**Язык форматирования системы ИРБИС**

***Язык форматирования*** ИРБИС – язык манипулирования данными в контексте [*записи*](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#.D0.9C.D0.BE.D0.B4.D0.B5.D0.BB.D1.8C_.D0.B1.D0.B0.D0.B7_.D0.B4.D0.B0.D0.BD.D0.BD.D1.8B.D1.85_.D0.98.D0.A0.D0.91.D0.98.D0.A1) базы данных, представляет собой средство, с помощью которого осуществляется разнообразное представление данных для многих операций и функций системы, в частности, для показа и печати документов, для создания словарей, для импорта/экспорта данных, для формально-логического контроля и т.д.

Общие сведения

**История появления**

Язык форматирования системы ИРБИС базируется на языке форматирования системы CDS/ISIS (ЮНЕСКО).

**Понятие формата**

Совокупность команд и функций, записанных в соответствии с требованиями языка форматирования, называется ***форматом***. В общем случае формат определяет некоторое подмножество данных из записи базы данных, которые затем могут использоваться системой для выполнения определенных функций.

Формат как отдельная структура сохраняется в текстовом файле с расширением [.pft](http://wiki.elnit.org/index.php/%D0%A4%D0%B0%D0%B9%D0%BB%D1%8B_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D0.B0.D0.B9.D0.BB.D1.8B_.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.D0.BE.D0.B2_.D0.98.D0.A0.D0.91.D0.98.D0.A1). Такой файл обычно называют ***файл формата***.

**Начало знакомства с языком форматирования ИРБИС**

Для понимания языка форматирования необходимо быть знакомым с такими понятиями как: поле данных – структурная единица документа (записи), имеющая числовую идентификацию (метку поля) и содержащая один элемент описания (например, первый индивидуальный автор, основное заглавие и т.д.): подполе – структурная единица поля, которая определяется двумя символами, предшествующими собственно данным подполя: первый из этих символов знак ^, второй символ может быть латинской буквой (прописные и строчные не различаются) или цифрой и называется разделителем подполя.

Для новичков может показаться, что некоторые форматы являются очень сложными для понимания. Однако, все форматы, даже громоздкие, состоят по сути из довольно простых команд или функций, разделенных запятыми или пробелами. Кажущаяся сложность связана с тем, что формат может состоять из многих таких команд. В связи с этим, ключом к пониманию форматов является последовательный анализ всех присутствующих в них команд.

Обрабатывая формат, система работает с тремя объектами: запись (документ) базы данных, собственно формат и рабочая область, в которой размещаются выходные данные. Команды выполняются последовательно в порядке их представления в формате. Одни из команд порождают выходные данные (например, значения полей данных), другие - инициируют некоторые действия (например, переход на другую строку, создание нескольких пустых строк и т.д.). Создаваемые данные запоминаются в рабочей области в виде текстовых строк, которые затем передаются для последующей обработки, например, для печати.

Все команды форматирования могут вводиться прописными или строчными буквами или их комбинацией. В форматах могут присутствовать комментарии. Признаком начала комментария является комбинация символов /\*. В качестве комментария принимается часть строки от признака начала комментария до конца строки.

**Выполнение формата**

**Форматёр**

*Форматёр* – модуль, реализующий набор функций для обработки данных с использованием языка форматирования.

В системе ИРБИС 64 присутствует несколько реализаций форматёра:

* серверный форматёр – [входит в библиотеку IRBIS64.dll](http://wiki.elnit.org/index.php/IRBIS64.dll#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D0.B8_.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.D1.91.D1.80.D0.B0),
* клиентский форматёр – включён в код клиентских приложений.

Клиентский форматёр используется клиентскими приложениями.

Серверный форматёр используется:

* серверным [АРМ Администратор](http://wiki.elnit.org/index.php/%D0%90%D0%A0%D0%9C_%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%BE%D1%80),
* в некоторых случаях клиентское приложение передаёт формат на исполнение серверному форматёру, при этом взаимодействие осуществляется через [сервер ИРБИС](http://wiki.elnit.org/index.php/TCP/IP_%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80_%D0%98%D0%A0%D0%91%D0%98%D0%A1_64).

**Экземпляр форматёра**

Значения глобальных переменных хранятся в конкретном запущенном экземпляре форматёра. Таким образом, область видимости переменных оказывается в непосредственной связи с экземпляром форматёра:

* В рамках одного экземпляра форматёра значения глобальных переменных, инициализированные в одном формате, доступны в другом формате.
* В разных экземплярах форматёра – собственные наборы глобальных переменных.

Например:

* В серверном [АРМ Администратор](http://wiki.elnit.org/index.php/%D0%90%D0%A0%D0%9C_%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%BE%D1%80) с момента запуска и до закрытия приложения будет использоваться один экземпляр форматёра, соответственно для всех выполняемых данным АРМ форматов будет один набор глобальных переменных.
* Для любого клиентского АРМ будет использоваться один экземпляр клиентского форматёра.
* В тех случаях, когда клиентское АРМ передаёт форматы на исполнение серверному форматёру, запросы на исполнение форматов могут выполняться разными экземплярами форматёра.

Таким образом, в контексте выполнения формата нельзя рассчитывать на то, что до присвоения глобальным переменным каких либо значений они будут пустыми.

Запись в базе данных, используемая в примерах

Если не оговорено противное, то все примеры по форматированию в последующих разделах ссылаются на запись, приведенную ниже. Эта запись присутствует в базе данных IBIS, поставляемой в дистрибутивном варианте системы в качестве учебной.

**MFN: 34**

600: ^AСухой П.О.^Bо нем

215: ^a383^cил.24л

920: PAZK

999: 0000000

907: ^A19941123^bdsm

964: 78.09.19

10: ^a5-203-01472-8^dЦ.договор.

101: rus

102: RU

900: ^b05^c22

210: ^cВоенное изд-во^aМ.^d1993

700: ^aКузьмина^bЛ.М.

675: 623.746

675: 623.746(092) Сухой П.О.

908: К89

903: 623.746/К89-239165

910: ^A0^BЭ93-18430^C19930830^Dхр

910: ^A2^Dхр

200: ^aКонструктор Сухой. Люди и самолеты

Команды вывода полей

Команды вывода полей – это команды, используемые для извлечения из записи и вывода конкретного поля или подполя. Имеется [специальная команда для извлечения и вывода номера записи MFN](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D0.B0_.D0.B2.D1.8B.D0.B2.D0.BE.D0.B4.D0.B0_MFN), хотя, строго говоря, MFN не является полем (MFN не имеет метки).

**Команда вывода поля**

Для указания на необходимость вывода поля из записи необходимо записать букву **V**, за которой должна следовать метка поля. Это может быть прописная или строчная буква. Ниже приведены примеры применения данной команды.

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| v210 | ^cВоенное изд-во^aМ.^d1993 |
| v200 | aКонструктор Сухой. Люди и самолеты |
| v700 | ^aКузьмина^bЛ.М. |
| v600 | ^AСухой П.О.^Bо нем |

**Команда вывода подполя**

Для вывода конкретного подполя из данного поля следует добавить соответствующий разделитель подполей к метке поля. Заметим, что можно использовать специальный разделитель подполей ^\* для вывода первого подполя, если оно есть. В этом случае, первому подполю необязательно должен предшествовать его разделитель. Разделитель подполей может быть прописной или строчной буквой (обязательно латинской).

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| v210^с | Военное изд-во |
| v200^a | Конструктор Сухой. Люди и самолеты |
| v700^a | Кузьмина |
| v600^\* | Сухой П.О. |

**Вывод фрагмента поля или подполя**

В некоторых случаях может возникнуть необходимость вывести часть поля, которая не является подполем, особенно тогда, когда поле имеет фиксированный формат (например, запись даты в виде ГГГГММДД). Это можно сделать командой смещение/длина, размещаемой сразу за командой вывода поля или подполя, к которому команда применяется.

Эта команда имеет вид:

\*<смещение>.<длина>

где:

* \*<смещение> указывает позицию первого символа фрагмента, который должен быть извлечен из поля или подполя (позиции символов нумеруются с нуля, т.е. первый символ находится в позиции 0, второй - в позиции 1 и т.д.); если смещение не указано, то оно считается равным 0;
* .<длина> указывает количество извлекаемых символов. Если длина опущена, то извлекается все поле (подполе) до конца, начиная с символа, указанного в смещении.

Ниже приведены примеры использования команды по отношению к записи, которая в качестве поля с меткой 1 содержит значение 88-Nov-05.

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| v1\*3.3 | Nov |
| v1.2 | 88 |
| v1\*7 | 05 |
| v1\*7,v1\*2.4 | 05-Nov |
| v1\*7,v1\*2.5,v1.2 | 05-Nov-88 |
| v210.3 | ^cВ |
| v210^c.3 | Вое |

Обратите внимание на различные способы манипулирования полями с подполями в последних двух примерах. Если производится ссылка на поле целиком (например, v210), то нулевое смещение указывает на первую фактическую позицию поля (включая возможные разделители, стоящие в начале), а при ссылке на подполе (например, v210^с) нулевое смещение указывает на первый символ данных, стоящий после разделителя подполей.

**Команда вывода MFN**

Для вывода номера записи в файле документов служит команда MFN, формат которой:

MFN

или

MFN(d)

где d – количество выводимых на экран цифр. Если параметр (d) опущен, то по умолчанию предполагается 6 цифр. Ниже приводятся примеры использования данной команды.

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| MFN | 000034 |
| MFN(3) | 034 |
| MFN(2) | 34 |
| MFN(1) | 3 |

Заметим, что для подавления в MFN впереди стоящих нулей можно использовать [функцию F](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D1.8F_F.28.D0.B2.D1.8B.D1.80-1.2C_.D0.B2.D1.8B.D1.80-2.2C_.D0.B2.D1.8B.D1.80-3.29).

Команды режима вывода

Система может выводить данные в трех различных режимах: ***режим проверки***, ***режим заголовка***, ***режим данных***.

В ***режиме проверки*** поля выводятся в том виде, в каком они хранятся в записи. При этом система не обеспечивает никаких разделителей между полями или экземплярами повторяющихся полей. Пользователь должен обеспечить адекватное разделение полей с помощью команд размещения, литералов или повторяющихся групп. Режим обычно используется для вывода записей с целью проверки правильности введенных данных.

***Режим заголовка*** обычно используется для печати заголовков при выводе указателей и таблиц. Все управляющие символы, введенные вместе с данными, такие как разделители терминов (< и >) игнорируются (за исключением указанных ниже случаев), а разделители подполей заменяются знаками пунктуации.

***Режим данных*** похож на режим заголовка, но дополнительно после каждого поля автоматически ставится точка (.), за которой следуют два пробела (или просто два пробела, если поле заканчивается каким-либо знаком пунктуации). Отметим, однако, что эта автоматическая пунктуация подавляется, если за командой вывода поля следует [суффикс-литерал](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#.D0.9B.D0.B8.D1.82.D0.B5.D1.80.D0.B0.D0.BB.D1.8B).

Когда система форматирует поле, содержащее подполе, в режимах заголовка или данных, она автоматически заменяет имеющиеся разделители подполей знаками пунктуации (при этом разделитель первого подполя, если он имеется, всегда игнорируется). Специальная комбинация символов "><" заме-няется на "; " (а отдельные символы “<” и “>” подавляются), обеспечивая простой способ форматирования полей, содержащих перечень ключевых фраз, заключенных в угловые скобки. Таблица стандартного замещения разделителей подполей выглядит так:

* ^a замещается на "; "
* от ^b до ^i замещается на ", "
* все другие замещаются на ". "

Команды режима вывода представляются в виде Mmc, где:

* М – признак команды режима вывода.
* m следующим образом определяет режим вывода:
  + Р режим проверки;
  + H режим заголовка;
  + D режим данных.
* с – определяет необходимость преобразования букв в прописные:
  + U буквы преобразуются в прописные;
  + L буквы преобразованию не подвергаются.

Команда режима вывода может появляться в формате столько раз, сколько это нужно и действует до следующей команды режима вывода. По умолчанию система использует команду MPL. Ниже приведены примеры использования команды.

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| mpl,v200 | ^aКонструктор Сухой. Люди и самолеты |
| mhl,v200 | Конструктор Сухой. Люди и самолеты |
| mdl,v200 | Конструктор Сухой. Люди и самолеты. |
| mdu,v200 | КОНСТРУКТОР СУХОЙ. ЛЮДИ И САМОЛЕТЫ. |
| mpl,v210 | ^cВоенное изд-во^aМ.^d1993 |
| mdl,v210 | Военное изд-во; М., 1993. |

Команды горизонтального и вертикального размещения

В языке форматирования имеются следующие команды для управления горизонтальным и вертикальным размещением данных:

* **Xn** – вставляет n пробелов;
* **Cn** – табуляция в позицию n;
* **/** – переход на новую строку, если предыдущая строка не была пустой;
* **#** – безусловный переход на новую строку;
* **%** – удаляет пустые строки (если они есть).

Команда / приводит к размещению последующих данных с начала следующей строки. Подряд расположенные команды / эквивалентны одной команде /. Команда / никогда не создает пустых строк.

Команда # выполняет те же действия, что и /, но переход на новую строку является безусловным.

Можно использовать комбинацию /# для создания одной (и только одной) пустой строки. Комбинация ## может привести к созданию одной или двух пустых строк в зависимости от того, была ли пустой текущая строка перед выполнением первой команды #.

Использование команды # может вызвать затруднение в тех случаях, когда выбираемое поле оказывается пустым. Эта ситуация хорошо иллюстрируется на следующем примере:

/#V10/#V20/#V30

Если все поля присутствуют в документе, то в результате поля 10, 20, и 30 будут располагаться с начала строк и каждому будет предшествовать одна пустая строка. Однако, если поле 20 в документе отсутствует, то между 10 и 30 полями будут вставлены две пустые строки. Это может оказаться нежелательным, если действительно требуется, чтобы между полями была пропущена именно одна пустая строка, независимо от наличия или отсутствия некоторых полей. Таким образом, приведенный выше формат не приведет к желаемому результату.

Эту проблему решает команда %. Ее действие заключается в подавлении всех последовательно расположенных пустых строк, если они имеются, между текущей строкой и последней непустой строкой. Таким образом, формат

%##V10%##V20%##V30 ...

приведет к созданию одной и только одной пустой строки между каждым полем, независимо от их наличия или отсутствия в документе.

Команду Сn имеет смысл применять только в случае использования моноширинных шрифтов. Команды / # % не применимы в случае использования в форматах показа средств RTF или HTML. Команды Сn и Xn не работают в [HTML-форматах](http://wiki.elnit.org/index.php?title=HTML-%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B0%D1%85&action=edit&redlink=1).

Литералы

Литерал – это строка символов, заключенная в соответствующие ограничители, которая вносится в выводимый текст в таком виде, как она приведена в формате. Литералы могут использоваться, например, для именования полей.

Существуют литералы трех типов:

* ***Условный литерал*** определяет текст, который будет выведен только в случае присутствия в записи соответствующего ему поля/подполя. Если поле является повторяющимся, то текст будет выведен только один раз, независимо от количества экземпляров поля/подполя. Условные литералы заключаются в двойные кавычки, например:

"Заглавие: "

* ***Повторяющийся литерал*** определяет текст, который будет выведен только в случае присутствия в записи соответствующего ему поля или подполя. Однако, если поле повторяющееся, литерал будет выводиться для каждого экземпляра поля/подполя. Ограничителем для повторяющихся литералов служит символ вертикальной черты:

|Автор: |

* ***Безусловный литерал*** определяет текст, который будет выведен независимо от наличия поля в записи. Безусловные литералы заключаются в одинарные кавычки, например:

'Краткое содержание'

или

'Заголовок'

Литерал не должен содержать ограничителей литерала. Например, безусловный литерал не может содержать одинарную кавычку (хотя может содержать двойные кавычки и/или вертикальные черты).

Условные и/или повторяющиеся литералы связываются с полем или подполем своим расположением в формате. Литералы, предшествующие команде вывода поля/подполя, называемые***префикс-литералами***, выводятся перед содержимым поля/подполя, в то время, как литералы, следующие за командой вывода поля/подполя, называемые ***суффикс-литералами***, выводятся после содержимого поля/подполя.

Если за повторяющимся префикс-литералом непосредственно следует знак **+** (например, **|xxx|+**), то он будет выведен перед каждым, кроме первого, экземпляром поля/подполя.

Если повторяющемуся суффикс-литералу непосредственно предшествует знак **+** (например, **+|xxx|**), то он будет выведен после каждого, кроме последнего, экземпляра поля/подполя.

С полем/подполем может быть связан более чем один литерал. В этом случае различные литералы должны подчиняться следующим правилам и порядку:

* **префикс-литералы:**
  + Один или более условных префикс-литералов. За условным префикс-литералом могут следовать другие условные префикс-литералы, команды вертикального и горизонтального размещения, и/или команды режима вывода. Все команды между первым условным префикс-литералом и соответствующей ему командой вывода поля/подполя становятся условными и будут выполнены только при наличии поля/подполя, иначе они игнорируются.
  + Один и только один повторяющийся префикс-литерал. Если такой литерал есть, то он должен непосредственно предшествовать соответствующей ему команде вывода поля/подполя.
* **суффикс-литералы:**
  + Один и только один повторяющийся суффикс-литерал. Если такой литерал есть, то он должен следовать непосредственно за соответствующей ему командой вывода поля/подполя.
  + Один и только один условный суффикс-литерал. Если такой литерал есть, то он должен следовать непосредственно за повторяющимся суффикс-литералом или связанной с ним командой вывода поля/подполя.
  + Суффикс-литералы не должны разделяться запятыми и не должно быть запятой между командами вывода поля/подполя и первым суффикс-литералом, так как запятая обозначает конец суффикс-литералов, связанных с данной командой вывода поля/подполя.

***Пустые литералы*** – литералы нулевой длины, такие, как "" или || вполне допустимы и могут использоваться, как префикс-литералы для обеспечения условного вертикального размещения или как суффикс-литералы для временного подавления автоматической пунктуации, которая обеспечивается в режиме данных.

Литералы подвергаются преобразованию в прописные буквы, если расположены после соответствующей команды режима вывода.

Примеры различных типов литералов:

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| 'MFN:      ',mfn(3) | MFN:      034 |
| mdl,"Заглавие: "v200 | Заглавие: Конструктор Сухой. Люди и самолеты. |
| v675 | 623.746623.746(092) Сухой П.О. |
| v675|; | | 623.746; 623.746(092) Сухой П.О.; |
| v675+|; | | 623.746; 623.746(092) Сухой П.О. |
| ; |v675 | ; 623.746; 623.746(092) Сухой П.О. |
| ; |+v675 | 623.746; 623.746(092) Сухой П.О. |
| "УДК"/v675+|; | | УДК 623.746; 623.746(092) Сухой П.О. |
| (|v675|)| | (623.746)(623.746(092) Сухой П.О.) |
| "(УДК: ",v675+|; |")" | (УДК: 623.746; 623.746(092) Сухой П.О.) |
| mdl,v210 | Военное изд-во; М., 1993. |
| mdl,v210"" | Военное изд-во; М., 1993 |

Команды фиктивного вывода поля

Команды фиктивного вывода поля реализует условный вывод литералов в зависимости от наличия или отсутствия данного поля или подполя без вывода содержимого самого поля. Задаются они следующим образом: **Dt** или **Dt^x** или **Nt** или **Nt^x**, где:

* **D** указывает, что все соответствующие ей условные литералы должны выводиться только при наличии поля,
* **N** указывает, что они должны выводиться только при отсутствии поля;
* **t** метка поля, управляющего выводом литералов;
* **^x** необязательный код разделителя подполя. Если он приведен, то это означает, что вывод литералов зависит от наличия или отсутствия определенного подполя (отметим, что отсутствие поля говорит и об отсутствии подполей в этом поле).

Обычно команде фиктивного вывода поля предшествует, по крайней мере, один условный префикс-литерал (который может быть пустым), за которым могут следовать один или более других условных префикс-литералов, команды вертикального и горизонтального размещения и/или команды режима вывода. Команда фиктивного вывода поля не может иметь суффикс-литералов.

Несколько примеров использования этих команд:

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| "(Отсут.)"n700,v700 | Конструктор Сухой. Люди и самолеты |
| "(Отсут.)"n80,v80+|; | | (Отсут.) |
| "[нет даты]"n907^А,V907^A | 19941123 |
| "[нет сведений]"n911,v911 | [нет сведений] |

Выражения

Язык форматирования позволяет вычислять значения и/или сравнивать значения с помощью выражений. Выражения построены таким образом, что при их вычислении возвращается значение. Таким значением может быть строка символов (например, содержимое поля или литерал), тогда выражение называется строковым, либо число, тогда выражение называется числовым, либо логическое значение (истина или ложь), тогда выражение называется логическим. В системе также имеется набор функций, которые, на основе задаваемых аргументов, выполняют некоторые специфические действия и возвращают значения. Функции, возвращающие в качестве значения число, называются числовыми функциями, возвращающие строку символов - строковыми функциями, а возвращающие логическое значение - логическими функциями. Только строковые функции могут использоваться непосредственно в качестве команд форматирования. Числовые выражения могут использоваться либо в логических выражениях, либо в качестве аргументов функций. Логические выражения и логические функции могут использоваться только в контексте [команды IF](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D0.B0_IF).

**Числовые выражения**

***Числовое выражение*** формируется из операндов, которые имеют числовое значение, и [арифметических операторов](http://wiki.elnit.org/index.php/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%8B_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.90.D1.80.D0.B8.D1.84.D0.BC.D0.B5.D1.82.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.B8.D0.B5_.D0.BE.D0.BF.D0.B5.D1.80.D0.B0.D1.82.D0.BE.D1.80.D1.8B) (**+**, **-**, **\***, **/**).

В качестве операндов в числовых выражениях могут выступать: ***числовые константы***, ***числовые функции***, [команда вывода номера записи MFN](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D0.B0_.D0.B2.D1.8B.D0.B2.D0.BE.D0.B4.D0.B0_MFN), числовые выражения.

***Числовые константы*** могут быть представлены в виде целых чисел, чисел с десятичной точкой, например 5, 18, 98.65, или чисел в ***экспоненциальной форме***, например, 1.5Е5, обозначающее 1.5 умноженное на 10 в 5-ой степени, т.е. 150000. Числа могут быть со знаком или без.

[***Числовые функции*** рассматриваются в отдельной главе](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.A7.D0.B8.D1.81.D0.BB.D0.BE.D0.B2.D1.8B.D0.B5_.D1.84.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D0.B8). Пример числовой функции val(v10).

Числовые выражения при использовании их в качестве операндов должны быть заключены в круглые скобки, например:

(val(v20)-5)

Отметим, что поскольку команды вывода полей (например, v10 или v10^a) дают текстовые строки, они не могут использоваться в качестве операндов числовых выражений. Однако, для преобразования содержимого поля или подполя в числовое значение может быть использована [функция VAL](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D1.8F_VAL.28.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.29).

Аналогичным образом, числовое выражение не может быть непосредственно выведено на экран или печать, а должно быть предварительно преобразовано в строку символов с помощью[функции F](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D1.8F_F.28.D0.B2.D1.8B.D1.80-1.2C_.D0.B2.D1.8B.D1.80-2.2C_.D0.B2.D1.8B.D1.80-3.29).

Ниже приведены примеры числовых выражений. Предполагается, что MFN=10, v1^a=10, v1^b=20 и v2=30.

|  |  |
| --- | --- |
| **Выражение** | **Значение** |
| 0.155е+3 | 155 |
| 1е-3 | 0.001 |
| 2\*3+9 | 15 |
| 2\*(3+9) | 24 |
| 10-(4\*(2-1)) | 6 |
| 15\*0.001 | 0.015 |
| mfn+100 | 110 |
| val(v2)+val(v1^a)\*7.5 | 105 |
| (val(v1^a)-val(v1^b))/100 | -0.1 |

**Строковые выражения**

***Строковым выражением*** являются:

* [Безусловный литерал](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#.D0.9B.D0.B8.D1.82.D0.B5.D1.80.D0.B0.D0.BB.D1.8B), например 'некоторый текст'.
* [Команда вывода поля/подполя](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D1.8B_.D0.B2.D1.8B.D0.B2.D0.BE.D0.B4.D0.B0_.D0.BF.D0.BE.D0.BB.D0.B5.D0.B9), которая может включать [команду смещение/длина](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#.D0.92.D1.8B.D0.B2.D0.BE.D0.B4_.D1.84.D1.80.D0.B0.D0.B3.D0.BC.D0.B5.D0.BD.D1.82.D0.B0_.D0.BF.D0.BE.D0.BB.D1.8F_.D0.B8.D0.BB.D0.B8_.D0.BF.D0.BE.D0.B4.D0.BF.D0.BE.D0.BB.D1.8F), например v26^c\*2.2.
* [Строковая функция](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A1.D1.82.D1.80.D0.BE.D0.BA.D0.BE.D0.B2.D1.8B.D0.B5_.D1.84.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D0.B8), например S(v24,v25,v26).

**Логические выражения**

***Логическое выражение*** формируется из операндов, которые могут принимать одно из двух значений, **истина** или **ложь**, и [логических операторов](http://wiki.elnit.org/index.php/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%8B_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.9B.D0.BE.D0.B3.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.B8.D0.B5_.D0.BE.D0.BF.D0.B5.D1.80.D0.B0.D1.82.D0.BE.D1.80.D1.8B) (**NOT**, **AND**, **OR**). Логические выражения используются для вычисления истинности одного или более условий.

Операндами логического выражения могут быть:

* [Выражения отношения](http://wiki.elnit.org/index.php/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%8B_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.9E.D0.BF.D0.B5.D1.80.D0.B0.D1.82.D0.BE.D1.80.D1.8B_.D0.BE.D1.82.D0.BD.D0.BE.D1.88.D0.B5.D0.BD.D0.B8.D1.8F_.D0.B8_.D0.B2.D1.8B.D1.80.D0.B0.D0.B6.D0.B5.D0.BD.D0.B8.D1.8F_.D0.BE.D1.82.D0.BD.D0.BE.D1.88.D0.B5.D0.BD.D0.B8.D1.8F) – сравнивают два числовых или строковых значения и определяют, удовлетворяют ли они отношению, строятся на основе [операторов отношения](http://wiki.elnit.org/index.php/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80%D1%8B_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.9E.D0.BF.D0.B5.D1.80.D0.B0.D1.82.D0.BE.D1.80.D1.8B_.D0.BE.D1.82.D0.BD.D0.BE.D1.88.D0.B5.D0.BD.D0.B8.D1.8F_.D0.B8_.D0.B2.D1.8B.D1.80.D0.B0.D0.B6.D0.B5.D0.BD.D0.B8.D1.8F_.D0.BE.D1.82.D0.BD.D0.BE.D1.88.D0.B5.D0.BD.D0.B8.D1.8F) (**=**, **<>**, **<**, **<=**, **>**, **>=**, **:**). Например mfn < 10.
* [Логические функции](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.9B.D0.BE.D0.B3.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.B8.D0.B5_.D1.84.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D0.B8), например p(v24).
* Логические выражения.

Примеры логических выражений:

|  |  |
| --- | --- |
| **Выражение** | **Значение** |
| mfn=34 | Истина |
| not mfn=34 | Ложь |
| not (not mfn=34) | Истина |
| v20 = 'люди' | Ложь |
| v200 : 'люди' | Истина |
| v200 : 'ЛЮДИ' | Истина |
| v210^c.6 = 'Военное' | Ложь |
| v210^c.7 = 'Военное' | Истина |
| (v200 : 'люди') AND (v210^c.7= 'Военное') | Истина |

Функции

***Функция*** вычисляет *значение* (называемое ***значением функции*** или ***возвращаемым значением***), которое затем подставляется вместо функции при вычислении выражения.

Функции могут иметь один или более *аргументов*, которые необходимо задать для вычисления функции. Аргументы заключаются в скобки и разделяются запятыми.

Аргументы могут быть следующих трех типов: [формат](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#.D0.9F.D0.BE.D0.BD.D1.8F.D1.82.D0.B8.D0.B5_.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.D0.B0), [числовое выражение](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.A7.D0.B8.D1.81.D0.BB.D0.BE.D0.B2.D1.8B.D0.B5_.D0.B2.D1.8B.D1.80.D0.B0.D0.B6.D0.B5.D0.BD.D0.B8.D1.8F), [команда вывода поля](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D1.8B_.D0.B2.D1.8B.D0.B2.D0.BE.D0.B4.D0.B0_.D0.BF.D0.BE.D0.BB.D0.B5.D0.B9).

Аргументом функции может быть произвольный формат, который может содержать любую разрешенную команду за исключением [функции REF](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D1.8F_REF.28.D0.B2.D1.8B.D1.80.D0.B0.D0.B6.D0.B5.D0.BD.D0.B8.D0.B5.2C_.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.29), в которой формат так же используется в качестве аргумента. Функции передается не сам формат, а текст, полученный в результате вычисления формата.

При использовании числового выражения в качестве аргумента функции, оно сначала вычисляется, и затем вычисленное значение передается функции.

В качестве аргумента могут быть команды вывода поля или подполя. Не допускается использовать в команде спецификацию смещение/длина.

Список функций с их описанием приведён в статье [**Функции языка форматирования ИРБИС**](http://wiki.elnit.org/index.php/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D1%8F%D0%B7%D1%8B%D0%BA%D0%B0_%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_%D0%98%D0%A0%D0%91%D0%98%D0%A1).

Форматные выходы. UNIFOR

***Форматные выходы*** – вид функций, реализованных на основе определённой технологии вне рамок языка форматирования, и предназначенных для выполнения таких операций по форматированию, которые не могут быть выполнены с помощью собственно языка форматирования. Технология ***форматных выходов*** подразумевает реализацию определённого стандартного интерфейса передачи данных для таких функций и возврата значений. В рамках языка форматирования ***форматные выходы*** являются строковыми функциями с форматом в качестве аргумента.

Форматные выходы имеют следующий синтаксис в языке форматирования:

&имя(формат)

где:

* **&** – идентификатор вызова форматного выхода,
* **имя** – имя форматного выхода,
* **формат** – аргумент.

К использованию предлагается [форматный выход UNIFOR](http://wiki.elnit.org/index.php/UNIFOR), реализующий множество функций вне базовых рамок языка форматирования.

Команда IF

Команда **IF** предоставляет возможность строить контекстно-зависимые форматы, то есть форматы, способные изменять выходные значения в зависимости от содержимого форматируемой записи.

Команда **IF** записывается следующим образом:

**IF** ***условие*** **THEN** ***формат-1*** **ELSE** ***формат-2*** **FI**

где:

* ***Условие*** – [логическое выражение](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#.D0.9B.D0.BE.D0.B3.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.B8.D0.B5_.D0.B2.D1.8B.D1.80.D0.B0.D0.B6.D0.B5.D0.BD.D0.B8.D1.8F),
* ***формат-1*** – формат, который вычисляется в том и только том случае, когда значением логического выражения является **истина**,
* ***формат-2*** – формат, который вычисляется в том и только том случае, когда значением логического выражения является **ложь**.

Фраза **ELSE** ***формат-2*** не является обязательной, и может быть опущена. Ключевые слова **IF**, **THEN** и **FI** обязательны всегда. ***Формат-1*** может быть опущен, если присутствует предложение**ELSE** (то есть если условие истинно, то ничего не будет выведено). Таким образом, команда **IF** может также принимать одну из следующих дополнительных форм:

**IF *условие* THEN *формат-1* FI**

**IF *условие* THEN ELSE *формат-2* FI**

Поскольку нет никаких ограничений на команды, которые можно использовать в ***формате-1*** и ***формате-2***, то команды **IF** могут быть вложены друг в друга любое количество раз. Ключевое слово **FI**, в таком случае, должно закрывать каждую команду **IF** (можно рассматривать **IF** и **FI** как пару скобок). Например:

if p(v1) then v24 else if p(v2) and a(v3) then v5 fi fi

Команда **IF** особенно полезна для создания обобщенных форматов в интегрированных базах данных, которые содержат различные типы записей. В этом случае надо иметь различные метки для каждого типа записи (должно быть поле, содержащее код, идентифицирующий тип записи). Таким образом, проверяя тип документа с помощью команды **IF**, можно порождать в одном формате специфическое форматирование записей каждого типа.

Повторяющиеся группы

**Определение повторяющейся группы**

Повторяющаяся группа состоит из набора команд форматирования, заключенных в круглые скобки. Смысл этих команд обычный, за исключением того, что обработка повторяющихся полей производится особым образом.

Чтобы понять концепцию повторяющейся группы, надо, прежде всего, знать, каким образом система работает с повторяющимися полями. При отсутствии каких-либо дополнительных указаний система трактует все экземпляры (в том порядке, в котором они находятся в записи) как единую строку текста.

Повторяющаяся группа изменяет порядок обработки системой экземпляров повторяющихся полей таким образом, что происходит последовательная обработка отдельно каждого экземпляра повторяющегося поля.

Когда система обнаруживает скобку, открывающую повторяющуюся группу, происходит следующее:

1. Счетчик экземпляров (повторений) устанавливается в единицу.
2. Заключенный в скобки формат обрабатывается таким образом, что команды вывода полей выводят только те экземпляры повторяющихся полей, которые соответствуют текущему значению счетчика экземпляров.
3. Если в процессе текущего просмотра всей повторяющейся группы ничего не выводится (то есть в пределах группы больше не оказалось экземпляров повторяющегося поля), то процесс обработки повторяющейся группы завершается. В противном случае значение счетчика экземпляров увеличивается на 1 и повторяется выполнение шагов 2 и 3.

Отметим еще раз, что все команды форматирования внутри повторяющейся группы обрабатываются последовательно экземпляр за экземпляром (как объяснялось выше), включая, таким образом, и поля, указываемые в команде IF, выражениях и функциях, а также строковых функциях, используемых как команды. Поэтому нежелательно использовать безусловные литералы в повторяющейся группе. И если все же безусловные литералы используются, то они будут выдаваться на один раз больше, чем это надо.

**Использование повторяющейся группы**

В подавляющем большинстве случаев для правильной обработки повторяющихся полей достаточно использование простых команд форматирования, таких, например, как команда режима вывода или повторяющихся литералов, как это показано ниже:

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| mpl,v70 | Grieve, B.J.Went, F.W. |
| mdl,v70 | Grieve, B.J. Went, F.W. |
| v70+|; | | Grieve, B.J.; Went, F.W. |

Однако, возможны случаи, когда появляется необходимость отформатировать повторяющиеся поля другим способом. Например, иногда необходимо форматировать каждый экземпляр поля с новой строки, чего можно достигнуть только с использованием повторяющейся группы, как это показано ниже:

|  |  |
| --- | --- |
| **Формат** | **Выходные данные** |
| v675/v200^a | 623.746623.746(092) Сухой П.О. Конструктор Сухой. Люди и самолеты |
| (v675/),v200^a | 623.746 623.746(092) Сухой П.О. Конструктор Сухой. Люди и самолеты |

В первом случае команда перехода на новую строку (/) выполняется после форматирования всех экземпляров поля 675, а во втором случае она выполняется после каждого экземпляра поля.

Пример ниже иллюстрирует использование повторяющихся полей с подполями (предполагается, что запись содержит два экземпляра поля 20). В первом формате использование повторяющейся группы дает возможность вывести различные подполя каждого экземпляра повторяющегося поля в виде таблицы. Второй формат аналогичен первому, но не содержит повторяющейся группы.

Если нужно вывести литерал перед данными, создаваемыми повторяющейся группой, то можно использовать безусловный или условный литерал. Отметим, однако, что условный литерал должен быть связан с командой вывода поля (повторяющаяся группа не является командой вывода поля). Для этой цели необходимо использовать команды фиктивного вывода поля.

Содержимое записи:

210 ^aNew York^сMcGraw Hill^d1988

210 ^aLondon^cAcademic Press^d1975

Формат 1:

/(v210^a,c11,v210^c,c30,v210^d/)

Вывод:

New York McGraw Hill 1988

London Academic Press 1975

Формат 2:

/v210^a,c11,v210^c,c30,v210^d/

Вывод:

New YorkLondon

McGraw HillAcademic Press

19881975

Для избежания недоразумений, связанных с тонкостями использования повторяющейся группы, рекомендуется воспользоваться следующим типовым примером:

(if p(v20) then ... fi)

где вместо троеточия следует поставить свои команды, которые будут выполнены на каждое повторение поля 20.

Если существуют дополнительные условия на вывод, то их также следует помещать в тело условного оператора в приведённом типовом примере.

Например, вывод всех повторений, кроме первого, можно сделать так:

(if p(v20) then if &uf('+90') <> '1' then ... fi fi)

Повторяющиеся группы не могут быть вложенными (то есть повторяющаяся группа не может содержать другую повторяющуюся группу), за исключением случая, когда внутренняя группа содержится в аргументе формата функции REF. Так, например, допустимым является следующий формат:

(v10,ref(val(v20),v10,(v20,v30)))

но приведенный ниже формат является недопустимым, что вызовет сообщение об ошибке:

(v10,(v20,v30))

Использование повторяющейся группы является обязательным в следующих случаях:

* Использование повторяющегося поля в качестве аргумента функции L.
* Первый аргумент функции REF ссылается на повторяющееся поле.

Также нужно учитывать возможность употребления повторяющейся группы, когда повторяющееся поле используется в логическом выражении команды IF.

Глобальные переменные

Для записи и чтения глобальных переменных предназначены [специальные форматные выходы](http://wiki.elnit.org/index.php/UNIFOR#.D0.A4.D1.83.D0.BD.D0.BA.D1.86.D0.B8.D0.B8_.D0.B4.D0.BB.D1.8F_.D1.80.D0.B0.D0.B1.D0.BE.D1.82.D1.8B_.D1.81_.D0.B3.D0.BB.D0.BE.D0.B1.D0.B0.D0.BB.D1.8C.D0.BD.D1.8B.D0.BC.D0.B8_.D0.BF.D0.B5.D1.80.D0.B5.D0.BC.D0.B5.D0.BD.D0.BD.D1.8B.D0.BC.D0.B8).

Для вывода значения глобальной переменной предназначена команда **g**, после которой должен быть записан номер глобальной переменной.

Например, команда

g1

выводит значение первой глобальной переменной.

Команда **g** в некоторых случаях может служить краткой альтернативой форматным выходам для чтения глобальных переменных.

Об области видимости глобальных переменных см. в [подразделе *Экземпляр форматёра*](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.AD.D0.BA.D0.B7.D0.B5.D0.BC.D0.BF.D0.BB.D1.8F.D1.80_.D1.84.D0.BE.D1.80.D0.BC.D0.B0.D1.82.D1.91.D1.80.D0.B0).

Вложенные форматы

Механизм вложенных форматов позволяет избегать дублирования программного кода.

В качестве ссылки на вложенный формат используется имя файла формата без расширения, обозначенное символами ASCII с кодами 1C (File Separator) и 1D (Group Separator). 1C указывают непосредственно перед именем формата, а 1D – непосредственно после.

В процессе интерпретации вместо таких ссылок на файлы вложенных форматов вставляется их содержимое.

Альтернативным способом выполнения файла форматов является использование [форматного выхода &uf('6…](http://wiki.elnit.org/index.php/UNIFOR).

Использование команд (тэгов) RTF и HTML

В форматах допускается использование команд RTF, обеспечивающих представление неплоских текстов, содержащих шрифтовые выделения, горизонтальную и вертикальную верстку и т.п. Форматы, в которых используются команды RTF, называются RTF-форматы.

Команды RTF вводятся в формат в виде литералов.

Если предполагается, что формат будет содержать RTF команды, то он должен быть целиком заключен в фигурные скобки:

'{'формат '}'

Все команды RTF начинаются с символа \ (обратный слеш).

Любая команда RTF должна заканчиваться знаком пробела, или непосредственно за ней может следовать другая команда RTF. Перечень основных команд RTF:

|  |  |
| --- | --- |
| **Команда** | **Назначение** |
| \par | Начало нового параграфа (абзаца) |
| \liNNN | Левый отступ для всех строк параграфа – NNN (в пунктах) |
| \riNNN | Правый отступ для всех строк параграфа – NNN (в пунктах) |
| \fiNNN | Левый отступ для первой строки параграфа – NNN (в пунктах) |
| \pard | Начало форматирования нового абзаца, все предыдущие команды форматирования параграфа отменяются |
| \fsNNN | Размер шрифта – NNN (удвоенный размер в пунктах) |
| \ul \ul0 | Начало и конец подчеркивания |
| \i \i0 | Начало и конец курсива |
| \b \b0 | Начало и конец жирности |
| \tab | Табуляция |
| \qr | Правое выравнивание строк |
| \ql | Левое выравнивание строк |
| \qc | Цетральное выравнивание строк |
| \qj | Равномерное выравнивание строк |
| \page | Переход на новую страницу |
| \txNNN | Позиция табуляция – NNN (в пунктах) |

В форматах, содержащих команды RTF, неприменимы [команды вертикальной верстки](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D1.8B_.D0.B3.D0.BE.D1.80.D0.B8.D0.B7.D0.BE.D0.BD.D1.82.D0.B0.D0.BB.D1.8C.D0.BD.D0.BE.D0.B3.D0.BE_.D0.B8_.D0.B2.D0.B5.D1.80.D1.82.D0.B8.D0.BA.D0.B0.D0.BB.D1.8C.D0.BD.D0.BE.D0.B3.D0.BE_.D1.80.D0.B0.D0.B7.D0.BC.D0.B5.D1.89.D0.B5.D0.BD.D0.B8.D1.8F).

В форматах показа могут использоваться HTML-теги в качестве альтернативы RTF-командам. Совместное использование RTF-команд и HTML-тегов недопустимо.

Форматы, в которых используются HTML-теги, называются HTML-форматы.

HTML-теги вводятся в формат в виде литералов.

Если требуется явно определить формат как HTML-формат, необходимо в самом начале формата указать фиктивный тэг:

'</>'

В форматах, содержащих HTML-теги, неприменимы [команды вертикальной верстки, а также команды Xn и Cn](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#.D0.9A.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D1.8B_.D0.B3.D0.BE.D1.80.D0.B8.D0.B7.D0.BE.D0.BD.D1.82.D0.B0.D0.BB.D1.8C.D0.BD.D0.BE.D0.B3.D0.BE_.D0.B8_.D0.B2.D0.B5.D1.80.D1.82.D0.B8.D0.BA.D0.B0.D0.BB.D1.8C.D0.BD.D0.BE.D0.B3.D0.BE_.D1.80.D0.B0.D0.B7.D0.BC.D0.B5.D1.89.D0.B5.D0.BD.D0.B8.D1.8F).

Предлагаются дополнительные возможности при адресации ресурсов ИРБИС в тегах <A HREF=…> и <IMG SRC=…>:

<A HREF="IRBIS:N,описание\_ссылки">

где:

* N – число, обозначающее вид ссылки. Принимает значения 0, 1, 2, 3.
  + Для значений 0, 1, 2 – структура и смысл ссылок аналогичен ссылкам, формируемым с помощью форматного выхода &unifor(‘+I..). Важное замечание: форматный выход &unifor(‘+I..) ПРИМЕНИМ ТОЛЬКО В RTF-форматах (т.е. в форматах с HTML-тэгами он не применим).
  + Значение 3 определяет ссылку на внешний объект через относительную адресацию в рамках системы ИРБИС:

<A HREF="IRBIS:3,path,dbname,filename">

где:

* path – определяет относительный путь к внешнему объекту в рамках ИР-БИС и принимает значения: 0 – основная директория системы (для ИРБИС32 – та, где находятся исполняемые модули; для ИРБИС64 – та, где находятся ис-полняемые модули сервера); 1 – общая директория баз данных (по умолчанию \DATAI); 10 – директория конкретной БД;
* dbname – имя БД (имеет смысл только при path=10). По умолчанию – предполагается текущая БД;
* filename – имя файла внешнего объекта.

<IMG SRC="IRBIS:path,dbname,filename">

где параметры path, dbname, filename имеют тот же смысл, что указано выше, и определяют ссылку на графический ресурс (картинку) в рамках ИРБИС.

Реализуется специальная возможность для отображения штрих-кодов в HTML-форматах, а именно:

<IMG SRC="IRBIS:!!ШТРИХ\_КОД!!">

Форматы для формально-логического контроля (ФЛК)

Для формально-логического контроля данных (как на уровне поля, так и документа в целом) применяются специальные форматы. Форматы для ФЛК используются следующим образом:

* форматированию подвергается контролируемый документ;
* первый символ результата форматирования определяет результат ФЛК, а именно:
  + 0 – означает положительный результат контроля;
  + 1 – означает отрицательный результат и обнаруженные ошибки считаются непреодолимыми, т.е. подлежат обязательному устранению;
  + 2 – означает отрицательный результат, но при этом ошибки считаются преодолимыми, т.е. их можно не исправлять.
* остальной результат форматирования (начиная со второго символа) в случае отрицательного контроля выдается пользователю в качестве сообщения.

Собственно форматы ФЛК, как правило, содержат команды IF. Многочисленные примеры таких форматов находятся в директории БД IBIS.

Форматы с переменными метками

В некоторых режимах работы системы применяются форматы, в которых используются, так называемые, переменные метки (например, при вводе данных через словарь – т.н. форматы для переноса данных). [Исходным для таких форматов обязательно является термин словаря.](http://wiki.elnit.org/index.php/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%B8%D0%B8_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%B0#.D0.9F.D0.BE.D0.B8.D1.81.D0.BA.D0.B8_.D0.BF.D0.BE_.D1.81.D0.BB.D0.BE.D0.B2.D0.B0.D1.80.D1.8E)

Переменные метки обозначаются в таких форматах в виде \*\*\* (три символа *звёздочка*). Форматы с переменными метками используются следующим образом:

* Определяется первая [индексная ссылка](http://wiki.elnit.org/index.php/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0_%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%B0_%D0%BF%D0%BE%D0%BB%D0%B5%D0%B9#.D0.A2.D0.92.D0.9F_.D0.B4.D0.BB.D1.8F_.D0.B8.D0.BD.D0.B2.D0.B5.D1.80.D1.82.D0.B8.D1.80.D0.BE.D0.B2.D0.B0.D0.BD.D0.BD.D0.BE.D0.B3.D0.BE_.D1.84.D0.B0.D0.B9.D0.BB.D0.B0) для исходного термина: а именно, соответствующие номер документа (MFN), метка поля (TAG) и повторение (OCC).
* Все переменные метки в формате заменяются на значение найденной метки поля (TAG) и форматированию подвергается документ с найденным номером (MFN).
* В качестве результата форматирования берется строка с номером OCC (нумерация начинается с 1). Если нет строки с таким номером, результатом форматирования является пустая строка.

Примеры форматов с переменными метками находятся в директории БД IBIS.