**IRBIS64.dll**

IRBIS64.dll – программная библиотека для доступа к [базам данных ИРБИС 64](http://wiki.elnit.org/index.php/%D0%91%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85_%D0%98%D0%A0%D0%91%D0%98%D0%A1) и обработки данных с использованием [языка форматирования](http://wiki.elnit.org/index.php/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_%D0%98%D0%A0%D0%91%D0%98%D0%A1).

IRBIS64.dll использует менеджер памяти ShareMem.

Общий порядок работы с базами данных ИРБИС с помощью библиотеки IRBIS64.dll

Для работы с базами данных ИРБИС с помощью библиотеки IRBIS64.dll предусмотрен следующий порядок:

* Инициализировать переменную типа TIrbisSpace с помощью функции IrbisInit. Эта переменная предназначена для хранения контекста работы с базой и требуется при вызове большинства функций библиотеки IRBIS64.dll.
* Инициализировать структуры isisucw и isisacw с помощью функции irbis\_uatab\_init. Эти структуры требуются для корректной работы некоторых команд форматирования.
* Открыть базу данных с помощью функций IrbisInitMST, IrbisInitTerm, IrbisInitInvContext.
* Работать с открытой базой данных.
* Закрыть базу данных с помощью функций IrbisCloseMst, IrbisCloseTerm или IrbisClose.

Для проверки успешности выполнения IrbisInitMST, IrbisInitTerm и других функций библиотеки IRBIS64.dll принято определять константу **ZERO = 0**.

// Инициализировать запись TIrbisSpace и вернуть указатель на неё.

function IrbisInit: PIrbisSpace; export;

// Открыть файлы MST и XRF.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace;

// DataBase – полный путь к файлам MST и XRF с указанием имени файла, но без расширения

// (предполагается, что эти файлы находятся в одной папке, одинаково называются и отличаются

// расширением);

// ANumberShelfs – количество полок.

// В случае успешного открытия возвращает ZERO.

function IrbisInitMST(SP: PIrbisSpace; DataBase: Pchar; ANumberShelfs: integer): integer; export;

// Открыть инверсный файл.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace;

// DataBase – полный путь к инверсному файлу с указанием имени файла, но без расширения.

// В случае успешного открытия возвращает ZERO.

function IrbisInitTerm(SP: PIrbisSpace; DataBase: Pchar): integer; export;

После окончания работы с библиотекой IRBIS64.dll необходимо освободить занятые во время работы ресурсы с помощью процедуры IrbisClose.

*Примечание: процедура IrbisClose в том числе запускает процедуры IrbisCloseMST, IrbisCloseTerm.*

// Освободить ресурсы, занятые во время работы с библиотекой IRBIS64.dll:

// закрыть файлы, освободить память.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace.

procedure IrbisClose(SP: PIrbisSpace); export;

// Закрыть файлы MST и XRF.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace.

procedure IrbisCloseMST(SP: PIrbisSpace); export;

// Закрыть инверсный файл.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace.

procedure IrbisCloseTerm(SP: PIrbisSpace); export;

Работа с записями на полке

// Прочитать из базы данных запись с указанным MFN и поместить на полку.

// Аналог функции RECORD в ISIS.

// Параметры:

// SP – указатель на переменную типа TIrbisSpace.

// Shelf – полка, на которую будет помещена запись.

// mfn – MFN записи, которую необходимо прочитать.

// В случае успешного выполнения возвращает ZERO.

function IrbisRecord(PS: PIrbisSpace; Shelf, mfn: longint): integer; export;

// Возвращает: порядковый номер поля с заданной меткой или 0, если поле отсутствует в записи.

// Аналог функции FIELDN в ISIS.

function IrbisFieldN(PS: PIrbisSpace; shelf, met, occ: integer): integer; export;

// Получить элемент данных записи, полученной с помощью вызова IrbisRecord или аналога ISISRLOCK.

// Аналог функции FIELD в ISIS.

// Где:

// nf - порядковый номер;

// subfields - подполе или подполя (указываются только символы, идентифицирующие подполе, без зннака ^).

function IrbisField(PS: PIrbisSpace; shelf, nf: integer; subfields: PChar): Pchar; export;

// nf - порядковый номер поля в справочнике

// если nf<1 или nf>кол-ва полей, то поле добавляется последним

// возврат: 0 - успешное завершение <>0 - не успешное

// АНАЛОГ ISIS FLDADD

function Irbisfldadd(PS: PIrbisSpace; Shelf,met, nf: integer; pole: Pchar): integer;export;

// замена поля

// возврат: 0 - успешно; <>0 - неуспешно

// если pole=nil (*) - удаляет!!*

// аналог FLDREP

function Irbisfldrep(PS: PIrbisSpace; Shelf,nf: integer; pole: Pchar): integer;export;

// аналог NFIELDS

function Irbisnfields(PS: PIrbisSpace;Shelf:integer): integer;export;

// вернуть число повторений поля с даной меткой

// аналог NOCC

function Irbisnocc(PS: PIrbisSpace; Shelf,met: integer): integer;export;

// вернуть метку поля по порядковому номеру поля в записи

// аналог FLDTAG

function Irbisfldtag(PS: PIrbisSpace; Shelf,nf: integer): integer; export;

// опустошает запись

function Irbisfldempty(PS:PIrbisSpace;Shelf:integer):integer;export;

// формирование заготовки для новой записи с номером Amfn

function Irbisnewrec(PS: PIrbisSpace; shelf: integer): integer; export;

// изменитьномер записи

function Irbischangemfn(PS: PIrbisSpace; shelf,newmfn: integer):integer;export;

// Вернуть MFN записи, находящейся на полке.

function IrbisMFN(PS: PIrbisSpace; shelf: integer): longint; export;

// Вернуть число, равное: максимальному номеру MFN из имеющихся в базе записей + 1.

// Это тот номер MFN, который будет присвоен первой новой записи, добавленной в базу.

function IrbisMaxMFN(PS: PIrbisSpace): longint; export;

Функции форматёра

[Форматёр](http://wiki.elnit.org/index.php/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D0.BE.D1.80.D0.BC.D0.B0.D1.82.D1.91.D1.80) – модуль, реализующий набор функций для обработки данных с использованием [языка форматирования](http://wiki.elnit.org/index.php/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B_%D0%98%D0%A0%D0%91%D0%98%D0%A1).

// Задать формат для дальнейшей обработки.

// Line - формат.

function Irbis\_InitPFT(PS: PIrbisSpace; Line: PChar): integer; far;

// Выполнить формат.

// shelf - полка с записью

// FmtExitDLL = 'IRBIS64'

// PS.workerbuf - результат выполнения формата.

function Irbis\_Format(PS: PIrbisSpace; shelf, alt\_shelf, trm\_shelf, LwLn: Integer; FmtExitDLL : PChar): integer; far;

Пример использования языка форматирования с динамически формируемой записью (без сохранения в БД):

var

IrbisSpace: PIrbisSpace;

value: String;

formatResult: String;

begin

...

Irbisnewrec(IrbisSpace, 0);

Irbisfldadd(IrbisSpace, 0, 952, 0, PChar('^b' + value));

Irbis\_InitPFT(IrbisSpace, '&uf(''+3C'',"TXT="v952^b)');

Irbis\_Format(IrbisSpace, 0, 1, 0, MAXRECORD, 'IRBIS64');

SetString(formatResult, IrbisSpace.workerbuf, strlen(IrbisSpace.workerbuf));

Функции работы с инверсным файлом

Поисковый ключ term должен иметь длину >=255 используйте тип TKey.

// Найти термин term.

// Возвращает:

// 0 - термин найден

// TERM\_NOT\_EXISTS = -202 в term - следующий термин

// TERM\_LAST\_IN\_LIST = -203 в term - #0

// TERM\_FIRST\_IN\_LIST = -204 в term - первый термин словаря

function Irbisfind(PS: PIrbisSpace; term: Pchar): integer; export;

// дать следующий термин

// возврат 0 и term TERM\_LAST\_IN\_LIST и #0

function Irbisnxtterm(PS: PIrbisSpace; term: Pchar): integer; export;

// дать предидущий термин

// возврат 0 и term TERM\_FIRST\_IN\_LIST и #0

function Irbisprevterm(PS: PIrbisSpace; term: Pchar): integer; export;

// Возвращает число ссылок на термин.

// Функция предназначена для использования после вызова Irbisfind.

function Irbisnposts(PS: PIrbisSpace): longint; export;

// Разобрать ссылку на: mfn, tag, occ, cnt.

// Возможные значения opt:

// opt = 1 функция возвращает mfn (номер записи);

// opt = 2 функция возвращает tag (метка поля);

// opt = 3 функция возвращает occ (повторение поля);

// opt = 4 функция возвращает cnt (номер слова).

// Возвращает: в случае успеха положительное значение (больше ноля).

function IrbisPosting(PS: PIrbisSpace; opt: smallint): longint; export;

// инициализация работы со ссылками - не вызывается напрямую процедура скрыта в FIND

function Irbisinitpost(PS: PIrbisSpace): integer; export;

// Перейти к следующей ссылке.

// Возвращает: 0 - успешно выполнено; -1 - конец списка.

function IrbisNxtPost(PS: PIrbisSpace): integer; export;

// установить указатель постингов на заданном чтобы читать дальше NXTPost с него!!!

function IrbisFindPosting(PS: PIrbisSpace; const Term: PChar; const posting: TifpItemPosting): integer; export;

Пример поиска термина в инверсном файле при помощи функции Irbisfind:

procedure example(IrbisSpace: PIrbisSpace; prefixedTermin: String);

var

isLongTermin: Boolean;

prefixedTermin\_: String;

res: Integer;

irbisKey: TIrbisKey;

currentKey: String;

begin

isLongTermin := length(prefixedTermin) >= LONGTERM;

if not isLongTermin then begin // искомый термин может поместиться в словаре

prefixedTermin\_ := prefixedTermin;

end else begin // искомый термин не может поместиться в словаре

prefixedTermin\_ := Copy(prefixedTermin, 0, LONGTERM);

end;

res := IrbisFind(IrbisSpace, irbisKey);

if (res = ZERO) and (not isLongTermin) then begin

// термин найден

end else begin

// Термин не найден, но искомый термин такой длины,

// что если и был помещён в словарь, то оказался обрезан.

// Попробуем перебрать термины индексного файла,

// которые могут быть началом искомого термина.

while true do begin

SetString(currentKey, irbisKey, strlen(irbisKey));

// если найденный термин индексного файла не соответствует началу искомого термина,

// то прекращаем перебор

if not havePrefix(prefixedTermin\_, currentKey, False) then

break;

// перебрать записи, на которые указывают постинги

for i := 1 to IrbisNPosts(IrbisSpace) do begin

// откроем запись

if IrbisNxtPost(IrbisSpace) <> 0 then break;

if IrbisPosting(IrbisSpace, 1) <= 0 then

continue;

if IrbisRecord(IrbisSpace, shelf, IrbisPosting(IrbisSpace, 1)) <> ZERO then

continue;

// здесь можно читать содержимое записи, чтобы проверить, содержит ли запись искомый длинный термин.

end;

// Если термин был последним в индексном файле, то прекращаем перебор

if res = -203 then break;

// Взять следующий термин из индексного файла

res := Irbisnxtterm(IrbisSpace, irbisKey);

end;

end;

Функции, осуществляющие запись в БД

// опустошение БД

function IrbisDBEmptyTime(IrbisSpace:PIrbisSpace;seconds:integer):integer;export;

// заблокировать БД

function IrbisLockDBTime(IrbisSpace:PIrbisSpace;seconds:integer):integer;export;

// снять блокировку БД

function IrbisUnLockDBTime(IrbisSpace:PIrbisSpace;seconds:integer):integer;export;

// базовая функция записи с полки с актуализацией и разблокировкой

function IrbisRecUpdateTime(IrbisSpace:PIrbisSpace;Shelf:integer;KeepLock:integer;Updif:boolean;seconds:integer;

var result\_update:integer;var result\_updif:integer):integer;export;

// базовая функция актуализации записи

function IrbisRecIfUpdateTime(IrbisSpace:PIrbisSpace;Shelf, mfn:integer;seconds:integer):integer;export;

// базовая функция блокировки записи

// аналог RECORD

function IrbisRecLockTime(IrbisSpace: PIrbisSpace; Shelf,mfn: longint;seconds:integer): integer;export;

// базовая функция разблокировки записи

function IrbisRecUnLockTime(IrbisSpace:PIrbisSpace;mfn:integer;seconds:integer):integer;export;