

LibRed API

目次

1. 序	3
2. libRedのインストール	3
3. 環境	3
4. C API	4
4.1. 関数	4
4.1.1. redOpen()	4
4.1.2. redClose()	4
4.2. Redオブジェクト	4
4.2.1. redDo()	5
4.2.2. redDoFile()	5
4.2.3. redDoBlock()	5
4.2.4. redCall()	5
4.3. 関数ポインタ	6
4.3.1. redRoutine()	6
4.4. CからRedオブジェクト	6
4.4.1. redSymbol()	6
4.4.2. redUnset()	7
4.4.3. redNone()	7
4.4.4. redLogic()	7
4.4.5. redDatatype()	7
4.4.6. redInteger()	7
4.4.7. redFloat()	7
4.4.8. redPair()	8
4.4.9. redTuple()	8
4.4.10. redTuple4()	8
4.4.11. redBinary()	8
4.4.12. redImage()	8
4.4.13. redString()	8
4.4.14. redWord()	9
4.4.15. redBlock()	9
4.4.16. redPath()	9
4.4.17. redLoadPath()	9
4.4.18. redMakeSeries()	10
4.5. CからRedオブジェクト	10
4.5.1. redCInt32()	10
4.5.2. redCDouble()	10

4.5.3. redCString()	10
4.5.4. redTypeOf()	10
4.6. RedAction()	11
4.6.1. redAppend()	11
4.6.2. redChange()	11
4.6.3. redClear()	11
4.6.4. redCopy()	11
4.6.5. redFind()	11
4.6.6. redIndex()	11
4.6.7. redLength()	12
4.6.8. redMake()	12
4.6.9. redMold()	12
4.6.10. redPick()	12
4.6.11. redPoke()	12
4.6.12. redPut()	12
4.6.13. redRemove()	12
4.6.14. redSelect()	13
4.6.15. redSkip()	13
4.6.16. redTo()	13
4.7. RedWord()	13
4.7.1. redSet()	13
4.7.2. redGet()	13
4.8. RedPath()	13
4.8.1. redSetPath()	14
4.8.2. redGetPath()	14
4.9. RedField()	14
4.9.1. redSetField()	14
4.9.2. redGetField()	14
4.10. RedLog()	14
4.10.1. redPrint()	14
4.10.2. redProbe()	15
4.10.3. redHasError()	15
4.10.4. redFormError()	15
4.10.5. redOpenLogWindow()	15
4.10.6. redCloseLogWindow()	15
4.10.7. redOpenLogFile()	15
4.10.8. redCloseLogFile()	16
4.11. RedForm()	16
5. Visual Basic API	16
5.1. RedForm()	16
5.2. redLogic()	17

[illegible]

```
long a, blk;

a = redSymbol("a");
redSet(a, redBlock(0));           // 00000000000000000000000000000000

blk = redGet(a);
redPrint(blk);                     // 00000000

for(i = 0; i < 100, i++) {
    // redAppend(blk, redNone());   // 0000000000000000
    redAppend(redGet("a"), redNone()); // 00000000
}
```

4. C API

C API C/C++ C FFI Red

4.1. □□□□□□□□

libRed 0.0.0.0 API 0.0.0.0

NOTE `libRed`

4.1.1. redOpen()

```
void redOpen(void)
```

Red API

NOTE redOpen-2

4.1.2. redClose()

```
void redClose(void)
```

Red

4.2. Red

Red Red Red

4.2.1. redDo()

```
red_value redDo(const char* source)
```

Red Red Red

```
redDo("a: 123");

redDo("view [text {hello}]");

char *s = (char *) malloc(100);
const char *caption = "Hello";
redDo(sprintf(s, "view [text \"%s\"]", caption));
```

4.2.2. redDoFile()

```
red_value redDoFile(const char* filename)
```

filename Red *filename*
Red OS Unix

```
redDoFile("hello.red");
redDoFile("/c/dev/red/demo.red");
```

4.2.3. redDoBlock()

```
red_value redDoBlock(red_block code)
```

```
redDoBlock(redBlock(redWord("print"), redInteger(42)));
```

4.2.4. redCall()

```
red_value redCall(red_word name, ..., red_integer 0)
```

name word Red any-

function! **Red** **null** 0

```
redCall(redWord("random"), redInteger(6));    // 16 integer!
```

4.3.

Red **Red** **print** **ask** **Red** **redRoutine()**

4.3.1. **redRoutine()**

```
red_value redRoutine(red_word name, const char* spec, void* func_ptr)
```

name *spec* *func_ptr* **C** **Red** **routine** **C** **Red** **redUnset()**

```
#include "red.h"
#include <stdio.h>

red_integer add(red_integer a, red_integer b) {
    return redInteger(redCInt32(a) + redCInt32(b));
}

int main(void) {
    redRoutine(redWord("c-add"), "[a [integer!] b [integer!]]", (void*) &add);
    printf(redCInt32(redDo("c-add 2 3")));
    return 0;
}
```

4.4. **C** **Red**

libRed **API** *references* **Red**

4.4.1. **redSymbol()**

```
long redSymbol(const char* word)
```

C **string** *word* **ID** **ID** **word** **ID** **libRed** **API**

□

```
long a = redSymbol("a");
redSet(a, redInteger(42));
printf("%l\n", redGet(a));
```

4.4.2. redUnset()

```
red_unset redUnset(void)
```

□ *unset!* □□□□□□

4.4.3. redNone()

```
red_none redNone(void)
```

□ *none!* □□□□□□

4.4.4. redLogic()

```
red_logic redLogic(long logic)
```

□ *logic!* □□□□□□□□ *logic* □□□□0□□□□□□□□ *false* □□□□□□□□□□□□ *true* □□□□□□□□□□□□

4.4.5. redDatatype()

```
red_datatype redDatatype(long type)
```

type □□□□□□ID□□□□□□□□ *datatype!* □□□□□□□□□□ID□□RedType□□□□□□□□□□□□□□

4.4.6. redInteger()

```
red_integer redInteger(long number)
```

number □□□□□□□□□□□□ *integer!* □□□□□□□□□□

4.4.7. redFloat()

```
red_float redFloat(double number)
```

number □□□□□□□□□□□□ *float!* □□□□□□□□□□

4.4.8. redPair()

```
red_pair redPair(long x, long y)
```

integer pair!

4.4.9. redTuple()

```
red_tuple redTuple(long r, long g, long b)
```

integer RGB tuple! 8

4.4.10. redTuple4()

```
red_tuple redTuple4(long r, long g, long b, long a)
```

integer RGB tuple! 8

4.4.11. redBinary()

```
red_binary redBinary(const char* buffer, long bytes)
```

bytes **binary!**

4.4.12. redImage()

```
red_image redImage(long width, long height, const void* buffer, long format)
```

image! **width** **height**

- **RED_IMAGE_FORMAT_RGB**: 24BPP 24-bit per pixel
- **RED_IMAGE_FORMAT_ARGB**: 32BPP 32-bit per pixel

4.4.13. redString()

```
red_string redString(const char* string)
```

string string! UTF-8 redSetEncoding()

4.4.14. redWord()

```
red_word redWord(const char* word)
```

Cstring word! UTF-8 redSetEncoding(word) error!

4.4.15. redBlock()

```
red_block redBlock(red_value v,...)
```

block!series null 0

```
redBlock(0); // block
redBlock(redInteger(42), redWord("hi"), 0); // [42 hi] block
```

4.4.16. redPath()

```
red_path redPath(red_value v, ...)
```

path!series null 0

```
redDo("a: [b 123]");
long res = redDo(redPath(redWord("a"), redWord("b"), 0);
printf("%l\n", redCInt32(res)); // 123
```

4.4.17. redLoadPath()

```
red_path redLoadPath(const char* path)
```

C path!series

```
redDo(redLoadPath("a/b")); // a/b path!
```

4.4.18. redMakeSeries()

```
red_value redMakeSeries(unsigned long type, unsigned long slots)
```

type series! slots series type RedType
series

```
redMakeSeries(RED_TYPE_PAREN, 2); // paren! series

long path = redMakeSeries(RED_TYPE_SET_PATH, 2); // set-path!
redAppend(path, redWord("a"));
redAppend(path, redInteger(2)); // path `a/2:`
```

4.5. CRed

Red C

4.5.1. redCInt32()

```
long redCInt32(red_integer number)
```

Red integer! 32

4.5.2. redCDouble()

```
double redCDouble(red_float number)
```

Red float! C

4.5.3. redCString()

```
const char* redCString(red_string string)
```

Red string! UTF-8 redSetEncoding

4.5.4. redTypeOf()

```
long redTypeOf(red_value value)
```

Red ID RedType

4.6. Redaction

redCall

Redaction action

4.6.1. redAppend()

```
red_value redAppend(red_series series, red_value value)
```

value is *series* series

4.6.2. redChange()

```
red_value redChange(red_series series, red_value value)
```

series is *value* series

4.6.3. redClear()

```
red_value redClear(red_series series)
```

series is series

4.6.4. redCopy()

```
red_value redCopy(red_value value)
```

4.6.5. redFind()

```
red_value redFind(red_series series, red_value value)
```

value is *series* NONE

4.6.6. redIndex()

```
red_value redIndex(red_series series)
```

series word

4.6.7. redLength()

```
red_value redLength(red_series series)
```

series 系列名

4.6.8. redMake()

```
red_value redMake(red_value proto, red_value spec)
```

spec 系列名 proto 系列名

4.6.9. redMold()

```
red_value redMold(red_value value)
```

Red 系列名

4.6.10. redPick()

```
red_value redPick(red_series series, red_value value)
```

series 系列名 value 值

4.6.11. redPoke()

```
red_value redPoke(red_series series, red_value index, red_value value)
```

series 系列名 value 值

4.6.12. redPut()

```
red_value redPut(red_series series, red_value index, red_value value)
```

series 系列名 value 值

4.6.13. redRemove()

```
red_value redRemove(red_series series)
```

series 系列名

4.6.14. redSelect()

```
red_value redSelect(red_series series, red_value value)
```

series 和 *value* 都是 `red_value` 类型的，*value* 为 `NONE` 时，返回 `series` 中所有元素。

4.6.15. redSkip()

```
red_value redSkip(red_series series, red_integer offset)
```

返回 *series* 中从 *offset* 开始的所有元素。

4.6.16. redTo()

```
red_value redTo(red_value proto, red_value spec)
```

spec 和 *proto* 都是 `red_value` 类型的，返回 *proto* 中符合 *spec* 的所有元素。

4.7. Redword 类型

Redword 类型是 Red 类型的一种，它是由 Red 类型派生出来的，它的特点是它只能存储字符串。

4.7.1. redSet()

```
red_value redSet(long id, red_value value)
```

id 是 `long` 类型的，表示 Redword 的 ID，*value* 是 `red_value` 类型的，表示 Redword 的值。返回 `red_value` 类型的，表示 Redword 的值。

4.7.2. redGet()

```
red_value redGet(long id)
```

id 是 `long` 类型的，表示 Redword 的 ID，返回 `red_value` 类型的，表示 Redword 的值。

4.8. Red 类型

Red 类型是 Red 类型的一种，它是由 Red 类型派生出来的，它的特点是它只能存储字符串。libRed 是 Red 类型的库，它提供了 Red 类型的所有功能。word 是 Red 类型的值，它是由 Red 类型派生出来的，它的特点是它只能存储字符串。

4.10.2. redProbe()

```
red_value redProbe(red_value value)
```

value probe value

4.10.3. redHasError()

```
red_value redHasError(void)
```

API error! null

4.10.4. redFormError()

```
const char* redFormError(void)
```

UTF-8 null

4.10.5. redOpenLogWindow()

```
int redOpenLogWindow(void)
```

Red print print 1 0

NOTE Windows

4.10.6. redCloseLogWindow()

```
int redCloseLogWindow(void)
```

1 0

NOTE Windows

4.10.7. redOpenLogFile()

```
void redOpenLogFile(const string *name)
```

name OS

4.10.8. redCloseLogFile()

```
void redCloseLogFile(void)
```

redOpenLogFile() の閉じ処理

NOTE

Microsoft Office のインストールが完了していない場合は、MS Office のインストールが必要です。

4.11. 型定義

libRed API の型定義は Red の型定義と一致します。redTypeOf redMakeSeries() の redDataType() の Red の型定義と一致します。

```
RED_TYPE_<DATATYPE>
```

型定義は Red の型定義と一致します。

5. Visual Basic API

VB の VBA の MS Office の型定義は Visual Basic API の型定義と一致します。

- redBlock(), redPath(), redCall() は Red の型定義と一致します。null は 0 を返します。
- redBlockVB(), redPathVB(), redCallVB() は VB の型定義と一致します。

VisualBasic	Red
vbInteger	integer!
vbLong	integer!
vbSingle	float!
vbDouble	float!
vbString	string!

5.1. 型定義

VB の VBA の libRed の型定義は stdcall の ABI の型定義と一致します。

```
red build libRed stdcall
```

libRed.bas の型定義は Red の型定義と一致します。

5.2. redLogic()

```
Function redLogic(bool As Boolean) As Long
```

VB **boolean** **Red** **logic!** **VisualBasic**

5.3. redBlockVB()

```
Function redBlockVB(ParamArray args() As Variant) As Long
```

block!**VisualBasic**

```
redProbe redBlockVB()           ' VisualBasic  
redProbe redBlockVB(42, "hello") ' [42 "hello"] VisualBasic
```

5.4. redPathVB()

```
Function redPathVB(ParamArray args() As Variant) As Long
```

path!**series****VisualBasic**

```
redDo("a: [b 123]")  
res = redDo(redPathVB("a", "b"))  
Debug.print redCInt32(res) ' 123VisualBasic
```

5.5. redCallVB()

```
Function redCallVB(ParamArray args() As Variant) As Long
```

any-function!

Red**VisualBasic**

```
redCallVB("random", 6); ' 16integer!
```

5.6. **VisualBasic**

Red**VisualBasic**

API

redRoutine()

VB module UserForm

Excel Red Console

```
Sub RegisterConsoleCB()  
    redRoutine redWord("print"), "[msg [string!]]", AddressOf onConsolePrint  
End Sub  
  
Function onConsolePrint(ByVal msg As Long) As Long  
    If redTypeOf(msg) <> red_unset Then Sheet2.AppendOutput redCString(msg)  
    onConsolePrint = redUnset  
End Function
```