуют и так много всевозможных вики движков в сети, которые свободно распространяются. Мы не хотим тратить время на приведение **WIP** в состояние для стороннего использования, написания документации, проверки секретности и т.д. Сейчас это просто маленький прекрасный **REBOL** инструмент, который нам нравиться использовать ... хороший молоток."

REBOL.ORG REBOL.COM REBOL.NET REBOL.NL EN:WIKI:REBOL RIX RU:WIKI:REBOL ALTME Twitter Для начинающих Ссылки Cheyenne-server

Что такое REBOL

Это о *языке*... технология, дающая возможность быть всем другим технологиям.

Мы верим, что язык, сам по себе, когда в паре с человеческим интеллектом, гораздо продуктивнее в обозначениях, чем традиционные программные технологии.

Пример REBOL приложений предприятия только 2K из 42K. Reblets выполняются с обоих сторон, сервера и клиента.

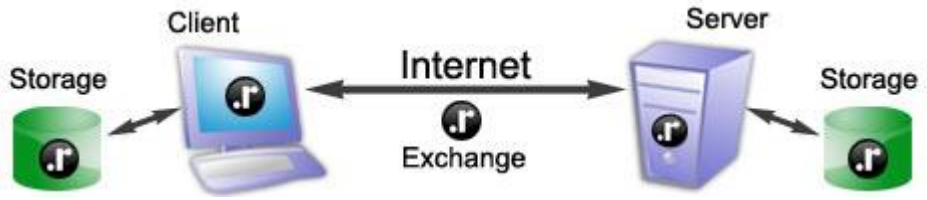
Краткие факты

- **Мы боремся против идеи, что современные программы должны быть большими и сложными.**
- Фактически, мы уверены, что такие системы - примеры большинства потенциальных ошибок, с губительными социально-экономическими последствиями.
- **REBOL построен как *относительные выражения базовых объектов языка*.**
- Он основан на символьной обработке и денотационной семантике и его мощь в относительных выражениях, диалектах.
- **REBOL Technologies учреждена в 1998.**
- Технология спроектирована Карлом Сассенритом, архитектором мультизадачных операционных систем.
- **REBOL 3.0 сейчас в разработке.**
- Полностью переписан на основании, что на освоение REBOL 2 требуется больше года.
- **REBOL произносится "reb-ol".**
- Как в rebel yell или "rebel with a cause".

Концепции

Технически говоря, REBOL выгоден и полезен как легковесные **предметно-специфические подязыки и микроформаты**. REBOL вводит концепт диалектов: маленькие, эффективные, предметные языки для кодирования, данных и метаданных.

- **REBOL это "двух-сторонняя" технология.**
- Это интегрированное решение распределенных вычислений, чтобы работать лучше, когда что-то исполняется и на стороне сервера и на стороне клиента. (Однако, что не требуется, вы можете использовать его только на клиенте или только на сервере.)



- **REBOL обеспечивает легковесную платформу для Интернет приложений.**
- Сила REBOL в том, что его приложения малы и не растут при распределении, всегда содержательны и изменяемы. Они размером с типичную веб-страницу, куда вы можете всегда встроить REBOL приложение.
- **REBOL больше. чем просто язык программирования.**
- это также язык для представления данных и метаданных. Он обеспечивает содержательную архитектуру для вычислений, хранения и изменения информации.
- **REBOL принципиально маленький и легковесный.**
- Это требует несколько секунд загрузки и он не требует для выполнения специальных библиотек, но он включает сотни функций, встроенные типы данных, помощь, множество интернет протоколов, графику, звуки, и много чего ещё.
- **В чём REBOL другой**
- REBOL не традиционный язык программирования типа C, BASIC или Java.
- **Зачем использовать REBOL?**
- Более детально об этом, почему это полезно.

Ещё

- **Простая галерея** and **галерея аплетов предприятия**
- Скриншоты различных реблетов (*reblets)*.

- **REBOL это кросс-платформа.**
- Программы, написанные на Windows исполняются также хорошо на Linux, BSD, OS X и многих других платформах... при этом не требуется никаких изменений.
- **REBOL существует в различных пакетах.**
- Начинающие могут использовать свободную лицензию REBOL/Core и REBOL/View пакетов. Профессионалы найдут в REBOL/Command и REBOL/SDK мощные инструменты.
- **Как думают пользователи?**
- Здесь некоторые отзывы нашего признания.
- **Больше пяти миллионов загрузок**
- Включая REBOL/Core и REBOL/View.
- **Большие и маленькие организации используют REBOL**
- Чтобы создать интернет приложение и решения за нормальное время и цену.
- **REBOL держава AltME**
- Безопасная, легковесная платформа для приложений и взаимодействия, созданная SafeWorlds, Inc. Мы поддержим вас, чтобы вы попытались.

С чего начать

Учебники и примеры

Изучение REBOL требует от вас по-другому думать о программировании. REBOL - единственный, в своём роде. Мы уверены, что начинающим часто "даётся это" лучше, чем опытным программистам.

Чтобы стартовать...

- **Что такое REBOL?** Краткие факты.
- **REBOL быстрый старт** Карл Сассенрат, создатель REBOL, помогает вам стартовать.
- **Галерея** Скриншоты некоторых реблетов (*reblets)* с ссылками на исходный код.
- **Как смотреть исходный код** How to view the code for scripts and functions.
- **REBOL краткая карта ссылок** (Новое, в разработке)

Учебники

- **Учим REBOL** Содержательный учебник от Nick Antonaccio.
- **REBOL 3.0 руководство** Часть нового REBOL 3 документы вики.
- **Новый REBOL FAQ** Если у вас затруднения, попробуйте отсюда.
- **Обучение программированию на REBOL** курс программирования от Victor Pavlu.
- **Программирование с REBOL, Code Conscious** Большой сайт для новых REBOLеров, с Brett Handley.

- **REBOL программирование викибук** Открытые документы, созданные сообществом.

Специальные учебники

- **Руководство начинающему REBOL графический интерфейс** Старое. но легко читаемое руководство для новых пользователей.
- **REBOL для простого CGI** Чтение форм, разработка веб-старниц.
- **Создание и обратотка веб-форм с CGI** Как создать веб-форму и обработать её.
- **Использование REBOL CGI в Apache на Microsoft Windows** Точные детали, как получить это работающим.
- **REBOL CGI BBS учебник** Как создать BBS ("доску объявлений") в REBOL.
- **GUI как это?** Несколько старых учебников, но до сих пор полезных.

Примеры

- **The REBOL.org библиотека** Сотни примеров и полезные программы.
- **Только одна линия** Краткий тур, что можно в одну линию в REBOL.
- **Cookbook of REBOL примеры кода** Масса примеров для новых программистов.
- **Pongo Game** Простые break-out стили на примере игры.

Другая информация

- **REBOL индекс документации** Полный лист REBOL документации.
- **REBOL сеть разработчиков** Для REBOL разработчиков.
- **Carl's REBOL блог** Фантазии и рассуждения обо всех вещах REBOL.
- **Книги о REBOL** Некоторые опубликованные книги.
- **Краткое введение в REBOL скрипты** Введение с некоторым множеством примеров.
- **REBOL за 10 шагов** Лёгкое и глубокое введение в REBOL.
- **REBOL with a Cause** Принципы, лежащие в языке.
- **REBOL скриптовые языки** Статьи. опубликованные в Dr. Dobbs, Software Tools для профессиональных программистов.
- **Редакция кода REBOL** Экспертиза редакций того, что разработчики находят полезным.

Преодоление сложности программ ...

Программные системы стали слишком сложными и чем выше их уровень сложности, тем они более неустойчивые и ненадёжные. В конце концов***программы становятся проблемой, а не решением*** . Мы боремся против этой сложности, в наибольшей степени, доступным сильными средствами и возможностями: ***самого языка***.

R3 краткая справочная карта ***Новое!***
Одна страница со ссылками на общие возможности и функции. Это полезный ресурс для новых программистов, которые хотят увидеть основные функции и типы одним взглядом.

REBOL 3.0 альфа тест 16-Sep-2010
Улучшения, расширения, оптимизация. REBOL сейчас быстрее и даже более мощный. Новые типы данных, новые базовые функции, новые идеи и Юникод. Небольшая, расширяемая платформа, связывающая все операционные системы и устройства. Один язык, сервер и клиент, который не "надувается".

REBOL/SDK 2.7.7 выпуск 1-Март-2010
Новый SDK и Command пакеты, доступные в настоящее время. Устанавливаемые платформы MS Windows, OS X и Linux.

REBOL 2.7.7 выпуск 10-Январь-2010
Включены новые возможности типа SSL, ODBC, и больше. Также добавлены R2/Forward "mezzanine" функции (R3-подобные возможности). Инсталляция и характерный "view-root" вход. Ну и, устранены несколько ошибок

Ревизия основного веб-сайта 28-Январь-2010
Улучшена структура, перестроены страницы, добавлены новые возможности. Этот веб-сайт сильно изменён, реорганизована базовая структура, все основные страницы интегрированы в простой вики-движок для согласованного отображения и обепечения большего уровня внешних ссылок, чем прежде.

Сообщества

REBOL имеет активные сообщества разработчиков по всему миру с различными уровнями подготовки. Здесь сгруппирована информация о них и вы можете присоединиться.

Коллекции скриптов

- **REBOL.org** Сотни работающих программ и примеров. Код, чтобы пользоваться и учиться. Включает REBOL email лист подписки на архивы (1998-2010) и AltME дискуссии.
- **SoftInnov.org** Профессионалам, библиотеки хорошо написанных скриптов, включая Cheyenne веб-сервер, MySQL Driver, Windows Services библиотека поддержки, Visual Captcha, Scheduler, Uniserve мультипротокольный сервер, и более.
- **RG: REBOL** Полная коллекция хорошо написанных инструментов и программ, включая Google чаты, Twitter клиент, Сборка REBOL приложений в OS X, QuarterMaster веб-приложения, и ёщё много чего.
- **REBOL.Wik.is** Исходный код проекта архива сайта, включая интерфейсы для OpenDBX, HylaFAX, JDBC bridge, SOAP, и другое.
- **Dobeash Development Tools** Домашняя страница *RebDB*, маленькая, но высокоэффективная база данных; *SQLite Driver*, обеспечение базового REBOL доступа к SQLite; *RebGUI*, легковесная альтернатива VID.
- **Power Mezz** Коллекция полезных скриптов и открытых модулей с REBOL сообществом. Не требуется изобретать велосипедов. С помощью *Gabriele Santilli.*

Дискуссионные форумы

Многие REBOL-еры используют REBOL-базовые форумы для дискуссий и частного взаимодействия:

- **R3 чат**
- Это главный форум для REBOL 3.0. Он работает во всех REBOL консолях в текстовой моде, но GUI версия в планах.
  - Run R3, type `chat` и следуйте инструкциям (для всех платформ.)
  - Type "help" для детальной информации или посетите R3 DevBase Chat Forum.
  - Чтобы увидеть публичные сообщения для всех браузеров зайдите на RebDev mobile/phone интерфейс.
  - Проблемы? Пожалуйста напишите нам.
- **REBOL FaceBook** Новая специальная группа по интересам для Facebook пользователей.
- **Forum: REBOL** Веб-базовый форум для R2 и R3, поддерживающийся с помощью Synapse.

- **AltME Worlds** Система частных мгновенных сообщений, где реболеры пакостят круглосуточно 24/7. Имеется несколько *слов* демонстрирующих REBOL, но более популярен REBOL3 (но это о большем. чем R3.)
    - Инструкции как присоединиться, а также: AltME руководство быстрого старта.
    - REBOL интернациональная дискуссия на REBOL.org. Вход в архив, первый - последний.
    - REBOL международная дискуссия, web mirror. Только последние 300 сообщений дискуссионного потока.
- **RebelBB France** Простой форум, написанный на REBOL и о нём, на французском.
- **Email List Server** Общий REBOL лист электронных почтовых адресов, хотя мы видим для этого новое место. Также, хорошая веб-каталог почтовых сообщений есть опубликованный на REBOL.org.
- **Nick's REBOL Forum** Микро-форум (только несколько линий REBOL) хостится на Nick Antonaccio. (Замечание: "the captcha question is first".)

Документация на других языках

- **Guide Utilisateur de REBOL/Core** **(French)** Полный французский перевод Core описания.
- **Guida A Rebol** **(Italian)** Краткий обзор на итальянском

Другие сайты

- **Carl's REBOL Blog** Архив замечаний, идей и вопросов REBOL's фантазии и основания, Карл Сассенрат.
- **REBOL на Twitter** Записи о статусных изменениях и бюллетени.
- **RebolTutorial** Включая статьи, примеры, мнения и учебники.
- **The REBOL Week** В целом, о событиях в REBOL сообществе.
- **Try REBOL** Интерактивный вебсайт для опытов с REBOL 3.
- **REBOL Hosting Service Providers** Если вы присматриваете хорошего качества и с низкими ценами сайт для REBOL скриптов.
- **RIX Search Engine** Специальный движок только для REBOL веб-сайтов, скриптов, дискуссий и документации.
- **REBOL 3.0 Blog** Информация относительно проекта REBOL 3.0 системы.
- **REBOL.net Developer Network** Включает внешнюю информацию, документацию, скрипты и DocBase Wiki.
- **REBOL DocBase Community Portal** Вики страницы с разработками и пользовательскими ссылками.
- **Reblets How-To** Простые примеры

Документация

Вводные материалы

- **Что такое REBOL?** Основное введение.
- **REBOL учебники и примеры** Несколько путей, чтобы стартовать.
- **Reblet галерея** + **Demo галерея** Скриншоты некоторых реблетов (*reblets)*.
- **REBOL язык в вопросах и ответах** Вопросы и ответы.
- **Книги о REBOL** Выбор книг по теме.

Руководства, Справочники, Указатели

- **REBOL 2 документация** Справочная информация, графика и более.
- **REBOL Краткая карта ссылок**
- **REBOL/SDK: среда разработки Kit** Профессиональная SDK и Command.
- **REBOL 3 Страницы проекта** Домашняя страница для новых разработок.
- **REBOL 3 документация** ***(в разработке)*** Включая описания, концепции, функции, ошибки.

Пользовательские комментарии

- **Страницы REBOL сообществ** Различные ссылки.
- **REBOL Wiki** **(DocBase)** Пользовательская совместная информация.
- **Редакции кода для REBOL** Полезно для разработчиков.
- **REBOL в одну строку** Примеры использования только 1 линии кода.
- **Что говорят пользователи** Поддерживающие отзывы.

Другие ссылки

- **REBOL Logos** Помоги рекламировать REBOL.
- **Как установить SSH Auto Login** Удобно если используется REBOL на сервере.
- **REBOL изменения** Мастер индексирования изменений документов.
- **Последние изменения веб-страниц** (Указатель страниц ещё не включён)
- **Sitemap** (Указатель всех страниц ещё не включён)

Продукты

С 1998 года REBOL Technologies конструируют и поддерживают REBOL® язык и связанные с ним продукты. Мы гордимся, что наши продукты маленькие и управляемые, с учётом мощности и безопасности.

REBOL/SDK - среда разработки Kit

- Всё, что вам необходимо для создания REBOL приложений
- Включены все REBOL/Command возможности (как описано ниже)
- 3 REBOL инкапсулятоа для создания исполняемых программ
- 3 новых REBOL ядра: /Base, /Pro, и /Face
- REBOL процессор 2.0для организации ваших программ
- Доступ функций к регистру Windows
- Исходные модули для различных компонент
- Свободный доступ к изменениям в течении года
- Приобретение REBOL/Command/SDK: US$249
- REBOL/Command/SDK апгрейд: US$50

REBOL/Command для профессиональных разработчиков

- MySQL и Oracle интерфейс
- ODBC (SQLServer, Access, Excel) интерфейс
- Доступ к внешним библиотекам (DLLs)
- SSL безопасные сокеты (HTTPS клиент)
- Встраивание RSA, AES, Blowfish, DSA, DH шифрование
- Shell commands, запуск внешних приложений
- Доступ функций к регистру Windows
- Быстрый CGI сервер интерфейс
- Свободный доступ к изменениям в течении года

- Приобретение REBOL/Command: US$149

REBOL услуги по разработке

- Разработка профессионального программного обеспечения
- Консультации, связанные с проектом REBOL
- Коммерческая поддержка
- Специальные пакеты и сборки
- Обслуживание пользовательских лицензий
- *Мы REBOL эксперты!*

Загрузки

Меньше, чем 1MB, но такой мощный. Эти свободные загрузки, в основном, для студентов, учащихся, любителей и и маленьких организаций.
Проверьте REBOL продукты для бизнеса и предприятий разработчиков.
Поддержите разработку REBOL!

REBOL/View 2.7 для графических приложений

Версия 2.7.7 выпуск 1-Январь-2010

- Включает все REBOL/Core возможности (как перечисленные ниже)
- Сотни возможностей в меньше, чем 1 MB
- Высокая скорость графических составляющих движка
- Пользовательский интерфейс диалект, операции с картинками, эффекты
- Эта версия включает SSL/TLS, DLL доступ, ODBC и шифрование
- Программы выполняются одинаково на Windows, OS X, Linux и BSD
- Свободная для конечного пользователя лицензия

- Ещё о REBOL/View

REBOL/Core для серверных приложений

Версия 2.7.7 выпуск 1-Январь-2010

- Продвинутый язык для программирования диалектов
- Легковесный для быстрой разработки приложений
- Не требует инсталляции
- Более 40 встроенных типов данных
- Компрессия, хэширование (SHA1, MD5), проверка контрольной суммы
- 10 интернет протоколов, включая HTTP, FTP, SMTP, POP
- 100's встроенных функций
- Программы выполняются одинаково на Windows, OS X, Linux и BSD
- Свободная для конечного пользователя лицензия
- Ещё о REBOL/Core

REBOL 3.0 альфа тест версия

Следующие поколение разработок

- Полностью переписанный REBOL
- Сотни новых функций
- Смотрите REBOL 3 Развитие детали