CoreSolutions Software Inc.

FileMaker Pro Naming Standards

Version 1.3

Exceptions to FileMaker Inc.'s FileMaker Development Conventions

CoreSolutions Software Inc. provides this paper to its employees and clients to document the differences that exist in naming conventions between FileMaker Inc.'s FDC document and CoreSolutions' development practices. Please see the Adherence section at the end of this document for more information.

Where some naming conventions are left to the discretion of the developer, consistency must be an underlying theme in any project.

Contact information:

CoreSolutions Software Inc. | 1-1615 North Routledge Park | London, ON N6H5L6

T: 519.641.7727 | TF: 800.650.8882 | W: www.coresolutions.ca

Table Occurrences

- Should be arranged in disconnected Table Occurrence Groups (TOG's) that support certain functionality within the solution. This is similar to the FTOG method detailed in the FDC document.
- Include a TOG for base tables only. These table occurrences will enforce referential integrity (cascading delete rules) and will serve as the table occurrence for items like calculations, solution update imports etc. These TO's will be named with a leading underscore, a 2- or 3-digit serial number, a 3-character base table code and the base table name.
- The TOG Wrapper is a text box that encloses the TOG. It will show the TOG group code in all caps with a description in plain text and will show the number series to use for the serial number. Also included will be the last used serial number. Each time a new TO is added to the TOG, its serial value is updated in the wrapper. The colour of the wrapper may be used to better separate different TOGs on the graph.

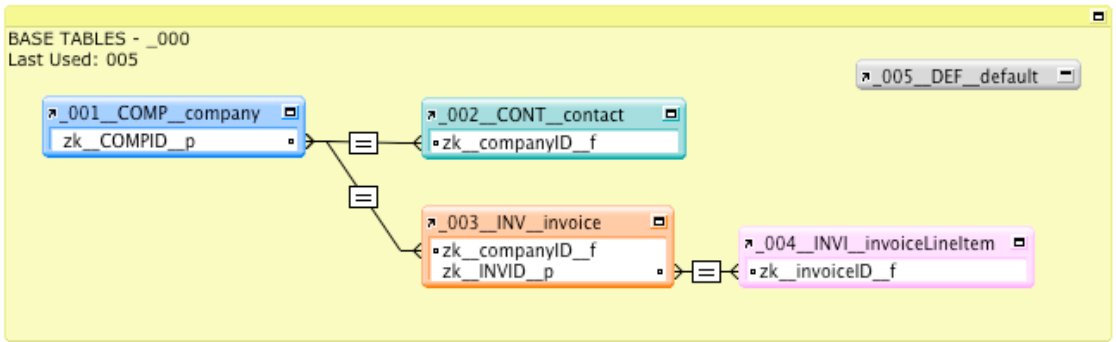
Base Table Group

Syntax:
[_]<<SerialNumber>>[__]<<BaseTableCode>>[__]<SourceTableName>

- **Base Table Code:** 2 to 4 character short form of the table name, all caps

Examples:

_001__COMP__company
_002__CONT__contact
_003__INV__invoice



CORESOLUTIONS software

Other Table Groups

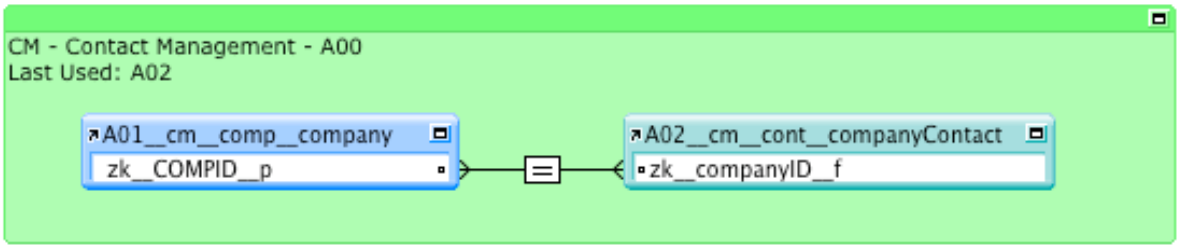
Syntax:
<<GroupLetter>><SerialNumber>[__]<<FunctionalGroupCode>>[__]<<BaseTableCode>>[__]DescriptiveName

- **Functional Group Code:** 2 to 4 character code that names the functional group in lower case. Indicated as the "title" of the enclosing text box around the TOG.
- **Group Letter:** each TOG should have a letter assigned to it by the developer. It could be, but doesn't need to be related to the Group Code. The intent is to put the serial number near the beginning of the TO name to make it easier to select from a list of TO's or to jump to a TO in the graph by typing the first few characters. Since FileMaker Pro does not permit a TO with a leading number, the Group Letter must precede it. In large systems the Group Letter may be two letters. The letter's case is left to the discretion of the developer.
- **Serial Number:** each TOG should have a series of 2- or 3-digit numbers named in the enclosing text box (TOG wrapper) of the TOG (ex: CM – Contact Management – 000). The developer should keep the last used number in that text box (ex: Last used: 245). There is no sequence to the numbers with relation to the TO's; it is left to the developer to decide what span the numbers should have. Since some dialogs display related TO's in alpha order, the numbering will affect that list and the developer can use this property to control the sort order of the TO's in a TOG. The numbering for each TOG does not need to be unique since the Group Letter will make re-used numbers unique. In other words TO's with a prefix of A001 and B001 are allowed. Some developers may prefer to use 2-digit numbers. Some developers may wish to use larger numbers to designate a sub-group within the TOG: D401, D402.
- **Base Table Code:** 2 to 4 character code from the base table occurrence in lower case.
- **Descriptive Name:** a name that describes the TO's function or purpose.

Example 1:

TOG wrapper
CM – Contact Management –A00
Last used: A02

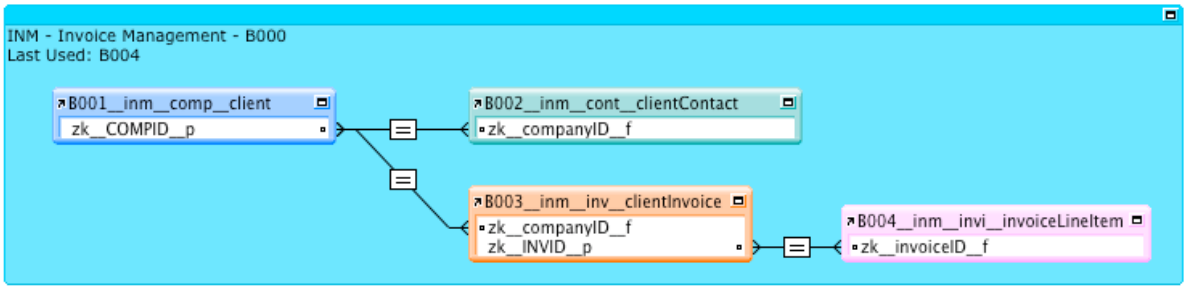
A01__cm__comp__company
A02__cm__cont__companyContact



Example 2:

TOG wrapper
INM – Invoice Management – B000
Last used: B004

B001__inm__comp__client
B003__inm__inv__clientInvoice



Field Names

Key Fields

The syntax used in the FDC paper (leading underscore for key fields only) is inconsistent with other aspects of the paper. In addition, it is not necessary to connect to table occurrences using key fields when the relationship is first being created: dragging any field from one TO to the other and double-clicking on the "=" symbol will bring up the Edit Relationship dialog. From here, clicking in either the right or left TO field list and typing the first few characters of the field you want will auto-scroll to that field. We feel we would rather spend that extra second editing the relationship than sacrifice consistency in field naming. Of course, keeping our original zk prefix for keys means less confusion for our developers too.

Storage is different from the FDC as well since all fields are assumed to be locally stored unless they are global.

Syntax:

[zk__]DescriptiveName[__]<function><storage><type>

- **zk__:** (lower case) Will serve to denote the field is a key field and sort it among the other developer fields.
- **DescriptiveName (Camel Case):** Key Field Descriptive Name. For Primary keys, this should be the table code in upper case followed by "ID". For foreign keys, use a descriptive name in camel case followed by "ID".
- **__:** Double Underscores should be used to separate the key syntax from the name chosen for the field.
- **function (lower case):** Denotes key field category and/or function it will serve.
  - p – Primary Key
  - f – Foreign Key
  - a – Alternate Key
  - c – Compound / Concatenated / Calculated Key
  - m – Multi-Line Key
- **storage (lower case):** Denotes field storage. All key fields are assumed to be locally stored except those that are global.
  - g = Globally Stored

- **type (lower case):** Denotes field type. Only necessary for non-primary or non-foreign keys since those are ALWAYS text.
  - t = Text
  - n = Number
  - d = date
  - i = Time
  - m = Time Stamp

Examples:

zk__INVID__p

zk__companyID__f

zk__selectedPortalRow__agt

Developer Fields

Two changes from the FDC: prefix always starts with "z" but can be further defined by its second function character. Storage follows the same rules as above.

Syntax:
[z]{function}[__]DescriptiveName[__]><storage><type><repetition>

- **z:** Indicates the field as a developer field and sorts to bottom of all dialogue boxes where field names are present.

- **function:** Provides a developer defined categorization for extending as necessary. As the developer you are able to define the various categories. Suggestions:
  - c = Calculations used by the developer
  - k = Key fields (See definition and syntax of Key Fields above)
  - i = Interface fields used to control or manipulate the interface
  - v = Variable or temporary fields

- **__:** Double Underscore denotes separation of prefix and suffix from Descriptive Name

- **DescriptiveName:** Developer selected desired name. Follows general field guidelines.

- **storage:** Used to denote the field storage. All fields are assumed to be locally stored so only designating the globally stored fields is required.
  - g = Globally Stored

- **type:** Indicates the data type used or returned.
  - Non-Calculated "Standard" Result Types
    - t = Text
    - n = Number
    - d = Date
    - i = Time
    - m = Time Stamp
    - r = Container
  - Calculated Result Types
    - ct = Text
    - cn = Number
    - cd = Date
    - ci = Time
    - cm = Time Stamp
    - cr = Container
  - Summary Result Type
    - s = Summary

- **repetition:** indicates the data is stored with repetitions
    - p = repetitions

Field Suffix Reference	Text	Number	Date	Time	Time Stamp	Container
Utility Fields / General & Calculated Fields Using Notation						
General (local Storage)	__t	__n	__d	__i	__m	__r
General (Global Storage)	__gt	__gn	__gd	__gi	__gm	__gr
Calculated (local Storage)	__ct	__cn	__cd	__ci	__cm	__cr
Calculated (Global Storage)	__gct	__gcn	__gcd	__gci	__gcm	__gcr
Repeating (Add "p" to end of notation, in lowercase)	p	p	p	p	p	p
Summary (Local Storage)	N/A	__s	N/A	N/A	N/A	N/A

Examples:

invoiceTotal__s

zc__nameModifier__t

zc__dateModified__d

zv_totalPages__gnp

zc_recordCount__cn

zc_currentUserName__gct

Layouts

For systems where the Status Toolbar is exposed to the user, the developer should use human-readable layout names. For other layouts not exposed to the user (reports, or where the toolbar is not exposed) use the layout naming convention described below.

Syntax:
<<FunctionPrefix>>[__]DescriptiveName{[__]orientationCode}

- **Function Prefix:** Provides general syntax to allow for developer defined prefixing determined by their specific needs and preferences, while defining a universally understood syntax. If you choose not to use any Functional Prefixing simply omit it and start with the Logical Name. There is no need to include the first "__". The general recommendation is to use a one to five character indicator and select a case convention for the function.

- **Descriptive Name:** Provides method to give a 'somewhat' user-friendly name to the layout.

- **Orientation Code:** for printed layouts use an optional orientation code to describe the layout's orientation and page size using the following codes. Scripts can use this suffix to programmatically determine the correct page setup script step to execute.
  __pleg = portrait, legal
  __plet = portrait, letter
  __lleg = landscape, legal
  __llet = landscape, letter

- **Separators:** The usage of "__" double underscores is the character. This allows the usage of underscores within Descriptive Name and Table Occurrence Name, while still providing readability and parsing capabilities.

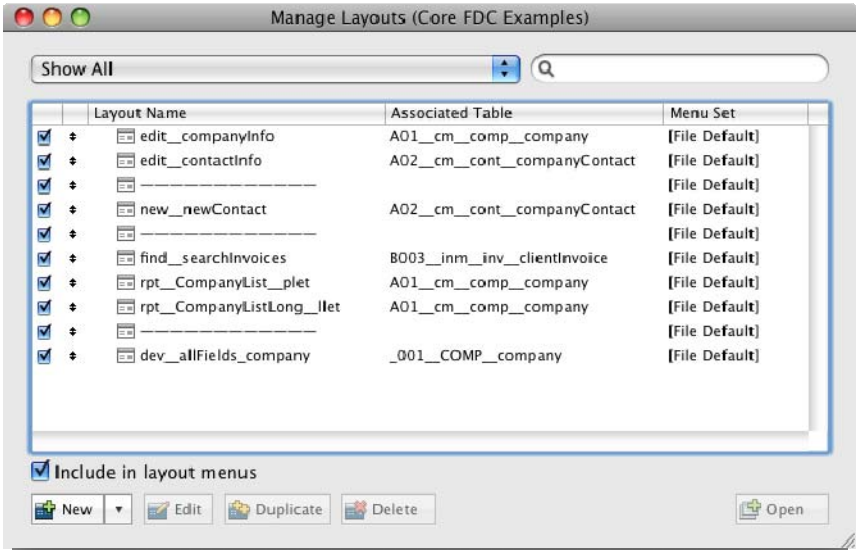
Examples:

edit__companyInfo

find__searchInvoices

dev__allFields_company

rpt__CompanyListLong__llet


Manage Layouts (Core FDC Examples)
Show All
Layout Name
Associated Table
Menu Set
edit__companyInfo
A01__cm__comp__company
[File Default]
edit__contactInfo
A02__cm__cont__companyContact
[File Default]
new__newContact
A02__cm__cont__companyContact
[File Default]
find__searchInvoices
B003__inm__inv__clientInvoice
[File Default]
rpt__CompanyList__plet
A01__cm__comp__company
[File Default]
rpt__CompanyListLong__llet
A01__cm__comp__company
[File Default]
dev__allFields_company
_001__COMP__company
[File Default]
Include in layout menus
New
Edit
Duplicate
Delete
Open

Custom Functions

Public Custom Functions

Syntax:
CustomFunctionName[__][cf]

- **Suffixed with "__cf"** (cf lowercase): Identifies within any calculation that the function is a custom function. All functions are assumed to be a public type unless otherwise indicated.

Examples:

ArrayBuilder__cf

DateFormat__cf

Private / Subordinate Custom Functions

A Private / Subordinate Custom function is one intended only to be called by another custom function, whether by a Public Custom function or another private custom function. We have chosen to use the syntax to indicate Private/Subordinate using the "Sub" term to make it consistent with the ordination of scripts. Families of custom functions, where there is at least one subordinate function, should be named similarly; the names should start out the same (see the example below).

Syntax:
CustomFunctionName[__][cfSub]

- **Suffixed with "__cf"** (cf lowercase, Sub in upper Camel Case): Identifies within any calculation that the function is a custom function and is of the Subordinate type.

Examples:

Luhn__cf

LuhnProduct__cfSub

LuhnDoubleEven__cfSub

Custom Function Parameters

Parameters defined in a custom function cannot be share the same name as a field or another function. For that reason, all parameters for custom functions should have a leading underscore. Using this method, it's possible to use a parameter of "_Date" without impinging on the existing Date function where any other name for the parameter might not be suitable.

Syntax:
CustomFunctionName[__][cf] (_parameterName1, _parameterName2)

- Parameters have a leading underscore and are in lowerCamelCase

Examples:

FormatDate__cf (_date, _long1, _day1)

TimeFormatAs__cf (_timeField, _type12or24)

Scripts

Robust solutions tend to have many scripts. As a result, finding the script that may be attached to a button is often cumbersome. For that reason, a numbering convention has been developed to easily refer to the script and to find it in a list.

Syntax:
<<NumberPrefix>>[ ]DescriptiveName[ ]{[]<ExpectedParameter>{;}{]}

Number Prefix: this is a four-character number with leading zeroes. Scripts can be numbered by one within a functional grouping (0401, 0402, etc) or by some multiple to allow "space" for future scripts (0405, 0410, 0411, 0415, etc). The developer may also wish to use decimals to denote sub-scripts or add a script in a particular order where the numbering has been done by one and no "space" exists (0401, 0401.1, 0401.2, 0402, etc).

Descriptive Name: detailed description of the function or purpose of the script. May also include text to indicate expected script parameters.

ExpectedParameter: if a script is designed to accept parameter(s), there are two different formats that can be used. If the parameter is a named parameter, indicate it with a leading underscore (_) and its name. If the script is expecting a simple value or one of several expected values, list the values separated by a slash.

Examples:

0101 Opening Script

0401 New Record

0401.1 New Record [Commit]

0601 Sort Column Contact [1/2/3/4/5/6]

0801 Tab Navigation [_table ; _view]

Script Groups

Scripts should be grouped by function with a folder as the header for the group.

Syntax:
<<NumberPrefix>>{ —— }<FunctionOrGroupName>{ —— }

Number Prefix: this is a four-character number with leading zeroes. Group numbers should be a multiple of 100.

Function or Group Name: Indicates what the group's basic function is. Over the years, common groups have been developed that most solutions will require. The developer is free to make up a new group or change the name of existing ones.

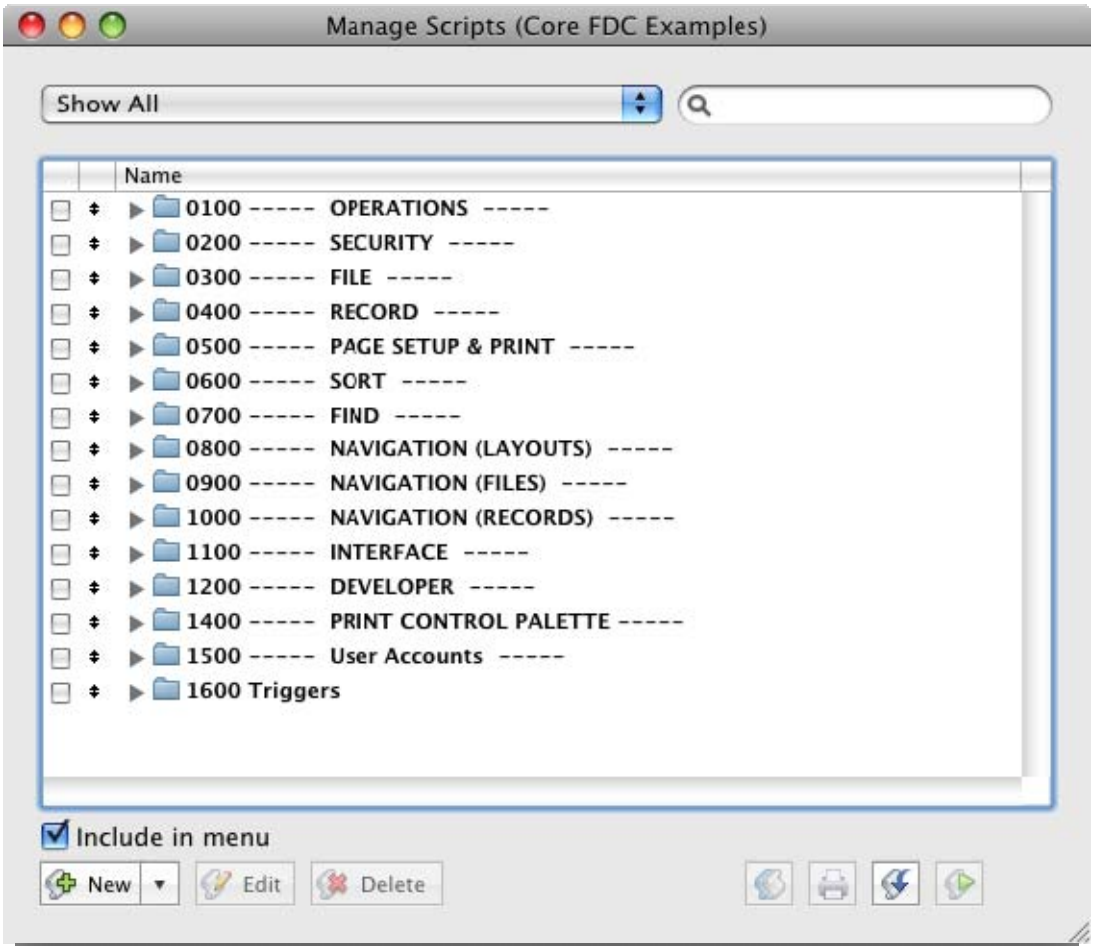
Examples:

0100 —— OPERATIONS ——

0500 —— PAGE SETUP & PRINT ——

1600 Triggers


Manage Scripts (Core FDC Examples)
Show All
Name
0100 OPERATIONS
0101 Opening Script
0102 Closing Script
0103 Quit
0104 Solution Update
0105 Declare Global Variables
0200 ----- SECURITY -----
0201 Check Permission for Child Records
0300 ----- FILE -----
0301 Solution Update
0400 ----- RECORD -----
0401 New Record
0401.1 New Record [Commit]
0403 New Record [Cancel]
0404 Delete Current Record
0405 Delete Portal Row
0406 Create Related Records
0500 ----- PAGE SETUP & PRINT -----
0501 Report using Control Palette
0502 Page Setup
0600 ----- SORT -----
0601 Sort Column Contact [1/2/3/4/5/6]
0700 ----- FIND -----
0701 Find
0702 Find All
0800 ----- NAVIGATION (LAYOUTS) -----
0801 Tab Navigation [_table ; _view]
0900 ----- NAVIGATION (FILES) -----
0901 Go to Related Record [from client]
1000 ----- NAVIGATION (RECORDS) -----
1001 Goto Record
1100 ----- INTERFACE -----
1101 Set WindowSize
1102 Zoom[100%; Status[Hide] LOCK
1103 Zoom[100%; Status[Show] UNLOCK
1104 Hide Window
1105 Close Current Window
1106 Set Date User in Notes
1200 ----- DEVELOPER -----
1201 Clear Input Globals
1202 Save Window Parameters
1203 Update Serial Numbers
Include in menu
New
Edit
Delete


Manage Scripts (Core FDC Examples)
Show All
Name
0100 ----- OPERATIONS -----
0200 ----- SECURITY -----
0300 ----- FILE -----
0400 ----- RECORD -----
0500 ----- PAGE SETUP & PRINT -----
0600 ----- SORT -----
0700 ----- FIND -----
0800 ----- NAVIGATION (LAYOUTS) -----
0900 ----- NAVIGATION (FILES) -----
1000 ----- NAVIGATION (RECORDS) -----
1100 ----- INTERFACE -----
1200 ----- DEVELOPER -----
1400 ----- PRINT CONTROL PALETTE -----
1500 ----- User Accounts -----
1600 Triggers
Include in menu
New
Edit
Delete

Adherence

CoreSolutions' FileMaker Pro Naming Standards conforms to the FileMaker Development Conventions detailed in the white paper dated Nov. 1, 2005. Areas that deviate from that document are stated above. It is intended that, according to the Adherence section of the FDC document, this document and the FDC whitepaper accompany any documentation supplied to our clients. The reason for this is so that if future development is required, documentation detailing our naming conventions and development standards is available to the client in the form of this document and FileMaker's FDC whitepaper (currently located at http://www.filemaker.com/downloads/pdf/FMDev_ConvNov05.pdf). The text from this document may also be included on a screen available from a solution's About screen.

Syntax Legend

The FDC document uses a specific style to indicate the various components of any syntax used within the various sections. The following table explains the various notations.

FDC Syntax Legend	
	No Enclosing Characters indicates optional and natural/free form naming
[ ]	Enclosed component are intended to be used as displayed
{ }	Enclosed components are optional and definable by developer
< >	Enclosed components are intended to be supplied but have a defined or inferred value
<< >>	Enclosed components are intended to be supplied and definable by developer

Version History

1.3 – December 2010 – changed naming rules for primary keys, layouts, scripts, script groups, updated graphics

1.2 – February 2010 – changed naming rules for table occurrences, layouts

1.1 – May 2007

1.0 – January/February 2006