FileMaker® Pro 13

Functions Reference

Edition: 01

Contents

Functions reference (alphabetical list)

About functions

Working with formulas and functions

A function is a predefined, named formula that performs a specific calculation and returns a single, specific value.

Most functions include three basic parts:

- the function
- a set of parentheses, if the function takes parameters
- the parameters required by the function

Each function returns a result of field type text, number, date, time, timestamp, or container.

A formula calculates a single value, based on constants (such as 1.07 or “hello”), operators (such as “+” or “>”), and field references (such as Subtotal or InvoiceTotal) you enter. For example, if sales tax in your area is 7% and you have a field named Subtotal, you could create a field named InvoiceTotal that gets the value of the formula `Subtotal * 1.07`.

If a formula is especially common or popular, FileMaker Pro gives it a name and defines exactly how you should use it. A named and predefined formula is called a function. For example, if you want to find the average of some test scores, you could write your own formula to add them all and divide by the total number of scores. A simpler approach would be to use the function named `Average` and follow the rules defined for its use.

Using this functions reference

The content in this document was originally written for the FileMaker Pro and FileMaker Pro Advanced Help. It has been collected in this format to allow solution developers to read the information independent of the help system. Links to help topics may not work in this format.

Aggregate functions

Aggregate functions perform statistical analysis on numbers (and also dates or times for some functions) in:

- several fields in a record.
- related fields whether displayed in a portal or not.
- repeating fields.

For example, you can use the `Sum` function to add the values listed in a portal, as an alternative to creating a report with grouped data and subtotals.

The parameter values can include a numeric constant (for example, 10) or any valid expression. A constant parameter in a formula for a repeating field affects the result for every repetition.

When repeating field parameters (field1; field2;...) include a non-repeating field, that value is used in the result for only the first repetition unless you use the Extend function.

Values in repetitions that exceed the number of repetitions in the calculated field are ignored. For example, a calculated field with three repetitions holds only three results, even when one field referenced in the calculation has five repetitions.

Click a function name for details.

This function	Returns
Average	The average of all valid, non-blank values in the specified field.
Count	The number of valid, non-blank values in the specified field.
List	The concatenation of all non-blank values in list form, separated by carriage returns.
Max	The highest valid value in a field or fields.
Min	The smallest valid non-blank value in a field or fields.
StDev	The standard deviation of a series of valid non-blank values in a field or fields.
StDevP	The standard deviation of a population represented by a series of valid non-blank values in a field or fields.
Sum	The total of all valid, non-blank values in the specified fields.
Variance	The variance of a series of valid non-blank values in a field or fields.
VarianceP	The variance of a population in a series of valid non-blank values in a field or fields.

Average

Purpose

Returns a value that is the average of all valid, non-blank values in `field`.

Format

```
Average(field{;field...})
```

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record`(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

Examples

A Student table has a portal showing scores for all exams a student has taken. The exam scores are in a table called Exams.

`Average(Exams::Score)` returns the student's average score for all exams she has taken.

In the following examples:

• Field1 contains two repetitions with values of 1 and 2.

• Field2 contains four repetitions with values of 5, 6, 7, and 8.

• Field3 contains 6.

`Average(Field2)` returns **6.5** when the calculation isn't a repeating field.

`Average(Field1;Field2;Field3)` returns **4**, **4**, **7**, **8** when the calculation is a repeating field.

Note When a referenced field is a repeating field, the `Average` function returns the average of the values in the first repetition field, then the average of the values in the second repetition field, and so on. Therefore, **(1+5+6)/3=4;(2+6)/2=4;7/1=7;8/1=8**.

Count

Purpose

Returns the number of valid, non-blank values in `field`.

Format

`Count(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

Examples

The Accounts layout has a portal showing installment payments made.

`Count(Payments::Payment)` returns the number of payments made on an account.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains 6.

`Count(Field2)` returns **4** when the calculation isn’t a repeating field.

`Count(Field1;Field2;Field3)` returns **3**, **2**, **1**, **1** when the calculation is a repeating field.

Note When a referenced field is a repeating field, the `Count` function returns the total number of valid, non-blank values in the first repetition field, then the number of valid, non-blank values in the second repetition field, and so on.

List

Purpose

Returns a concatenated list of non-blank values (separated by carriage returns) for a field or fields.

Format

`List(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; an expression that returns a field, repeating field, or set of non-repeating fields, or a variable.

Parameters in curly braces { } are optional.

Data type returned

text

Originated in

FileMaker Pro 8.5

Description

Use this function to return a list of values for:

- a single field `(table::field)`, which returns a single result over all repetitions (if any) for this field and over all matching related records, whether or not these records appear in a portal.
- several fields and/or literal values `(table::field1,constant,table::field2...)`, which returns a separate result for each repetition of the calculation across each corresponding repetition of the fields. If any fields are related, only the first related record is used.

Examples

In the following examples:

- Field1 contains white.
- Field2 contains black.
- Field3 contains three repetitions with values of red, green, blue.
- Related::Field4 refers to three records that contain 100, 200, 300.
- $f1 contains orange.

Note When referencing multiple repeating fields, `List()` returns the list of values across the first repetition in the calculation's first repetition, then the list of values across the second repetition in the second repetition, and so on.

Example 1

`List (Field1; Field2)` returns:

- white
- black

Example 2

`List(Field3)` returns:

- red
- green
- blue

Example 3

`List (Field1; Field2; Field3)` returns:

in calculation repetition 1:

- white
- black
- red

in calculation repetition 2:

- green

in calculation repetition 3:

- blue

Example 4

`List(Related::Field4)` returns:

- 100
- 200
- 300

Example 5

`List ($f1; Field2)` returns:

- orange
- black

Max

Purpose

Returns the highest valid value in `field`.

Format

`Max(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field (`repeatingField`).
- a field in matching related records specified by (`table::field`), whether or not these records appear in a portal.
- several non-repeating fields in a record (`field1;field2;field3...`).
- corresponding repetitions of repeating fields in a record (`repeatingField1;repeatingField2;repeatingField3`), if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by (`table::field1;table::field2;...`). You can include fields from different tables (`table 1::field A;table 2::field B...`).

Examples

The Accounts layout has a portal showing installment payments made.

`Max(Payments::PaymentDate)` returns the most recent date a payment was made on an account.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains 6.

`Max(Field2)` returns **8** when the calculation isn't a repeating field.

`Max(Field1;Field2;Field3)` returns **6**, **6**, **7**, **8** when the calculation is a repeating field.

Notes

- When a referenced field is a repeating field, the `Max` function returns the maximum value in the first repetition field, then the maximum value in the second repetition field, and so on.
- Aggregate functions such as `Min` or `Max` use the data type of the first parameter to perform all comparisons. For example, if the first parameter's data type is text, all other parameters are converted to text and then compared.

Min

Purpose

Returns the smallest valid non-blank value in `field`.

Format

`Min(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

Examples

A Contracts table has a portal showing bids submitted for each contract.

`Min(Bids::Price)` returns the lowest bid submitted for a contract.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains 6.

`Min(Field2)` returns **5** when the calculation isn't a repeating field.

`Min(Field1;Field2;Field3)` returns **1**, **2**, **7**, **8** when the calculation is a repeating field.

Notes

- When a referenced field is a repeating field, the `Min` function returns the minimum value in the first repetition field, then the minimum value in the second repetition field, and so on.
- Aggregate functions such as `Min` or `Max` use the data type of the first parameter to perform all comparisons. For example, if the first parameter's data type is text, all other parameters are converted to text and then compared.

StDev

Purpose

Returns the standard deviation of the sample represented by a series of non-blank values in `field`.

Format

`StDev(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField).`
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table 1::field A, table 2::field B,...)`. You can name a different table for each field `(table 1::field A;table 2::field B...)`.

$$\mathbf{StDev} = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n-1} - \frac{(x_1 + x_2 + \ldots + x_n)^2}{n(n-1)}}$$

Examples

A portal displays the related values 5, 6, 7, and 8 in a field called `Scores`.
`StDev(table::Scores)` returns **1.29099444...**.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains four repetitions with values of 6, 0, 4, and 4.
- Field4 contains one repetition with a value of 3.

`StDev(Field4)` results in an error because standard deviation of a single number is not defined.

`StDev(Field1;Field2;Field3)` returns **2.64575131...**, **3.05505046...**, **2.12132034...**, **2.82842712...** for a repeating field.

Note When a referenced field is a repeating field, the `StDev` function returns the standard deviation in the first repetition fields, then the standard deviation in the second repetition fields, and so on.

StDevP

Purpose

Returns the standard deviation of a population represented by a series of non-blank values in `field`.

Format

`StDevP(field{;field... })`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

$$\textbf{StDevP} = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n} - \left(\frac{x_1 + x_2 + \ldots + x_n}{n}\right)^2}$$

Examples

A portal displays the related values 5, 6, 7, and 8 in the field `Scores`. `StDevP(table::Scores)` returns **1.11803398...**.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains four repetitions with values of 6, 0, 4, and 4.

- Field4 contains one repetition with a value of 3.

`StDevP(Field4)` results in an error because the population standard deviation of a single number is not defined.

`StDevP(Field2)` returns **1.11803398...** for a non-repeating field.

`StDevP(Field1;Field2;Field3)` returns **2.16024689...**, **2.49443825...**, **1.5**, **2** for repeating fields.

Note When a referenced field is a repeating field, the `StDevP` function returns the standard deviation of a population in the first repetition fields, then the standard deviation of a population in the second repetition fields, and so on.

Sum

Purpose

Returns the total of all valid, non-blank values in `field`.

Format

`Sum(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

Examples

An Invoice table has a portal showing line items.

`Sum(LineItems::ExtendedPrice)` totals the amounts for all items on the invoice.

A TimeBilling table has a portal showing time worked on a project. Hours is a time field.

`Sum(Hours::BillableHours)` returns the total number of billable hours on a project. Thus, if the portal shows **40** hours and **15:30** hours, the total billable hours are 55:30, or 55 1/2 hours.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains 6.

If the calculation result isn't a repeating field:

- `Sum(Field2)` returns **26**.
- `Sum(Field1;Field2;Field3)` returns **12**.

If the calculation result is a repeating field:

- `Sum(Field2)` returns a repeating field with **26** in the first repetition.
- `Sum(Field1;Field2;Field3)` returns a repeating field with **12**, **8**, **7**, **8**.

Note When a referenced field is a repeating field, the `Sum` function returns the sum of the first repetition field, then the sum of the second repetition field, and so on.

Variance

Purpose

Returns the variance of a sample represented by a series of non-blank values in `field`.

Format

`Variance(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The variance of a distribution is a measure of how spread out the distribution is. `Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

$$\textbf{Variance} = \frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n - 1} - \frac{(x_1 + x_2 + \ldots + x_n)^2}{n(n - 1)}$$

Examples

A portal displays the related values 5, 6, 7, and 8 in `Scores`.

`Variance(table::Scores)` returns **1.66666666...**.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains four repetitions with values of 6, 0, 4,and 4.

- Field4 contains one repetition with a value of 3.

`Variance(Field4)` results in an error since the variance of a single value is not defined.

`Variance(Field1;Field2;Field3)` returns **7, 9.33333333...**, **4.5**, **8** if the calculation is a repeating field.

Student example

Two classes of students take an exam. Class 1 has scores of 70, 71, 70, 74, 75, 73, 72 and Class 2 has scores of 55, 80, 75, 40, 65, 50, 95. The variance for each class is:

Class 1: **3.80952380...**

Class 2: **361.90476190...**

The variance for Class 1 is much lower than the variance for Class 2, because the scores for Class 2 are more spread out.

VarianceP

Purpose

Returns the variance of a population represented by a series of non-blank values in `field`.

Format

`VarianceP(field{;field...})`

Parameters

`field` - any related field, repeating field, or set of non-repeating fields; or an expression that returns a field, repeating field, or set of non-repeating fields.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The variance of a population distribution is a measure of how spread out the distribution is. `Field` can be any of the following:

- a repeating field `(repeatingField)`.
- a field in matching related records specified by `(table::field)`, whether or not these records appear in a portal.
- several non-repeating fields in a record `(field1;field2;field3...)`.
- corresponding repetitions of repeating fields in a record `(repeatingField1;repeatingField2;repeatingField3)`, if the result is returned in a repeating field with at least the same number of repeats.
- several fields in the first matching record specified by `(table::field1;table::field2;...)`. You can include fields from different tables `(table 1::field A;table 2::field B...)`.

$$\mathbf{VarianceP} = \frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n} - \left(\frac{x_1 + x_2 + \ldots + x_n}{n}\right)^2$$

Examples

A portal displays the related values 5, 6, 7, and 8 in Scores.

`VarianceP(table::Scores)` returns **1.25**.

In the following examples:

- Field1 contains two repetitions with values of 1 and 2.
- Field2 contains four repetitions with values of 5, 6, 7, and 8.
- Field3 contains four repetitions with values of 6, 0, 4, and 4.

- Field4 contains one repetition with a value of 3.

`VarianceP(Field4)` results in an error since the variance of a single value is not defined.

`VarianceP(Field1;Field2;Field3)` returns **4.66666666..., 6.22222222..., 2.25, 4** if the calculation is a repeating field.

Student example

Two classes of students take an exam. Class 1 has scores of 70, 71, 70, 74, 75, 73, 72 and Class 2 has scores of 55, 80, 75, 40, 65, 50, 95. The population variance for each class is:

Class 1: **3.26530612...**

Class 2: **310.20408163...**

The population variance for Class 1 is much lower than the population variance for Class 2 because the scores for Class 1 are more tightly clustered.

Container functions

Container functions calculate, manipulate, and report on data in container fields.

Click a function name for details.

This function	Returns
Base64Decode	Container content from text encoded in Base64 format.
Base64Encode	The contents of the specified container field as text in Base64 format.
GetContainerAttribute	The file metadata of a container field.
GetHeight	The height of the image in a container field that holds images.
GetThumbnail	An image stored in the container field, resized according to specified values for width and height.
GetWidth	The width of the image in a container field that holds images.
VerifyContainer	A Boolean value representing the state of container data that's stored externally. A False result means that files saved externally were changed or deleted.

Base64Decode

Purpose

Returns container content from text encoded in Base64 format.

Format

`Base64Decode(text{;fileNameWithExtension})`

Parameters

`text` - Base64 text to decode.

`fileNameWithExtension` - the filename and extension for the file created from the decoded Base64 text.

Data type returned

container

Originated in

FileMaker Pro 13.0

Description

Base64 encoding does not retain the filename or extension of encoded content.

If a filename and extension are not specified in the `fileNameWithExtension` parameter, Base64Decode returns the container content with a generic filename and extension but does not change the content’s data format.

Examples

`Base64Decode(Products::Base64;"question.png")` returns when Products::Base64 is set to a string that begins with "iVBORw0KGgoAAAANSUhEUgAAAB8". The Base64 string in this example was shortened for readability.

Base64Encode

Purpose

Returns the contents of the specified container field as text in Base64 format.

Format

`Base64Encode(sourceField)`

Parameters

`sourceField` - the name of a container field.

Data type returned

text

Originated in

FileMaker Pro 13.0

Description

Base64 encoding does not retain the filename or extension of encoded content.

Base64Encode adds a line break after every 76 characters.

Examples

`Base64Encode(Products::Container)` returns a string that begins with **iVBORw0KGgoAAAANSUhEUgAAAB8** when Products::Container is set to . The Base64 string in this example was shortened for readability.

GetContainerAttribute

Purpose

Returns the file metadata of the specified container field.

Format

`GetContainerAttribute(sourceField;attributeName)`

Parameters

`sourceField` - the name of a container field.

`attributeName` - the name of a supported attribute (see below).

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 13.0

Description

Some attributes may not return a result. For example, the values for the latitude and longitude of a photo may not be available, or some audio metadata like album art may not be available because the metadata is stored outside the audio file. Some individual attributes in the group attribute `all` may not be applicable in some circumstances.

Attributes

Attribute	Returns	Data type returned
General		
`filename`	The name of the file inserted into the container field.	text
`MD5`	The result of applying the cryptographic hash function MD5 to a file inserted into the container field or a file referenced by a container field.	text
`storageType`	The method used to store the data in the container field: **Embedded**, **External (Secure)**, **External (Open)**, **File Reference**, **Text**.	text
`fileSize`	The size (in bytes) of the file inserted into the container field.	number
`internalSize`	The amount (in bytes) of the space inside the database file that is occupied by the container field.	number
`externalSize`	The amount (in bytes) of the space that is stored externally by the container field. This is either the size of the referenced file or the total size of all files in the container field (set up for open or secure storage).	number
`externalFiles`	A list of the external files associated with the container field (either files using open or secure storage or a file reference).	text

Attribute	Returns	Data type returned
Images		
`width`	A number representing the width of the image in pixels.	number
`height`	A number representing the height of the image in pixels.	number
`dpiWidth`	A number representing the horizontal DPI of the image.	number
`dpiHeight`	A number representing the vertical DPI of the image.	number
`transparency`	**1** if the image has an alpha channel, otherwise returns **0**.	number
Photos		
`orientation`	A number representing the orientation of the photo: **1 (Normal)** **2 (Flipped horizontally)** **3 (Rotated 180 degrees)** **4 (Flipped vertically)** **5 (Rotated 90 degrees counterclockwise and flipped vertically)** **6 (Rotated 90 degrees counterclockwise)** **7 (Rotated 90 degrees clockwise and flipped vertically)** **8 (Rotated 90 degrees clockwise)** **Note:** Photos that were inserted using earlier versions of FileMaker Pro are not automatically oriented; for such photos, **not applied** is appended to the result. For example, **3 (Rotated 180 degrees), not applied**.	text
`created`	The earliest available timestamp for the photo.	timestamp
`modified`	The latest available timestamp for the photo. If the photo has never been modified, an empty string is returned.	timestamp
`latitude`	The latitude of the location of the photo.	text
`longitude`	The longitude of the location of the photo.	text
`make`	The manufacturer of the camera used for the photo.	text
`model`	The camera model used for the photo.	text
Audio		
	Note Only MP3 and M4A files return results.	
`title`	The title of the audio.	text
`artist`	The name of the performer of the audio.	text
`album`	The name of the album containing the audio.	text
`year`	The year the audio was released.	text
`track`	The track number and count of the audio. For example, **3/12**, or **3** if the track count is not available.	text
`genre`	The genre of the audio.	text
`composer`	The composer of the audio.	text
`coverArt`	An image of the album cover.	container

Attribute	Returns	Data type returned
`duration`	The duration of the audio. For example, **0:03:16**.	time
`bitRate`	The number of kilobits per second (kbps) used in the audio.	number
Bar Codes		
`barcodeText`	The content of the bar code.	text
`barcodeType`	The type of the bar code.	text
Signatures		
`signed`	The timestamp when the signature was inserted.	timestamp
Groups		
`general`	Attributes listed in the General category above pertaining to the container field.	text
`audio`	Attributes listed in the Audio category above pertaining to the container field.	text
`image`	Attributes listed in the Images category above pertaining to the container field.	text
`photo`	Attributes listed in the Photos category above pertaining to the container field.	text
`barcode`	Attributes listed in the Bar Codes category above pertaining to the container field.	text
`signature`	Attributes listed in the Signatures category above pertaining to the container field.	text
`all`	Attributes in all the categories listed above pertaining to the container field.	text

Notes

- The `internalSize` can be much smaller than the `fileSize` (for example, container fields set up for open or secure storage, file references, or compressed files) or much larger than the `fileSize` (for example, container fields created by plug-ins).
- Using the attribute `MD5` allows you to prevent the insertion of duplicated files into a container field regardless of the filename.
- Bar codes and signatures are not considered images.
- For the group attributes `general`, `audio`, `image`, `photo`, and `all`, attributes are displayed in the format `attributeName: attributeValue` with one attribute per line. Some attributes are displayed differently in order to fit the attribute on one line:
  - `externalFiles`. Displays only the number of external files.
  - `transparency`. Displays **1 (True)** or **0 (False)**.
  - `coverArt`. Displays **png** or **jpg** depending on the type of image.
  - `bitRate`. Displays **kbps** after the number. If an audio uses a variable bit rate, **(VBR)** is appended to the result. For example: **Bit Rate: 247 kbps (VBR)**.

- `year`. Date information may be returned in parentheses after the year. For example: **Year: 2014 (11/10/2014)**.

Examples

Notice that the attributes in the following examples are enclosed in quotation marks.

`GetContainerAttribute(Image;"all")` returns:

[General]

Filename: IMG_003.JPG

Storage Type: Embedded

MD5: C35A3F668A1FB3F370969399A1FF04FE

File Size: 1964978

Internal Size: 1965064

External Size: 0

External Files: 0

[Image]

Width: 1936

Height: 2592

DPI Width: 72

DPI Height: 72

Transparency: 0 (False)

[Photo]

Orientation: 6 (Rotated 90 degrees counterclockwise)

Created: 11/14/2014 2:40:31 PM

Modified:

Latitude: 37.406167

Longitude: -121.983333

Make: Apple

Model: iPhone 4

`GetContainerAttribute(Product;"barcode")` returns:

[Bar Code]

Bar Code Text: 875720001107

Bar Code Type: UPC-A

`GetContainerAttribute(Package;"signature")` returns:

[Signature]

Signed: 11/10/2014 11:41:22 AM

GetHeight

Purpose

Returns the height in pixels of the content in a container field that holds images.

Format

`GetHeight(field)`

Parameters

`field` - any text, number, date, time, timestamp, or container field; or any text expression or numeric expression.

Data type returned

number

Originated in

FileMaker Pro 12.0

Description

Returns the height in pixels of images in a container field that holds images. Otherwise, `GetHeight` returns 0.

Examples

`GetHeight(product)` returns **768**.

GetThumbnail

Purpose

Returns a thumbnail image of the content in a container field, according to specified values for width and height.

Format

`GetThumbnail(field;width;height)`

Parameters

`field` - any text, number, date, time, timestamp, or container field; or any text expression or numeric expression.

`width` - the width for the thumbnail.

`height` - the height for the thumbnail.

Data type returned

container

Originated in

FileMaker Pro 12.0

Description

Returns an image that's stored in a container field according to specified values for width and height. The thumbnail image always maintains the proportions of the original image.

Note If the `field` parameter does not specify a field that contains image data, `field` must evaluate to the file path of an image. See Creating file paths.

Examples

`GetThumbnail(Dog;GetLayoutObjectAttribute("rectangle","width");`
`GetLayoutObjectAttribute("rectangle","height"))` returns an image stored in the Dog field that fits into the dimensions of the Rectangle layout object.

`GetThumbnail(Property;GetWidth(Property)/2;GetHeight(Property)/2)`returns an image that is 50 percent of the size of the original image in the Property field.

`GetThumbnail ( "image:question.png" ; 50 ; 50 )` returns a thumbnail of question.png with a maximum height and width of 50 points.

GetWidth

Purpose

Returns the width in pixels of the content in a container field that holds images.

Format

`GetWidth(field)`

Parameters

`field` - any text, number, date, time, timestamp, or container field; or any text expression or numeric expression.

Data type returned

number

Originated in

FileMaker Pro 12.0

Description

Returns the width in pixels of images in a container field that holds images. Otherwise, `GetWidth` returns 0.

Examples

`GetWidth(Product)` returns **1024**.

VerifyContainer

Purpose

Returns a Boolean value representing the validity of data stored externally in a container field.

Format

`VerifyContainer(field)`

Parameters

`field` - any text, number, date, time, timestamp, or container field; or any text expression or numeric expression.

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

Returns a Boolean value representing the validity of data stored externally in a container field. A 0 (False) value means the data was changed or deleted; otherwise, 1 (True) is returned.

Examples

`VerifyContainer(Photo)` returns:

- **0 (False)** if files saved externally were modified or deleted.
- **1 (True)** if no changes or deletions occurred.
- **?** if the Photo field is not a container field.

Date functions

Date functions calculate dates and manipulate date information.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Note System formats affect the way dates are displayed. See Opening files with different system formats.

Tip You can use zero (0) and negative numbers as Date function arguments. For example, the following formula returns **5/31/2014**:

```
Date(6;0;2014)
```

Click a function name for details.

This function	Returns
Date	The calendar date for the specified month, day, and year.
Day	A number in the range 1 through 31, representing the day of the month for a specified date.
DayName	A text string that is the full name of the weekday for the specified date.
DayNameJ	A text string that is the full name of the weekday for the specified date in Japanese.
DayOfWeek	A number representing the day of the week the specified date falls on.
DayOfYear	A number equal to the number of days from the beginning of the year of the specified date.
Month	A number in the range 1 through 12, representing the number of the month of the year in which the specified date occurs.
MonthName	The name of the month for the specified date.
MonthNameJ	The name of the month in Japanese for the specified date.
WeekOfYear	The number of weeks after January 1 of the year of the specified date.
WeekOfYearFiscal	A number between 1 and 53 representing the week containing a specified date, figured according to the specified starting day.
Year	A number representing the year in which the specified date occurs.
YearName	The Japanese year name of the specified date, provided in the specified format.

Date

Purpose

Returns the calendar date for `month`, `day`, and `year`.

Format

`Date(month;day;year)`

Parameters

`month` - the month of the year (a one-digit or two-digit number; see note).

`day` - the day of the month (a one-digit or two-digit number; see note).

`year` - the year (four digits between 0001 and 4000. For example, 2014 but not 14).

Important The order of the parameters in the `Date` function is always Month, Day, Year, no matter what operating system or FileMaker Pro date formats you are using.

Data type returned

date

Originated in

FileMaker Pro 6.0 or earlier

Description

The format of the result depends on the date format that was in use when the database file was created. In the United States, dates are generally in the format MM/DD/YYYY. You can change the date format in your operating system.

You can change how the date is displayed by assigning a different date format to the field in Layout mode. Changing the formatting in this way only affects the way the data is displayed, not how it is stored.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Note If you type a month greater than 12 or a day greater than the number of days in a month, FileMaker Pro adds the extra days or months to the result. The date function also allows zero and negative numbers as parameters. Decimal numbers are truncated to integers.

Examples

`Date(10;10;2014)` returns **10/10/2014**.

`Date(13;1;2014)` returns **1/1/2015** (one month after December 1, 2014).

`Date(6;0;2014)` returns **5/31/2014** (one day before June 1, 2014).

`Date(6;-2;2014)` returns **5/29/2014** (three days before June 1, 2014).

`Date(7;12;2014)-Date(7;2;2014)` returns **10**.

```
“Bill Due by: ” & Date(Month(DateSold) + 1;Day(DateSold);Year(DateSold))
```
returns **Bill Due by:** followed by a value that is one month later than DateSold.

Day

Purpose

Returns a number in the range 1 through 31, representing the day of the month on which `date` occurs.

Format

`Day(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use `Day`, for example, to identify the day of the month on which payments are due.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`Day(“5/15/2014”)` returns **15**. This example assumes that the system date format is MM/DD/YYYY.

`Day(DateSold)` returns the day of the month stored in DateSold.

`If(Day(Get(CurrentDate))= 15 and Month(Get(CurrentDate))=3;“Beware the Ides of March”;“”)` displays the text **Beware the Ides of March** only when the day of the month returned by `Get(CurrentDate)` is 15 and the month returned by `Get(CurrentDate)` is 3; otherwise it displays nothing.

DayName

Purpose

Returns a text string that is the full name of the weekday for `date`.

Format

`DayName(date)`

Parameters

`date` - any calendar date

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`DayName(Date(10;7;2014))` returns **Tuesday**.

`DayName(ProjectDue)` returns **Tuesday** when ProjectDue is 10/7/2014.

`DayName("10/7/2014")` returns **Tuesday**.

`"Return your selection by " & DayName(DueDate)` displays the text **Return your selection by** followed by the name of the day stored in DueDate.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

DayNameJ

Purpose

Returns a text string in Japanese that is the full name of the weekday for `date`.

Format

`DayNameJ(date)`

Parameters

`date` - any calendar date

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`DayNameJ(Date(4;4;2014))` returns 金曜日 .

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

DayOfWeek

Purpose

Returns a number representing the day of the week that `date` falls on.

Format

`DayOfWeek(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The number 1 represents Sunday, 2 represents Monday, 3 represents Tuesday, and so on. For example, you can find out on what day of the week a holiday occurs.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`DayOfWeek(“10/8/2014”)` returns **4**.

`DayOfWeek(Date(10;9;2014))` returns **5**.

`DayOfWeek(ProjectDue)` returns **3** when the date in ProjectDue is 10/7/2014.

DayOfYear

Purpose

Returns a number equal to the number of days from the beginning of the year of `date`.

Format

`DayOfYear(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`DayOfYear(Billing Date)` returns **32**, when Billing Date is 2/1/2014.

The following formulas return the total number of days in the current year:

```
DayOfYear(Date(12;31;Year(Get(CurrentDate))))
DayOfYear(Date(1;1;Year(Get(CurrentDate)) + 1) -1)
```

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Month

Purpose

Returns a number in the range 1 through 12, representing the number of the month of the year in which `date` occurs.

Format

`Month(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Month(“3/19/2014”)` returns **3**. This example assumes that the operating system date format is set to MM/DD/YYYY.

`Month(Payment)` returns **3**, where Payment contains March 19, 2014. (The Payment field must be of type date.)

`“Bill Due by: ” & Date(Month(DateSold) + 1;Day(DateSold);Year(DateSold))` returns **Bill Due by:** followed by a value that is one month later than DateSold.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

MonthName

Purpose

Returns the full name of the month for `date`.

Format

`MonthName(date)`

Parameters

`date` - any calendar date

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`MonthName(“6/6/2014”)` returns **June**.

`“Payment due by the end of: ” & MonthName(Date(Month(InvoiceDate) + 1;Day(InvoiceDate);Year(InvoiceDate)))` returns **Payment due by the end of May**, where InvoiceDate is 4/4/2014.

`“Payment for: ” & MonthName(Date(Month(Payment) + 1;Day(Payment);Year(Payment)))` returns **Payment for:** followed by the name of the month that is one past the month of the last payment.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

MonthNameJ

Purpose

Returns the name of the month of `date` in Japanese.

Format

`MonthNameJ(date)`

Parameters

`date` - any calendar date

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`MonthNameJ(“6/6/2014”)` returns 6月

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

WeekOfYear

Purpose

Returns the number of weeks after January 1 of the year of `date`.

Format

`WeekOfYear(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Fractions of weeks occurring at the beginning or end of the year count as full weeks, so the `WeekOfYear` function returns values 1 through 54.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`WeekOfYear(“1/1/2014”)` returns **1**.

`WeekOfYear(ProjectDue)` returns **6**, when ProjectDue is 2/2/2014.

`WeekOfYear(“1/1/2014”) - WeekOfYear(“2/2/2014”)` returns **-5**.

WeekOfYearFiscal

Purpose

Returns a number between 1 and 53 representing the week containing `date`, figured according to `startingDay`.

Format

`WeekOfYearFiscal(date;startingDay)`

Parameters

`date` - any calendar date

`startingDay` - any number between 1 and 7, where 1 represents Sunday

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`startingDay` indicates which day is considered the first day of the week.

The first week of the year is the first week that contains four or more days of that year. For example, if you select 1 (Sunday) as the starting day, then January 1 must be on Sunday, Monday, Tuesday, or Wednesday for that week to be the first week of the fiscal year. If you select 2 (Monday) as the starting day, then January 1 must be on Monday, Tuesday, Wednesday, or Thursday for that week to be the first week of the fiscal year.

It is possible, using this function, that dates in a particular year will be returned as the 53rd week of the previous year. For example, if in 2008 you selected Sunday (1) as the starting date, then January 1, 2, or 3 in 2009 would occur in week 53 of fiscal year 2008 (in 2009, January 1 is on a Thursday). The first day of fiscal year 2009 would be on Sunday, January 4, because you selected Sunday (1) as the starting day.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`WeekOfYearFiscal(Date(1;7;2008);1)` returns **2**.

`WeekOfYearFiscal(Date(1;1;2009);5)` returns **1**.

`WeekOfYearFiscal(Date(1;2;2009);1)` returns **53**.

Year

Purpose

Returns a number representing the year in which `date` occurs.

Format

`Year(date)`

Parameters

`date` - any calendar date

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

You can, for example, extract the year from a field containing the date an item was sold.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`Year(DateSold)` returns the year stored in DateSold.

`Year(“5/5/2014”)` returns **2014**.

`Year(Date(Month(Get(CurrentDate)) + 48;Day(Get(CurrentDate));Year(Get(CurrentDate))))` returns the year that is 48 months from today’s date.

YearName

Purpose

Returns the Japanese year name of `date`, provided in the specified format.

Format

`YearName(date;format)`

Parameters

`date` - any calendar date

`format` - a number (0, 1, or 2) that describes the display format

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If the value for `format` is blank or other than 0, 1, or 2, then 0 is used.

0 - 明治 8 (Meiji 8), 大正 8 (Taisho 8), 昭和 8 (Showa 8), 平成 8 (Heisei 8), 西暦xxxx (Seireki xxxx [before 1868.9.8])

1 - 明 8 (Mei 8), 大 8 (Tai 8), 昭 8 (Sho 8), 平 8 (Hei 8), (西) xxxx (Sei xxxx [before 1868.9.8])

2 - M8, T8, S8, H8, A.D.xxxx (before 1868.9.8)

Name of Emperor in 0 = Long, 1 = Abbreviated, 2 = 2 byte Roman. Seireki is returned when date is before listed emperors.

Examples

`YearName(DateField;0)` Returns 平成20 when DateField contains 7/15/2008.

Design functions

Design functions return information about the structure of open database files. For example, you could determine the names of all the layouts or fields in an open database file.

Note FileMaker Pro limits the information returned by a design function, according to the privilege set in effect when the function evaluates a database file. See Creating and managing privilege sets for more information about granting access to database files.

Design function parameters can be any of the following:

- filenames such as "Customer" or literal text such as "Jack"
- layouts such as `layoutName`
- other functions such as `Left(text;number)`

Important Literal text parameters such as filenames and layout names must be enclosed in quotation marks. Use quotation marks around field names to indicate the literal string is the parameter (omit quotation marks to indicate the value stored in the field is the parameter). You can use spaces before or after the parentheses that enclose parameters, but spaces are not necessary. Use a semicolon between parameters when a function requires more than one parameter.

Click a function name for details.

This function	Returns
DatabaseNames	A list of the names of all database files open on the computer (including files opened as a client), separated by carriage returns.
FieldBounds	The location of each side of the specified field and its rotation in degrees.
FieldComment	The specified field's comment.
FieldIDs	A list of all field IDs in the specified database file and layout, separated by carriage returns.
FieldNames	A list of the names of all fields on the specified layout, separated by carriage returns.
FieldRepetitions	The number of repetitions of the specified field as it is formatted on the specified layout (which could be different from the number of repetitions specified when the field was defined), and the orientation of the field repetitions (horizontal or vertical) on the layout.
FieldStyle	The formatting applied to the specified field on the specified layout.
FieldType	Information about the specified field.
GetNextSerialValue	The next serial number for the specified field in the specified database file.
LayoutIDs	A list of all layout IDs in the specified database file, separated by carriage returns.
LayoutNames	A list of the names of all layouts in the specified database file, separated by carriage returns.
LayoutObjectNames	A list of the names of all named layout objects, separated by carriage returns.
RelationInfo	A list of four values for each relationship directly related to the specified table.
ScriptIDs	A list of all script IDs in the specified database file, separated by carriage returns.
ScriptNames	A list of the names of all scripts in the specified database file, separated by carriage returns.
TableIDs	A list of all table IDs in the specified database file, separated by carriage returns.

This function	Returns
TableNames	A list of the names of all defined tables in the specified database file, separated by carriage returns.
ValueListIDs	A list of all value list IDs in the specified database file, separated by carriage returns.
ValueListItems	A list of the values in the specified value list, separated by carriage returns.
ValueListNames	A list of the names of all value lists in the specified database file, separated by carriage returns.
WindowNames	A list of the names of open windows in the specified database file.

DatabaseNames

Purpose

Returns a list of the names of all database files open on the computer, separated by carriage returns.

Format

`DatabaseNames`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

The names returned do not include file extensions.

Note If your database is hosted on another computer, `DatabaseNames` returns a list of the names of local client and remote database files open only on the client computer.

Examples

To determine whether Customers is one of the files currently open, use the `DatabaseNames` function with the `FilterValues` function in the formula:

```
FilterValues( DatabaseNames;"Customers")
```

If the formula returns any text value, then Customers is open.

If you want to know how many files with the same name are open, use the `DatabaseNames` function with the `PatternCount` function in the formula:

```
PatternCount(FilterValues(DatabaseNames;"Customers");"Customers")
```

This will tell you how many files named Customers are open.

FieldBounds

Purpose

Returns the location in points of each field boundary and the field’s rotation in degrees.

Format

`FieldBounds(fileName;layoutName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout in the specified database file.

`fieldName` - the name of a field on the specified layout.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

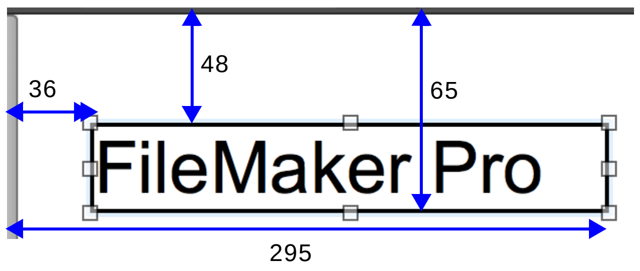
Description

The location returned is measured from the top left corner of the layout (regardless of printer margins) and is specified in this order: position of left field boundary, position of top field boundary, position of right field boundary, position of bottom field boundary, degree of rotation (measured in a counter-clockwise direction; 0 degrees for unrotated).

Note Your layout begins where your margins end. Because field boundaries are measured from the left side and top of the layout, boundaries returned by `FieldBounds` never change unless you move or re-size a field.

Examples

`FieldBounds(“Customers”;“Layout #1”;“Field”)` returns **36 48 295 65 0** in the example below. Notice that all parameters are enclosed in quotation marks.



FieldComment

Purpose

Returns the specified field's comment.

Format

`FieldComment(fileName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`fieldName` - the name of a field in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

The field name must be in the form `tablename::fieldname` to specify a field that exists in a table different from the current table.

Examples

`FieldComment("Customers"; "Phone Number")` returns **"Customer's home telephone number"** if it was entered as a comment for the Phone Number field.

`FieldComment("Customers"; "Accounts::Current Balance")` returns **"Customer's current balance**" if it was entered as a comment for the Current Balance field in the Accounts table.

FieldIDs

Purpose

Returns a list of all field IDs in `fileName` and `layoutName`, separated by carriage returns.

Format

`FieldIDs(fileName;layoutName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout or table in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Related fields are returned as **TableID::RelatedFieldID**.

For example, **12::4**, where **12** is the ID of the table and **4** is the ID of the related field.

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`FieldIDs(“Customers”;””)` returns IDs of all unique fields in the default table of Customers.

`FieldIDs(“Customers”;”Layout#5”)` returns IDs of all unique fields, including related fields, on Layout#5 in Customers.

FieldNames

Purpose

Returns a list of the names of all fields on `layoutName`, in `fileName` file, separated by carriage returns.

Format

`FieldNames(fileName;layoutName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout or table in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Related fields are displayed in `tablename::fieldname` format.

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Note If `FieldNames` returns a question mark (?) or the name of only one field, go to the Specify Calculation dialog box and make sure **Calculation result is** text. Also, you can increase the size of the field on the layout to show more field names.

Examples

`FieldNames(“Customers”;””)` returns a list of all the fields in the default table of the Customers database file.

`FieldNames(“Customers”;“Data Entry”)` returns a list of all the fields, including related fields, in the Customers database file that appear on the Data Entry layout.

FieldRepetitions

Purpose

Returns the number of repetitions of the repeating field `fieldName` as it is currently formatted on `layoutName` (which could be different from the number of repetitions when the field was defined), and the orientation of the field repetitions (horizontal or vertical) on the layout.

Format

`FieldRepetitions(fileName;layoutName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout in the specified database file.

`fieldName` - the name of a field on the specified layout.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If `fieldName` isn't a repeating field, it returns **1 vertical**.

Examples

`FieldRepetitions("Customers";"Data Entry";"Business Phone")` returns **3 vertical** if the Business Phone field is defined as a repeating field with five repetitions but is formatted to only show three repetitions in a vertical orientation on the Data Entry layout.

FieldStyle

Purpose

Returns the field formatting applied to `fieldName` on `layoutName` in the `fileName` file.

Format

`FieldStyle(fileName;layoutName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout in the specified database file.

`fieldName` - the name of a field on the specified layout.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If the field has a value list associated with it, the `FieldStyle` function also returns the name of the value list.

- A standard field returns **Standard.**
- A standard field with a vertical scroll bar returns **Scrolling.**
- A drop-down list returns **Popuplist.**
- A pop-up menu returns **Popupmenu.**
- A checkbox returns **Checkbox.**
- A radio button returns **RadioButton.**
- A drop-down calendar returns **Calendar**.

Examples

On the Data Entry layout in the Customers database file, `FieldStyle(“Customers”;“Data Entry”;“Current Customer”)` returns **RadioButton Yes/No List** when the Current Customer field is formatted as a radio button and is associated with the value list named Yes/No List.

FieldType

Purpose

Returns information about `fieldName`.

Format

`FieldType(fileName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`fieldName` - the name of a field in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Field names must be in the format `tablename::fieldname` to specify a field that exists in a table different from the current table. The result has four values separated by spaces:

- The first value is either Standard, StoredCalc, Summary, UnstoredCalc, External(Secure), External(Open), or Global.
- The second value is the field type: text, number, date, time, timestamp, or container.
- The third value is Indexed or Unindexed.
- The fourth value is the maximum number of repetitions defined for the field (if the field isn't defined as a repeating field, this value is 1).

Examples

`FieldType("Customers";"Phone Number")` returns **Standard Text Unindexed 3** when, in the Customers database file, the Phone Number field is defined as a text field that repeats a maximum of three times and the storage options are left unchanged. (Most fields are indexed when a find is performed in that field.)

`FieldType("Customers";"Current Balance")` returns **StoredCalc Number Indexed 1** when, in the Customers database file, the Current Balance field is defined as a stored, numeric calculation field that is indexed.

`FieldType("Customers";"Today's Date")` returns **Global Date Unindexed 1** when, in the Customers database file, the Today's Date field is defined as a global field of type date. Global fields are never indexed.

`FieldType("Customers";"Statement")` returns **External(Secure) Container Unindexed 1** when, in the Customers database file, the Statement field is defined as a container field that stores data externally using secure storage. Container fields cannot be indexed.

GetNextSerialValue

Purpose

Returns the next serial number of `fieldName` in `fileName`.

Format

`GetNextSerialValue(fileName;fieldName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`fieldName` - the name of the field whose next serial number you want to determine.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Field names must be fully qualified in the format `tablename::fieldname` to specify a field that exists in a table different from the current table.

Examples

`GetNextSerialValue(“Customers”;”CustID”)` returns the next serial number for the CustID field.

LayoutIDs

Purpose

Returns a list of all layout IDs in `fileName`, separated by carriage returns.

Format

`LayoutIDs(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`LayoutIDs(“Customers”)` returns a list of all the layout IDs in the Customers database file.

LayoutNames

Purpose

Returns a list of the names of all layouts in `fileName`, separated by carriage returns.

Format

`LayoutNames(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`LayoutNames(“Customers”)` returns a list of all the layouts in the Customers database file.

LayoutObjectNames

Purpose

Returns a list of the names of all named objects on `layoutName` in `fileName`, separated by carriage returns.

Format

`LayoutObjectNames(fileName;layoutName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`layoutName` - the name of a layout in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 8.5

Description

Layout objects without object names are not returned. If `layoutName` isn't specified, then no object names are returned.

Named tab controls, grouped objects, and portal objects that contain other named objects are followed by a list of those named objects enclosed in angle brackets (<>). The angle brackets are shown even if there are no named objects contained within the named tab controls, grouped objects, or portal objects.

Examples

`LayoutObjectNames ("Customers";"Data Entry")` returns a list of named objects in the Customers database file that appear on the Data Entry layout.

RelationInfo

Purpose

Returns a list of four values for each relationship directly related to `tableName`.

Format

`RelationInfo(fileName;tableName)`

Parameters

`fileName` - the name of an open database file (local or remote).

`tableName` - the name of a table in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Values in a list are separated by carriage returns, and lists are separated by two carriage returns. For each additional relationship connected to `tableName`, an additional list of four values is output.

The four values are:

- `Source:` Data Source Name of the database table connected to `tableName`.
- `Table:` the name of the table connected to `tableName`.
- `Options:` the options that were set in the right side of the Edit Relationship dialog box when the relationship was defined. This line is blank if the following options are not set; otherwise these options are separated by spaces.
  - Delete, if **Delete related records in this table when a record is deleted in the other table** is selected in the right side of the Edit Relationship dialog box.
  - Create, if **Allow creation of records in this table via this relationship** is selected in the right side of the Edit Relationship dialog box.
  - Sorted, if **Sort records** is selected in the right side of the Edit Relationship dialog box.
- `Relationships:` a list of the defined relationships, one per line. Field names are fully qualified, for example, `TableName::Field Name`.

Examples

A database file called Human Resources has three tables: Company, Employees, and Addresses. `Company::Company ID` is connected to `Employees::Company ID`, `Employees::Employee ID` is connected to `Addresses::Employee ID and Employees::DateOfHire` is connected to `Addresses::DateMovedIn`.

The relationships have the following criteria:

- You can create records in all tables.
- You cannot delete records in all tables.
- A sort was specified for the Addresses table for the Employees<-->Addresses relationship.

`RelationInfo(“Human Resources”;“Employees”)` returns:

Source: Human Resources

Table: Company

Options: Create

Company::Company ID = Employees::Company ID

Source: Human Resources

Table: Addresses

Options: Create Sorted

Addresses::Employee ID = Employees::Employee ID

Addresses::DateMovedIn >= Employees::DateOfHire

ScriptIDs

Purpose

Returns a list of all script IDs in `fileName`, separated by carriage returns.

Format

`ScriptIDs(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`ScriptIDs(“Customers”)` returns a list of all the script IDs in the Customers database file.

ScriptNames

Purpose

Returns a list of the names of all scripts in `fileName`, separated by carriage returns.

Format

`ScriptNames(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`ScriptNames("Customers")` returns a list of all the scripts in the Customers database file.

TableIDs

Purpose

Returns a list of all table IDs in `fileName`, separated by carriage returns.

Format

`TableIDs(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Each table ID is unique. Also, the ID is independent of when you create each table: the first table could have the smallest, middle, or largest value.

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`TableIDs("University Database")` returns

1065089

1065090

for the University Database database file if two tables have been defined for the file.

TableNames

Purpose

Returns a list of all table occurrences in the relationships graph for `fileName`, separated by carriage returns.

Format

`TableNames(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`TableNames(“University Database”)` returns table occurrences

Teachers

Coaches

for the University Database database file if a Teachers table and a Coaches table have been defined for the file.

ValueListIDs

Purpose

Returns a list of all value list IDs in `fileName`, separated by carriage returns.

Format

`ValueListIDs(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`ValueListIDs("Customers")` returns a list of all the value list IDs in the Customers database file.

ValueListItems

Purpose

Returns a list of the values in `valuelist`, separated by carriage returns.

Format

`ValueListItems(fileName;valuelist)`

Parameters

`fileName` - the name of an open database file (local or remote).

`valuelist` - the name of a value list in the specified database file.

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`ValueListItems(“Customers”;“Code”)` returns a list of all the items in the Code value list in the Customers database file.

ValueListNames

Purpose

Returns a list of the names of all value lists in `fileName`, separated by carriage returns.

Format

`ValueListNames(fileName)`

Parameters

`fileName` - the name of an open database file (local or remote).

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If no parameter is specified for `fileName`, FileMaker returns results for the current file.

Examples

`ValueListNames(“Customers”)` returns a list of all the value list names in the Customers database file.

WindowNames

Purpose

Returns a list of the names of windows that are currently open.

Format

`WindowNames{(fileName)}`

Parameters

`{fileName}` - the name of an open database file (local or remote).

Parameters in curly braces { } are optional.

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the optional `fileName` parameter to only return windows that are based on the specified file. The window could be visible, hidden, or minimized. The order of the names in the list matches the current stacking order of the windows. The visible windows are listed first, then the minimized windows, and then the hidden windows. If there are no databases or windows open, an empty string is returned.

Note Even if you close a file, it may remain open as a hidden file if the window of any other file is displaying data from that file. (For example, another window may be displaying related data from the file you attempted to close.) FileMaker Pro will close the file when you close all the dependent windows.

Examples

`WindowNames` returns **Customers** and **Invoices** separated by a carriage return when those windows are currently open.

`WindowNames(“contacts”)` returns a list of windows that are based on the contacts database file.

External functions

Use external functions to access FileMaker Pro plug-ins. Plug-ins add features to FileMaker Pro. For more information, see Setting plug-in preferences.

External functions are only available if FileMaker Pro plug-ins are installed and enabled on your computer. If no FileMaker Pro plug-ins are installed, you see only the generic external function definition in the Specify Calculation dialog box:

External (nameOfFunction; parameter)

Plug-ins written for version 7.0 and later

Each plug-in defines its own functions and parameters. See the documentation that came with the plug-in for each function's usage.

Plug-ins written for version 6.0 and earlier

These plug-ins are still supported and continue to use the External function to access the plug-in's functions. The first parameter is the name of the plug-in function to execute and the second is a parameter that is passed to that function. See the documentation that came with the plug-in for each function's usage.

This function	Does this
External	Enables access to FileMaker Pro plug-ins written for versions of FileMaker Pro prior to 7.0.

For more information, see Updating plug-ins.

External

Purpose

Accesses plug-ins created for versions of FileMaker Pro prior to 7.0 and uses the syntax `External(“function name”, parameter)`, where `function name` is in quotes and is the name of an external function.

Format

`External(nameOfFunction;parameter)`

Parameters

`nameOfFunction` - the name of the external function

`parameter` - the parameter(s) required by the external function. A parameter is required, even if it’s only 0.

Data type returned

Depends on the external function

Originated in

FileMaker Pro 6.0 or earlier

Description

Plug-ins created for FileMaker Pro version 7.0 and later do not use the `External(“function name”, parameter)` syntax. For more information, see External functions.

Financial functions

Financial functions calculate financial information, such as net present value and payments. For example, you can calculate the monthly payments required to buy a car at a certain loan rate using the `PMT` function.

Click a function name for details.

This function	Returns
FV	The future value of an initial investment, based on a constant interest rate and payment amount for the number of periods in months.
NPV	The net present value of a series of unequal payments made at regular intervals, assuming a fixed rate per interval.
PMT	The payment required by the term, interest rate, and principal.
PV	The present value of a series of equal payments made at regular intervals (periods), assuming a fixed interest rate per interval.

FV

Purpose

Returns the future value (FV) of an initial investment, based on a constant `interestRate` and `payment` amount for the number of `periods` in months.

Format

`FV(payment;interestRate;periods)`

Parameters

`payment` - payment to be made per period

`interestRate` - interest rate per period

`periods` - number of periods

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate FV. For example, you can calculate how much you'll earn on an investment in which you pay $50 a month for 60 months at a 6 percent annual interest rate.

Notes

- When `interestRate` is 0, this function returns the result of `payment * periods`.
- The FV function doesn't account for the present value of your investment, and it assumes that payment is made at the end of each period.

$$\mathbf{FV} = \mathbf{payment} * \frac{(\mathbf{1} + \mathbf{interestRate})^{\mathbf{periods}} - \mathbf{1}}{\mathbf{interestRate}}$$

Examples

`FV(50;.11/12;5 * 12)` returns **3975.90398429...**.

`FV(2000;.12;30) + 5000 * (.12 + 1) ^ 30` returns **632464.97928640...**.

`FV(500;.11/5;60)` returns **61141.65130790...**.

To set the decimal precision of the returned value, enclose the current formulas with the `Round` function. For example, `Round(Current Formula;2)`.

NPV

Purpose

Returns the net present value (NPV) of a series of unequal `payments` made at regular intervals, assuming a fixed `interestRate` per interval.

Format

`NPV(payment;interestRate)`

Parameters

`payment` - a repeating field containing unequal payment amounts, or an expression that returns a reference to one.

`interestRate` - interest rate.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate NPV. For example, if someone borrows money from you and pays you back in unequal amounts over a period of several years, you can use the NPV function to calculate the result.

$$\textbf{NPV} = \frac{\textbf{loan amount}}{\textbf{1} + \textbf{interestRate}} + \frac{\textbf{first payment}}{(\textbf{1} + \textbf{interestRate})^{\textbf{2}}} + \frac{\textbf{second payment}}{(\textbf{1} + \textbf{interestRate})^{\textbf{3}}} + \ldots + \frac{\textbf{n}^{\textbf{th}}\textbf{ payment}}{(\textbf{1} + \textbf{interestRate})^{\textbf{n}+\textbf{1}}}$$

Examples

`NPV(Loan;.05)` returns **156.91277445...**, when the repeating field, Loan, contains -2000 (the initial payment), 600, 300, 500, 700, and 400. The result (156.91277445...) is the actual profit in today's dollars that will be realized from this transaction.

`NPV(Amounts;.10)` returns **16758.35604870...**, when the repeating field, Amounts, contains -5000 (the initial investment), 10,000, 0, 10,000, and 10,000.

If you want each return value to return 2 decimal places, surround the current formulas with the correct `Round` function: `Round(Current Formula;2)`.

PMT

Purpose

Returns the payment (`PMT`) required by the `term`, `interestRate`, and `principal`.

Format

```
PMT(principal;interestRate;term)
```

Parameters

`principal` - principal amount.

`interestRate` - interest rate. If the interest rate is annual, divide the rate by 12.

`term` - length of time, expressed in number of months.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate `PMT`.

$$\mathbf{PMT} = \mathbf{payment} / \left(\frac{\mathbf{1} - (\mathbf{1} + \mathbf{interestRate})^{-\mathbf{periods}}}{\mathbf{interestRate}} \right)$$

Examples

In the following example, the `PMT` function calculates payments for purchasing a sports car costing $21,000, at an annual rate of 6.9% over 48 monthly payments.

`PMT(21000;.069/12;48)` returns the payment amount **$501.90**.

`PMT(Cost;.13;Years)` returns a payment amount, based on the purchase value stored in Cost, at a 13 percent rate, over the duration stored in Years.

`“Your payment will be ” & PMT(150000;.13/12;Months) & “.”` returns **Your payment will be**, followed by the payment amount, based on a total cost of $150,000, at a 13 percent annual percentage rate, over the duration stored in Months.

PV

Purpose

Returns the present value (PV) of a series of equal `payments` made at regular intervals (`periods`), assuming a fixed `interestRate` per interval.

Format

`PV(payment;interestRate;periods)`

Parameters

`payment` - payment amount to be made per period. Type a negative number for money you pay and a positive number for money you receive.

`interestRate` - interest rate per period.

`periods` - number of periods (intervals between payments).

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate PV.

$$\mathbf{PV} = \mathbf{payment} * \frac{\mathbf{1} - (\mathbf{1} + \mathbf{interestRate})^{-\mathbf{periods}}}{\mathbf{interestRate}}$$

Note When `interestRate` is 0, this function returns the result of `payment * periods`.

Examples

Your cousin borrowed $2,000 from you, offering to pay you back $500 a year for five years, for a total of $2,500 at the end of five years. If inflation was 5 percent annually, with the following entry you could find out what those payments are worth with the PV function.

`PV(500;.05;5)` returns **2164.73833531...**.

If you want the return value to return two decimal places, enclose the formula with the correct `Round` function: `Round(Current Formula;2)`.

Get functions

Use Get functions in scripts for error checking and prevention, or to capture information about the status of a database file or elements in it, or an action being performed.

Many Get functions return information that changes on a regular basis. For example, when the Get(CurrentTime) function is placed in a stored calculation field, the time will only update when a new record is created. If the calculation has other fields in it, but the calculation result still returns the current time, then the stored calculation result will only update when those other fields have been modified in the current record. If either of these calculations are unstored, the time will update as needed. For performance reasons, making a calculation field unstored is not always the best idea. Get functions are best used in a script where the status information from a Get function is up-to-date at the moment that the calculation is run.

To access the list of Get functions, in the Specify Calculation dialog box, choose **View all functions by type** or **View Get functions**. When you choose **View all functions by name**, you see only Get(flag).

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Click a function name for details.

This function	Returns
Get(AccountExtendedPrivileges)	A list of keywords for the enabled extended privileges, separated by carriage returns. The list that is returned is based on the account used to open the database file.
Get(AccountName)	The authenticated account name being used for the active database file.
Get(AccountPrivilegeSetName)	The name of the privilege set that is being used by the account used to open the database file.
Get(ActiveFieldContents)	The contents of the field that has the focus.
Get(ActiveFieldName)	The name of the field that has the focus.
Get(ActiveFieldTableName)	The name of the table that contains the active field (the field that has the focus).
Get(ActiveLayoutObjectName)	The name of the active layout object in the calculation's active window.
Get(ActiveModifierKeys)	A number representing the keyboard modifier keys (for example, Shift) that are being pressed.
Get(ActivePortalRowNumber)	The number of the portal row containing the focus.
Get(ActiveRepetitionNumber)	A number representing the active repetition of a repeating field (the repetition that has the focus).
Get(ActiveSelectionSize)	A number representing how many characters are selected.
Get(ActiveSelectionStart)	A number representing the starting character of the selected text.
Get(AllowAbortState)	A Boolean value representing the current state of `Allow user abort` script step.
Get(AllowFormattingBarState)	A Boolean value representing whether the formatting bar is allowed to be visible.
Get(ApplicationLanguage)	Text representing the current application language (for example, English).

This function	Returns
Get(ApplicationVersion)	Text representing the FileMaker application version.
Get(CalculationRepetitionNumber)	A number representing the repetition of the calculation field that is currently being calculated.
Get(ConnectionAttributes)	The name of the current file’s host and the name of the certificate authority that issued the SSL certificate used to secure the connection.
Get(ConnectionState)	A number representing the security state of the network connection for the current file.
Get(CurrentDate)	The current date according to the system calendar.
Get(CurrentExtendedPrivileges)	A list of keywords for the enabled extended privileges of the account that is being used to evaluate the calculation.
Get(CurrentHostTimestamp)	The host's current date and time (to the nearest second) according to the system clock.
Get(CurrentPrivilegeSetName)	The name of the privilege set that is being used to evaluate this calculation in the database.
Get(CurrentTime)	The current time (to the nearest second) according to the system clock.
Get(CurrentTimestamp)	The current date and time (to the nearest second) according to the system clock.
Get(CurrentTimeUTCMilliseconds)	The current time in Coordinated Universal Time to the nearest millisecond.
Get(CustomMenuSetName)	The name of the active custom menu set.
Get(DesktopPath)	The path to the desktop folder for the current user.
Get(Device)	A number indicating the type of computer or iOS device currently running a FileMaker product.
Get(DocumentsPath)	The path to the Documents folder for the current user.
Get(DocumentsPathListing)	A list of all the files and folders in the Documents folder returned by the Get(DocumentsPath) function.
Get(EncryptionState)	A value representing the current encryption state.
Get(ErrorCaptureState)	A Boolean value representing the state of `Error capture` script step.
Get(FileMakerPath)	The path to the folder of the currently running copy of FileMaker Pro.
Get(FileName)	The name of the currently active database file.
Get(FilePath)	The full path indicating the location of the file.
Get(FileSize)	The size (in bytes) of the currently active database file.
Get(FoundCount)	A number that represents the number of records in the current found set.
Get(HighContrastColor)	The name of the current high contrast default color scheme if **Use High Contrast** is selected in the Windows operating system Accessibility Options dialog box.
Get(HighContrastState)	A Boolean value representing the state of the **Use High Contrast** checkbox on the Accessibility Options dialog box.

This function	Returns
Get(HostApplicationVersion)	The version of FileMaker Pro or FileMaker Server running on the computer that is hosting the current database.
Get(HostIPAddress)	The IP address of the host machine for the current database.
Get(HostName)	The registered name of the computer that is hosting the database file.
Get(InstalledFMPlugins)	The display name, version number (if available), and enabled state of installed plug-ins.
Get(LastError)	A number representing the error, if any, in the execution of the most recently executed script step.
Get(LastMessageChoice)	A number corresponding to the button clicked in an alert message displayed by the Show Custom Dialog script step.
Get(LastODBCError)	A string that shows the error state published by ODBC standards, based on ISO/IEF standards.
Get(LayoutAccess)	A number corresponding to the layout access privileges assigned through the Manage Security dialog box.
Get(LayoutCount)	The total number of layouts in the database file.
Get(LayoutName)	The name of the layout currently displayed.
Get(LayoutNumber)	The number of the layout currently displayed, according to the list in the Manage Layouts dialog box.
Get(LayoutTableName)	The name of the table that the layout is displaying records from.
Get(LayoutViewState)	Information about how the database file is being viewed.
Get(ModifiedFields)	A list of fields that have been modified in the current record of the current table.
Get(MultiUserState)	A number representing the current multi-user state of the database file.
Get(NetworkProtocol)	The name of the network protocol that FileMaker Pro is using on this machine.
Get(NetworkType)	A number representing the type of network being used by FileMaker Pro to access the file that is performing the current script.
Get(PageNumber)	A number representing the current page being printed or previewed.
Get(PersistentID)	Text representing a unique identifier of the computer or device on which FileMaker is running.
Get(PreferencesPath)	The path to the preferences folder for the current user.
Get(PrinterName)	A string identifying the default printer name.
Get(QuickFindText)	The text that was entered in the Quick Find box.
Get(RecordAccess)	A number indicating the access privileges of the current record.
Get(RecordID)	The unique ID number of the current record.
Get(RecordModificationCount)	The total number of times changes to the current record have been committed.
Get(RecordNumber)	The number of the current record in the current found set.

This function	Returns
Get(RecordOpenCount)	The total number of open records in the current found set that haven’t yet been saved.
Get(RecordOpenState)	A number representing the state of the current record.
Get(RequestCount)	The total number of find requests currently defined for the current table.
Get(RequestOmitState)	A Boolean value representing the state of the **Omit** checkbox in Find mode.
Get(ScreenDepth)	The number of bits needed to represent the color or shade of gray of a pixel on the main screen.
Get(ScreenHeight)	The height, in points, of the screen in which the window of the current file is open.
Get(ScreenWidth)	The width, in points, of the screen in which the window of the current file is open.
Get(ScreenWidth)	A number indicating whether or not animations are enabled for the currently running script.
Get(ScriptName)	The name of the script currently running (or paused).
Get(ScriptParameter)	The script parameter passed into the current script.
Get(ScriptResult)	The script result from a performed subscript.
Get(SortState)	A number value representing the current sort state.
Get(StatusAreaState)	A number representing whether the status toolbar is hidden, visible, visible and locked, or hidden and locked.
Get(SystemDrive)	The drive letter (Windows) or the volume name (OS X) where the currently running operating system is located.
Get(SystemIPAddress)	The IP addresses of all the machines connected to a NIC (Network Interface Controller) card.
Get(SystemLanguage)	The language currently set on the current system.
Get(SystemNICAddress)	The hardware addresses of all the Network Interface Controller cards connected to the machine.
Get(SystemPlatform)	A number indicating the current platform.
Get(SystemVersion)	The version of the operating system of the machine on which the function is executed.
Get(TemporaryPath)	The path to the current user’s temporary folder used by FileMaker Pro.
Get(TextRulerVisible)	A Boolean value representing whether or not the text ruler is visible.
Get(TotalRecordCount)	The total number of records in the current table.
Get(TriggerCurrentPanel)	The index number and object name of the tab panel or slide panel to be switched from when the OnPanelSwitch script trigger is activated.
Get(TriggerGestureInfo)	Details about the gesture that activated an OnGestureTap script trigger.
Get(TriggerKeystroke)	A string containing the characters that activated an OnObjectKeystroke or OnLayoutKeystroke script trigger.

This function	Returns
Get(TriggerModifierKeys)	The state of the keyboard modifier keys as they were when the script trigger was activated.
Get(TriggerTargetPanel)	The index number and object name of the tab panel or slide panel to be switched to when the OnPanelSwitch script trigger is activated.
Get(UserCount)	The number of users who are currently accessing the file.
Get(UserName)	The name of the FileMaker Pro user, as specified in the **General** tab of the Preferences dialog box.
Get(UseSystemFormatsState)	A Boolean value representing the state of the **Use System Formats** menu command.
Get(UUID)	Text representing a Universally Unique Identifier (UUID).
Get(WindowContentHeight)	The height, in points, of the FileMaker Pro content area.
Get(WindowContentWidth)	The width, in points, of the FileMaker Pro content area.
Get(WindowDesktopHeight)	The height, in points, of the desktop space.
Get(WindowDesktopWidth)	The width, in points, of the desktop space.
Get(WindowHeight)	The height, in points, of the window on which the script is acting (not necessarily the foreground window).
Get(WindowLeft)	The horizontal distance, in points, of the outer edge of the window on which the script is acting (not necessarily the foreground window) relative to the left-most edge of the screen.
Get(WindowMode)	A number representing whether FileMaker Pro is in Browse mode, Find mode, Preview mode, or printing when the function is evaluated.
Get(WindowName)	The name of the current window of the file in which the calculation is defined.
Get(WindowOrientation)	A number indicating the orientation of the window that the current script is acting on.
Get(WindowStyle)	A number indicating whether the top-most open window is a document window, a floating document window, or a dialog window.
Get(WindowTop)	The vertical distance, in points, of the outer edge of the window on which the script is acting (not necessarily the foreground window) relative to the bottom edge of the menu bar.
Get(WindowVisible)	A Boolean value representing whether or not the current window is visible.
Get(WindowWidth)	The width, in points, of the window on which the script is acting (not necessarily the foreground window).
Get(WindowZoomLevel)	The zoom level of the current window.

Get functions example

This script uses the function `Get(CurrentDate)` to check each record in the found set to see if an account is past due. If an account is past due, the script shows a message and prompts the user to click a button labeled Ignore, Send Letter, or Send Mail (set up through the Show Custom Dialog script step). The script captures the user's response using `Get(LastMessageChoice)`. Then, based on the user's response, the script performs an action: it cancels the rest of the script, prints a "payment is late" letter, or sends email to the associated account.

```
Go to Layout ["LayoutName"]
Go to Record/Request/Page [First]
Loop
  If [DatabaseName::Date < Get(CurrentDate) - 30]
    Show Custom Dialog ["30 or more days late"]
    If [Get(LastMessageChoice) = 1]
      Halt Script
    Else If [Get(LastMessageChoice) = 2]
      Go to Layout ["Late Notice"]
      Print []
    Else
      Send Mail [To: DatabaseName::Client; Subject: "Late Notice";
Message: "Your account is past due."]
    End If
  End If
  Go to Record/Request/Page [Exit after last, Next]
End Loop
Go to Layout [original layout]
```

Get(AccountExtendedPrivileges)

Purpose

Returns a list of keywords, separated by carriage returns, for the enabled extended privileges. The list that is returned is based on the account used to open the database file. See also Get(CurrentExtendedPrivileges) function.

Format

`Get(AccountExtendedPrivileges)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 11.0

Description

Extended privileges are additional access rights assigned to an account's privilege set. See About accounts, privilege sets, and extended privileges.

Returns an empty list if a user doesn't have extended privileges assigned to the account used to open the database file.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

An account uses a privilege set that includes the extended privilege of Access Via FileMaker WebDirect™ (keyword "fmwebdirect"):

`Position(Get(AccountExtendedPrivileges); "fmwebdirect"; 1; 1)` returns a value greater than 0.

If you are logged in and running a script that is set to run with full access privileges, `Get(AccountExtendedPrivileges)`returns the extended privileges for your account, but `Get(CurrentExtendedPrivileges)`returns the extended privileges for the Admin account.

Get(AccountName)

Purpose

Returns the name of the authenticated account being used by the current user of the database file.

Format

`Get(AccountName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Use this function for FileMaker authentication. If a user is using the default Admin account, `Get(AccountName)` returns **Admin**. If a user is using the FileMaker Pro guest account then **[Guest]** will be returned.

For external server authentication, `Get(AccountName)` returns the name of the authenticated account being used by the current user of the database file, not the group the user belongs to (the group name appears in the **Account** list when you define accounts and privileges in FileMaker Pro). If an individual belongs to more than one group (account), the first group name listed when you choose **View By Authentication Order** while defining accounts and privileges determines access for the user.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available a http://help.filemaker.com.

Examples

Returns **Marketing** when Marketing is the name of the account that was used to log in to the database file.

Get(AccountPrivilegeSetName)

Purpose

Returns the name of the privilege set that is being used by the account used to open the database. See also Get(CurrentPrivilegeSetName) function.

Format

`Get(AccountPrivilegeSetName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 11.0

Description

If a user is using the default Admin account and you haven't modified access privileges for the database file, this function returns **[Full Access]**.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com

Examples

For an Administrator, `Get(AccountPrivilegeSetName)` might return **[Full Access]**.

For a user in the sales department, `Get(AccountPrivilegeSetName)` might return **[Data Entry Only]**.

For a user with Read-Only Access to a database who is running a script that is set to run with full access privileges, `Get(AccountPrivilegeSetName)`returns **[Read-Only Access]** but `Get(CurrentPrivilegeSetName)`returns **[Full Access]** (for the current script).

Get(ActiveFieldContents)

Purpose

Returns the contents of the field that has the focus.

Format

`Get(ActiveFieldContents)`

Parameters

None

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

When the focus is in a repeating field, this function returns the contents of the active repetition. The result type of the active field depends upon the data type of the active field and the result type assigned to the `Get(ActiveFieldContents)` calculation function.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **SomeShop** when the focus is in the Name field, and that field contains the data SomeShop.

This type of calculation is most useful if used in a script when you want to examine data in different fields as the script proceeds.

Get(ActiveFieldName)

Purpose

Returns the name of the field that has the focus.

Format

`Get(ActiveFieldName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Country**, when the focus is in the Country field.

Get(ActiveFieldTableName)

Purpose

Returns the name of the table that contains the active field (the field that has the focus).

Format

`Get(ActiveFieldTableName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

If there is no active field, this function returns an empty string.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

There are two fields, Teachers::Name and Coaches::Name, on the current layout. Creating a script that returns the result of `Get(ActiveFieldTableName)` to a third field will return **Teachers** when the script is performed after clicking in the Teachers::Name field, or will return **Coaches** after clicking in the Coaches::Name field.

Get(ActiveLayoutObjectName)

Purpose

Returns the object name of the active layout object in the calculation's current window; otherwise, returns an empty string.

Format

`Get(ActiveLayoutObjectName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.5

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

There is a named button on the current layout called `cancelButton`. When the focus is on the button, `Get(ActiveLayoutObjectName)` returns **cancelButton**.

Get(ActiveModifierKeys)

Purpose

Returns a number representing the keyboard modifier keys (for example, Control+Shift) that are being pressed.

Format

```
Get(ActiveModifierKeys)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The number returned is calculated by summing numbers representing each modifier key being pressed. The values assigned to the keys are:

- Shift = 1
- Caps Lock = 2
- Ctrl (Windows) and Control (OS X) = 4
- Alt (Windows) and Option (OS X) = 8
- Command (OS X) = 16

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns the number **9** when Shift+Alt is pressed on a computer running Windows.

You could use this function in a script that includes a custom dialog box script step (with an **OK** and **Cancel** button) to perform some special action if the user presses the Alt (or Option) key while clicking **OK**.

Get(ActivePortalRowNumber)

Purpose

Returns the number of the portal row containing the focus.

Format

`Get(ActivePortalRowNumber)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

When no portal row contains the focus, this function returns **0**. If there are multiple windows open in the current database file, each window can have its own portal row number value, but results are returned for only the foreground window. If a user navigates to a portal without selecting a specific portal row and without making an object active within a specific portal row, Get(ActivePortalRowNumber) returns row 0 rather than row 1.

Notes

- If a field on a layout is defined as `Get(ActivePortalRowNumber)`, the window must be refreshed before the field will display the current portal row number.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when the fifth row of a portal has the focus, or when the focus is in a field in the fifth portal row.

Returns **1** after the `Go to Portal Row [First]` script step runs.

Returns **0** when a portal is not selected.

Get(ActiveRepetitionNumber)

Purpose

Returns a number representing the active repetition of a repeating field (the repetition that has the focus).

Format

```
Get(ActiveRepetitionNumber)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The first repetition returns **1**. If the current field isn't a repeating field, this function returns **1**.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when the focus is in the fifth repetition of a repeating field.

Get(ActiveSelectionSize)

Purpose

Returns a number representing how many characters are selected.

Format

`Get(ActiveSelectionSize)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

Returns **0** if there is no selection.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **4** when 4 characters are selected.

Get(ActiveSelectionStart)

Purpose

Returns a number representing the starting character of the selected text.

Format

```
Get(ActiveSelectionStart)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

Returns the cursor's current position if no text is selected.

If there are multiple windows open in the current database file, a result is returned for only the foreground window.

Notes

- In FileMaker WebDirect, Get(ActiveSelectionStart) returns a value only if the selected text is in a field that displays as an edit box.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when the selection starts at character 5.

Get(AllowAbortState)

Purpose

Returns **1** if `Allow user abort` script step is on; otherwise, returns **0**.

Format

`Get(AllowAbortState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** if `Allow user abort` script step is on.

Get(AllowFormattingBarState)

Purpose

Returns a Boolean value representing whether the formatting bar is allowed to be visible.

Format

`Get(AllowFormattingBarState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns **1** if the formatting bar is allowed; otherwise, returns **0**.

The Allow Formatting Bar script step sets the formatting bar state. For more information, see Allow Formatting Bar script step.

Notes

- In FileMaker WebDirect, this function is not supported and returns **0**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** if the formatting bar is allowed to be visible.

Get(ApplicationLanguage)

Purpose

Returns text representing the current application language.

Format

```
Get(ApplicationLanguage)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

The text that is returned by this function is in the English language.

For hosted databases, Get(ApplicationLanguage) returns the client's current language.

Note In FileMaker WebDirect, Get(ApplicationLanguage) returns the web browser's current language.

FileMaker Pro supports:

- English
- French
- Italian
- German
- Swedish
- Spanish
- Dutch
- Japanese
- Simplified Chinese
- Brazilian Portuguese
- Korean

Examples

Returns **English** when the current application language is English.

Get(ApplicationVersion)

Purpose

Returns text representing the FileMaker application and version.

Format

`Get(ApplicationVersion)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **Pro *version*** for FileMaker Pro
- **ProAdvanced *version*** for FileMaker Pro Advanced
- **Runtime *version*** for FileMaker Runtime
- **Web Publishing Engine *version*** for FileMaker Server Web Client
- **xDBC *version*** for xDBC Client
- **Server *version*** for FileMaker Server
- **Go *version*** for FileMaker Go® on the iPhone or iPod touch
- **Go_iPad *version*** for FileMaker Go on the iPad

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Pro 13.0v1** in FileMaker Pro 13.0v1.

Get(CalculationRepetitionNumber)

Purpose

Returns a number representing the repetition of the calculation field that is currently being calculated.

Format

```
Get(CalculationRepetitionNumber)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The first repetition returned is 1. If the current field isn't a repeating field, the function returns **1**.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when FileMaker Pro is calculating the fifth repetition of a repeating field.

Get(ConnectionAttributes)

Purpose

Returns the name of the current file's host and the name of the certificate authority that issued the SSL certificate used to secure the connection.

Format

```
Get(ConnectionAttributes)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 13.0

Description

Get(ConnectionAttributes) returns an empty string if:

- the current file is not hosted
- the host is not FileMaker Server
- the host does not use an SSL certificate to secure the connection to the client

Examples

If the host is named "group_server" and the current SSL certificate was issued by XYZ Inc., Get(ConnectionAttributes) returns:

[Peer Certificate]
commonName: group_server
CA Issuers: XYZ Inc.

Get(ConnectionState)

Purpose

Returns a number representing the security state of the network connection for the current file.

Format

```
Get(ConnectionState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 12.0

Description

Returns a value indicating whether the FileMaker Pro or FileMaker Go connection to the host uses SSL, including whether the Server name matches the Server-side certificate (providing the highest security). Returns:

- **0** for no network connection for the current file.
- **1** for a non-secured connection (FileMaker Server with SSL disabled, or to a FileMaker Pro host).
- **2** for a secured connection (SSL) when the server name doesn't match the certificate (default FileMaker Server installation).
- **3** for a secured connection with a fully verified server name in the certificate.

Notes

- You do not have to use Get(ConnectionState) to check the security of web published files. In FileMaker WebDirect, the web browser verifies the SSL connection.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **3** when the Server name matches the name indicated in a Server root certificate obtained from a trusted authority.

Get(CurrentDate)

Purpose

Returns the current date according to the system calendar.

Format

`Get(CurrentDate)`

Parameters

None

Data type returned

date

Originated in

FileMaker Pro 6.0 or earlier

Description

The format of the result of this function varies based on the date format that was in use when the database file was created. In the United States, dates are generally in the format MM/DD/YYYY. You can change the date and time formats in your operating system.

If the result is displayed in a field, it is formatted according to the date format of the field in the current layout.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

Returns **2/2/2014** when the system date is set to February 2, 2014.

`Get(CurrentDate)-Date(1;5;2014)` returns **4** when the system date is set to January 9, 2014.

Get(CurrentExtendedPrivileges)

Purpose

Returns a list of keywords, separated by carriage returns, for the enabled extended privileges. The list that is returned is based on the account that is being used to evaluate this calculation. See also Get(AccountExtendedPrivileges) function.

Format

`Get(CurrentExtendedPrivileges)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Extended privileges are additional access rights assigned to an account's privilege set. See About accounts, privilege sets, and extended privileges.

Returns an empty list if a user doesn't have extended privileges assigned for the current database file.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

The account that is evaluating this calculation uses a privilege set that includes the extended privilege of Access Via FileMaker WebDirect (keyword "fmwebdirect"):

`Position(Get(CurrentExtendedPrivileges); "fmwebdirect"; 1; 1)` returns a value greater than 0.

If you are logged in and running a script that is set to run with full access privileges, `Get(AccountExtendedPrivileges)`returns the extended privileges for your account, but `Get(CurrentExtendedPrivileges)`returns the extended privileges for the Admin account.

Get(CurrentHostTimestamp)

Purpose

Returns the host's current date and time (to the nearest second) according to the system clock.

Format

`Get(CurrentHostTimestamp)`

Parameters

None

Data type returned

timestamp

Originated in

FileMaker Pro 7.0

Description

The format of the value returned is determined by the database file's settings. You can use your client system's settings in the operating system.

Notes

- The client machine and host machine may be in different times zones so `Get(CurrentHostTimestamp)` and `Get(CurrentTimestamp)` may return different date/time values. Also, the current date and time are characteristics of the host system, but the format of the date and time is a characteristic of the database file.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Important For users who are connected over a network, the `Get(CurrentHostTimestamp)` function can affect the performance of the database file. For example, if you use the function in an unstored calculation field, and the field is visible in a list view, each display of the field requires an additional network access. Stored calculation fields are a better use of the function. For example, if you automatically enter a timestamp for each newly created record using a stored calculation field, you minimize network access.

Examples

Returns **1/1/2014 11:30:01 AM** when the system clock shows January 1, 2014 11:30:01 AM on the host machine.

Get(CurrentPrivilegeSetName)

Purpose

Returns the name of the privilege set that is being used to evaluate this calculation in the database file. See also Get(AccountPrivilegeSetName) function.

Format

`Get(CurrentPrivilegeSetName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If a user is using the default Admin account and you haven't modified access privileges for the database file, this function returns **[Full Access]**.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- If you select the **Run script with full access privileges** script option, this function returns **[Full Access]**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

For current user Administrator, `Get(CurrentPrivilegeSetName)` might return **[Full Access]**.

For a current user in the sales department, `Get(CurrentPrivilegeSetName)` might return **[Data Entry Only]**.

For a user with Read-Only Access to a database who is running a script that is set to run with full access privileges, `Get(AccountPrivilegeSetName)`returns **[Read-Only Access]** but `Get(CurrentPrivilegeSetName)`returns **[Full Access]** (for the current script).

Get(CurrentTime)

Purpose

Returns `CurrentTime` (to the nearest second) according to the system clock.

Format

`Get(CurrentTime)`

Parameters

None

Data type returned

time

Originated in

FileMaker Pro 6.0 or earlier

Description

The format of the value returned is determined by the operating system settings.

Notes

- If the calculation result for this script step is set to integer format, it will return the total number of seconds elapsed since the start of the current day.
- In client/server and peer-to-peer environments, `Get(CurrentTimestamp)` evaluates the status of the client machine running the script (not the host machine). For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **11:30:00 AM** when the function result is in text format and the system clock shows 11:30:00 AM.

Get(CurrentTimestamp)

Purpose

Returns the current date and time (to the nearest second) according to the system clock.

Format

`Get(CurrentTimestamp)`

Parameters

None

Data type returned

timestamp

Originated in

FileMaker Pro 7.0

Description

The format of the value returned is determined by the operating system settings.

Note In client/server and peer-to-peer environments, `Get(CurrentTimestamp)` evaluates the status of the client machine running the script (not the host machine). For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1/1/2014 11:30:00 AM** when the system clock shows January 1, 2014 11:30:00.

Get(CurrentTimeUTCMilliseconds)

Purpose

Returns the current time in Coordinated Universal Time (UTC) to the nearest millisecond.

Format

`Get(CurrentTimeUTCMilliseconds)`

Parameters

None

Data type returned

number, time

Originated in

FileMaker Pro 13.0

Description

Returns the current time without time zone adjustments in the form of the number of milliseconds since 1/1/0001. UTC time zone adjustments must be applied to get your local time.

Examples

`Get(CurrentTimeUTCMilliseconds)` returns **63568967107528** if the time in UTC is 10:25:07.528 PM on 6/3/2015.

`GetAsTimestamp((Get(CurrentTimeUTCMilliseconds)+ (Location::TimeAdjustment * 3600000))/1000)` returns **11/10/2015 1:43:55.304 PM** if the time in UTC is 8:43:55:.304 PM on 11/10/2015 and the TimeAdjustment field has a value of -7.

Get(CustomMenuSetName)

Purpose

Returns the name of the active custom menu set.

Format

```
Get(CustomMenuSetName)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

If the active menu set isn't a custom menu set, an empty string is returned.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Custom Menu Set #1** when this custom menu set is active.

Returns an empty string when the **[Standard FileMaker Menus]** menu set is active.

Get(DesktopPath)

Purpose

Returns the path to the desktop folder for the current user.

Format

`Get(DesktopPath)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In Windows, the path format is /Drive:/Users/UserName/Desktop/.

In OS X, the path format is /DriveName/Users/username/Desktop/.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **/C:/Documents and Settings/John Smith/Desktop/** for a user named John Smith in Windows.

Returns **/Macintosh HD/Users/johnsmith/Desktop/** for a user named John Smith in OS X.

Get(Device)

Purpose

Returns a number indicating the type of computer that is currently running FileMaker Pro or FileMaker WebDirect, or the type of iOS device that is currently running FileMaker Go.

Format

```
Get(Device)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 13.0

Description

Returns:

- **0** if the device is unknown
- **1** if the device is a Mac
- **2** if the device is a PC running Windows
- **3** if the device is an iPad
- **4** if the device is an iPhone or iPod touch

Examples

Returns **2** if FileMaker Pro or FileMaker WebDirect is currently running on a PC with Windows.

Returns **3** if FileMaker Go or FileMaker WebDirect is currently running on an iPad.

Get(DocumentsPath)

Purpose

Returns the path to the Documents folder for the current user.

Format

`Get(DocumentsPath)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In Windows, the path format is /Drive:/Users/UserName/Documents/.

In OS X, the path format is /DriveName/Users/username/Documents/.

When running on FileMaker Server, Get(DocumentsPath) returns the location of the Documents folder, which is in the same folder as the server's Backups, Databases, and Scripts folders. The Documents folder is used as a shared location that scripts from different sessions or other processes on the machine can use to import or export files. For more information, see FileMaker Server Help.

Notes

- In FileMaker WebDirect, this function is not supported and always an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

For FileMaker Pro, for a user named John Smith, returns:

/C:/Users/John Smith/Documents/ in Windows.

/Macintosh HD/Users/johnsmith/Documents/ in OS X

For FileMaker Server, returns:

/C:/Program Files/FileMaker/FileMaker Server/Data/Documents in Windows Vista.

/Macintosh HD/Library/FileMaker Server/Data/Documents in OS X

Get(DocumentsPathListing)

Purpose

Returns a list of all the files and folders in the Documents folder returned by the Get(DocumentsPath) function.

Format

`Get(DocumentsPathListing)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 10.0

Description

Each pathname in the Documents folder is listed separated by a line break. Files and folders are named according to FileMaker Pro naming conventions.

Use Get(DocumentsPathListing) with the Import Records script step and Export Records script step to determine if a file exists in the Documents folder before using the Open File script step to open the file. Get(DocumentsPathListing) ensures that multiple scripts can safely read from and write to the same FileMaker Pro database.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

For FileMaker Server, returns the following pathnames:

In Windows:

/C:/Program Files/FileMaker/FileMaker Server/Data/Documents/lastmonthsales.xlsx
/C:/Program Files/FileMaker/FileMaker Server/Data/Documents/forecastsales.xlsx
/C:/Program Files/FileMaker/FileMaker Server/Data/Documents/SAP
/C:/Program Files/FileMaker/FileMaker Server/Data/Documents/SAP/sap001.txt
/C:/Program Files/FileMaker/FileMaker Server/Data/Documents/SAP/sap002.txt

In OS X:

/MacintoshHD//Library/FileMaker Server/Data/Documents/lastmonthsales.xlsx
/MacintoshHD//Library/FileMaker Server/Data/Documents/forecastsales.xlsx
/MacintoshHD//Library/FileMaker Server/Data/Documents/SAP
/MacintoshHD//Library/FileMaker Server/Data/Documents/SAP/sap001.txt
/MacintoshHD//Library/FileMaker Server/Data/Documents/SAP/sap002.txt

Get(EncryptionState)

Purpose

Returns a value representing the file's current encryption state.

Format

`Get(EncryptionState)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 13.0

Description

Returns:

- **0** if the database is not encrypted
- **1** and the Shared ID as a return delimited list if the database is encrypted

Examples

In an encrypted database file with the Shared ID of 31725, `Get(EncryptionState)`returns:

- 1
- 31725

Get(ErrorCaptureState)

Purpose

Returns **1** if the `Set Error capture` script step is on; otherwise, returns **0**.

Format

`Get(ErrorCaptureState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** if the `Set Error capture` script step is on.

Get(FileMakerPath)

Purpose

Returns the path to the folder of the currently running copy of FileMaker Pro.

Format

`Get(FileMakerPath)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In Windows, the path format is /Drive:/Program Files/FileMaker/FileMaker Pro 13/.

In OS X, the path format is /DriveName/Applications/FileMaker Pro 13/.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **/C:/Program Files/FileMaker/FileMaker Pro 13/** in Windows.

Returns **/MacintoshHD/Applications/FileMaker Pro 13/** in OS X.

Get(FileName)

Purpose

Returns the name of the currently active database file, without the filename extension.

Format

```
Get(FileName)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Contacts** when Contacts is the active file.

Get(FilePath)

Purpose

Returns the full path indicating the location of the currently active database file.

Format

`Get(FilePath)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

In Windows, the full path is file:/drive:/folder/filename for local files. For remote files, the full path is file://volume/folder/filename.

In OS X, the full path is file:/volume/folder/filename for local and remote files.

If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **file:/driveletter:/databaseName** for local files in Windows.

Returns **file://volumename/myfoldername/databaseName** for remote files in Windows.

Returns **file:/path/databaseName** for local and remote files in OS X.

Returns **fmnet:/networkaddress/databaseName** for FileMaker Pro networked files.

Get(FileSize)

Purpose

Returns the size (in bytes) of the currently active database file.

Format

`Get(FileSize)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **15000** when the current file size is 15000 bytes.

Get(FoundCount)

Purpose

Returns a number that represents the number of records in the current found set.

Format

`Get(FoundCount)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If there are multiple windows open in the current database file, each window can have its own found count value, but results are returned for only the foreground window.

If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **7** when there are 7 records in the current found set.

Get(HighContrastColor)

Purpose

Returns the name of the current high contrast default color scheme if high contrast is enabled in the Windows Ease of Access Center and high contrast color scheme is active.

Format

```
Get(HighContrastColor)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns an empty string if **Turn high contrast** is unavailable, inactive, or if this function is used in OS X.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **High Contrast White** when the Windows theme is set to High Contrast White.

Get(HighContrastState)

Purpose

Returns a Boolean value representing the state of the **Turn high contrast** option in the Windows Ease of Access Center.

Format

```
Get(HighContrastState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** if **Use High Contrast** is unavailable, inactive, or if the function is used in OS X.
- **1** if **Use High Contrast** is available and active.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Get(HostApplicationVersion)

Purpose

Returns the version of FileMaker Pro or FileMaker Server running on the computer that is hosting the current database.

Format

```
Get(HostApplicationVersion)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 9.0

Description

Displays a value when used with the same or higher version of FileMaker Pro or FileMaker Server software. If the current database is not shared or hosted, this function returns an empty string. Also returns an empty string when used from the host computer itself.

Examples

Returns **Pro 13.0v1** when the host computer is running FileMaker Pro 13 version 1.

Returns **ProAdvanced 13.0v1** when the host computer is running FileMaker Pro 13 Advanced version 1.

Returns **Server 13.0v1** when the host computer is running FileMaker Server 13 version 1.

Get(HostIPAddress)

Purpose

Returns the IP address of the host machine for the current database.

Format

```
Get(HostIPAddress)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

Returns the IP address used to connect to the host machine for the current database. If the current database isn't being hosted, an empty string is returned.

Note In FileMaker WebDirect, returns the physical IP address of the host machine.

If IPv4 and IPv6 addresses are available for remotely hosted files, the address is returned in the most common or default format. This might not be the same format that was used when connecting to the host.

If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns one of the following when the current database is being hosted:

- IPv4: **14.156.13.121**
- IPv6: **[2001:0DB8:85A3:08D3:1319:8A2E:0370:7334]**

Note If the host machine has both IPv4 and IPv6 addresses, Get(HostIPAddress) returns only the IP address that the client used to connect to the host.

Returns one of the following when accessing a locally hosted database:

- **127.0.0.1** if connected to 127.0.0.1
- **[::1]** if connected to localhost
- **14.156.13.121** if connected to the computer's IP address or evaluated in FileMaker WebDirect.

Get(HostName)

Purpose

Returns the registered name of the computer that is hosting the database file.

Format

`Get(HostName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

On the computer that is hosting the database file:

- Windows 7: Choose **Start** menu > **Control Panel** > **System and Security** > **System** > and then click **See the name of this computer**. **Computer name** displays the current registered name.
- Windows 8: In the navigation panel of a File Explorer window, choose **Computer**, then choose **Computer** menu > **Open Control Panel** > **System and Security** > **System**. **Computer Name** displays the current registered name.
- OS X: In the Sharing System Preference, **Computer Name** displays the current registered name.

Notes

- If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Fred Jones** when Fred Jones is the registered name of the host computer in use.

Get(InstalledFMPlugins)

Purpose

Returns the name, version number (if available), and enabled state of installed plug-ins.

Format

```
Get(InstalledFMPlugins)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

The Get(InstalledFMPlugins) function is useful for determining whether an installed plug-in is newer or older than a plug-in required by a file.

If multiple plug-ins are installed, Get(InstalledFMPlugins) returns values for each plug-in on separate lines, separated by carriage returns.

Get(InstalledFMPlugins) returns plug-in version information only when plug-in developers have entered version information in the resource file (Windows) or the info.plist file (OS X).

OS X: Plug-ins are stored as packages.

The enabled state is returned as follows:

- **Enabled** The plug-in is enabled in the FileMaker Pro preferences and can be loaded.
- **Disabled** The plug-in is disabled in the FileMaker Pro preferences and cannot be loaded.
- **Ignored** The plug-in failed to load, which could be due to software incompatibility.

Examples

When:

MyPlugin1 is installed and is enabled in the Plug-ins tab in the Preferences dialog box.

MyPlugin2 is installed and is disabled in the Plug-ins tab in the Preferences dialog box.

MyPlugin3 could not be loaded.

Get(InstalledFMPlugins) returns:

MyPlugin1;1.0;Enabled

MyPlugin2;1.1;Disabled

MyPlugin3; ;Ignored

Get(LastError)

Purpose

Returns a number representing the error, if any, in the execution of the most recently executed script step.

Format

`Get(LastError)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to detect and control the outcome of errors. See FileMaker Pro error codes.

Notes

- OS X: In FileMaker Pro, if an error occurs while performing an AppleScript from the Manage Scripts feature, the AppleScript error code will be returned.
- For ODBC imports and Execute SQL script steps, if an error occurs while performing a SQL query, returns FileMaker error 1408. For detailed information about the error, use the Get(LastODBCError) function. If there is no information about the error, returns FileMaker error 1409.
- For working with ODBC data sources in the relationships graph, returns FileMaker error 1408.
- Some script triggers allow for the activating command or event to be canceled if the script executed by the script trigger returns a False value. When a command or event is canceled this way, the error code is set to 20.
- When you perform a script that uses this function with control script steps, the control script steps do not clear the last error condition reported by FileMaker Pro.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Tip To create a script that responds to errors without displaying alerts, use this function with the Set Error Capture script step with the **On** option.

Examples

Returns **0** when the most recent script step executed successfully.

Returns **401** when no records are found after the Perform Find script step has been executed.

Get(LastMessageChoice)

Purpose

Returns a number corresponding to the button clicked in an alert message that is displayed by the Show Custom Dialog script step.

Format

```
Get(LastMessageChoice)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **1** for the first button (by default, labeled OK)
- **2** for the second button (by default, labeled Cancel)
- **3** for the third button

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Get(LastODBCError)

Purpose

Returns a string that shows the error state published by ODBC standards, based on ISO/IEF standards.

Format

`Get(LastODBCError)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

- For ODBC imports and Execute SQL script steps, returns a detailed, textual ODBC error message.
- For working with ODBC data sources in the relationships graph, returns the readable error string that is generated by the ODBC driver.

Notes

- You can set the Set Error Capture state to "on" to suppress the error messages. You can also use `Get(LastError)` to get generic errors. See FileMaker Pro error codes.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

For ODBC imports and Execute SQL script steps, returns **[DataDirect][Macintosh ODBC Driver Manager] Data source name not found and no default driver specified (-1)** when a data source name wasn't found and the driver wasn't specified.

Get(LayoutAccess)

Purpose

Returns a number based on record access privileges available through the current layout.

Format

`Get(LayoutAccess)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

You assign the privileges in the Custom Layout Privileges dialog box.

Returns:

- **0** if the custom layout privileges of an account's privilege set allow **no access** to **Records via this layout**
- **1** if the custom layout privileges of an account's privilege set allow **view only** access to **Records via this layout**. If the database is opened with read-only access, FileMaker Pro returns **1** even if the layout has read-write access privileges
- **2** if the custom layout privileges of an account's privilege set allow **modifiable** access to **Records via this layout**

Notes

- `Get(LayoutAccess)` returns information about record access privileges defined for only the current layout. It ignores current record access privileges for all other layouts. To fully check access through a layout, consider the return values of `Get(LayoutAccess)` and the Get(RecordAccess) function.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.
- See Editing layouts privileges for more details about limiting access through layouts.

Examples

Returns **1** when the layout allows view-only access to records.

Get(LayoutCount)

Purpose

Returns the total number of layouts in the database file.

Format

```
Get(LayoutCount)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **3** when the file has three layouts.

Get(LayoutName)

Purpose

Returns the name of the layout currently displayed.

Format

```
Get(LayoutName)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If there are multiple windows open in the current database file, each window can have its own layout name value, but results are returned for only the foreground window.

Notes

- You can use the Get(LayoutNumber) function as an alternative to Get(LayoutName) if there are multiple layouts with the same name.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Product List** when the Product List layout is displayed.

Returns **Customer Invoice** when the Customer Invoice layout is displayed.

Get(LayoutNumber)

Purpose

Returns the number of the layout currently displayed, according to the list in the Manage Layouts dialog box.

Format

```
Get(LayoutNumber)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If there are multiple windows open in the current database file, each window can have its own layout number value, but results are returned for only the foreground window.

Notes

- You can use Get(LayoutNumber) as an alternative to the Get(LayoutName) function if there are multiple layouts with the same name.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **3** when the current layout is third in the list of layouts in Manage Layouts.

Get(LayoutTableName)

Purpose

Returns the name of the table from which the current layout is displaying records.

Format

`Get(LayoutTableName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

If no windows are open, an empty string is returned.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

There are two layouts, Teachers Layout and Coaches Layout, with corresponding tables named Teachers and Coaches in the table Instructors. An unstored calculation of `Get(LayoutTableName)` returns **Teachers** when the current layout is Teachers Layout and returns **Coaches** when the current layout is Coaches Layout.

Get(LayoutViewState)

Purpose

Returns a number indicating the currently active database file view.

Format

```
Get(LayoutViewState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** (zero) if the database file is in Form View
- **1** if the database file is in List View
- **2** if the database file is in Table View

If there are multiple windows open in the current database file, each window can have its own layout view state value, but results are returned for only the foreground window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Get(ModifiedFields)

Purpose

Returns a list of fields that have been modified in the current record of the current table.

Format

`Get(ModifiedFields)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 13.0

Description

This function returns a list of carriage return-delimited values.

Examples

When the Invoices::Customer Name and Invoices::Company fields are modified and the record is open, `Get(ModifiedFields)` returns:

- Customer Name
- Company

Get(MultiUserState)

Purpose

Returns a number representing the level of sharing for the database file using FileMaker Network.

Format

`Get(MultiUserState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** when network sharing is off
- **1** when network sharing is on, you're accessing the database file from the host computer, and either all users or a specific group of users (based on their privilege set) have network access to the database file
- **2** when network sharing is on, you're accessing the database file from a client computer, and either all users or a specific group of users (based on their privilege set) have network access to the database file

Notes

- If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **0** when access is denied to other users.

Get(NetworkProtocol)

Purpose

Returns the name of the network protocol (TCP/IP) that FileMaker Pro is using on this machine.

Format

`Get(NetworkProtocol)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **TCP/IP**.

Get(NetworkType)

Purpose

In FileMaker Go, returns a number indicating the type of network being used to access the current file.

Format

```
Get(NetworkType)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 13.0

Description

In FileMaker Go, returns:

- **0** if the current file is a local file on an iOS device
- **1** if the network type is unknown
- **2** for a cellular network
- **3** for a Wi-Fi network

Note In other products in the FileMaker product line, this function is not supported and returns an empty string.

Examples

Returns **3** when the file is being accessed from a Wi-Fi network.

Get(PageNumber)

Purpose

Returns a number representing the current page being printed or previewed.

Format

```
Get(PageNumber)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If nothing is being printed or previewed, **0** is returned.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **4** when page 4 is being printed or previewed.

Get(PersistentID)

Purpose

Returns text representing a unique identifier of the computer or device on which FileMaker is running.

Format

`Get(PersistentID)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

Returns a unique, unchanging identifier for the computer on which FileMaker Pro is running, the device on which FileMaker Go is running, or the current FileMaker WebDirect session in the form of a 32-digit hexadecimal string. Get(PersistentID) helps you identify devices that access your solution.

Notes

- If web browser cookies are cleared during a FileMaker WebDirect session, the value returned by Get(PersistentID) changes.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

For a FileMaker Pro client or a FileMaker Go client, Get (PersistentID) returns a value such as 78569d0bd40b898a64e7d08ccdea8220.

Get(PreferencesPath)

Purpose

Returns the path to the preferences and default options folder for the current user.

Format

```
Get(PreferencesPath)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In Windows, the path format is /Drive:/Users/UserName/AppData/Local/.

In OS X, the path format is /DriveName/Users/UserName/Library/Preferences/.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **/C:/Users/John Smith/AppData/Local/** for a user named John Smith in Windows.

Returns **/MacintoshHD/Users/John Smith/Library/Preferences/** for a user named John Smith in OS X.

Get(PrinterName)

Purpose

Returns a string identifying the default printer name.

Format

```
Get(PrinterName)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

In Windows, returns a string with each of these entries separated by a comma:

- the printer name
- the driver name
- the name of the printer port

In OS X, returns a string with these entries separated by the word **on**:

- the queue name of the printer (if provided)
- the IP address of the printer

If any of this information isn't available, **<Unknown>** is inserted in the result (except for queue name in OS X).

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **HP LaserJet 4, WINSPOOL, LPT1** in Windows.

Returns **24.109.265.43** in OS X.

Get(QuickFindText)

Purpose

Returns the text that was entered in the Quick Find box.

Format

`Get(QuickFindText)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 11.0

Description

Returns the text that was entered the last time quick find was performed.

Examples

Returns **New York** if the last search that was entered in the Quick Find box was for New York.

Get(RecordAccess)

Purpose

Returns a number based on the current record's access privileges, assigned through the Custom Record Privileges dialog box.

Format

`Get(RecordAccess)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** if the custom record privileges of an account's privilege set have neither **View** nor **Edit** privileges set to **yes** for the current record
- **1** if the custom record privileges of an account's privilege set have **View** set to **yes** for the current record, or if **View** is set to **limited** and the calculation defined for limited access returns a value of **true**

  Note If both **View** and **Edit** are set to **yes**, `Get(RecordAccess)` returns **2**
- **2** if the custom record privileges of an account's privilege set have **Edit** set to **yes** for the current record, or if **Edit** is set to **limited** and the calculation defined for limited access returns a value of **true**

Notes

- `Get(RecordAccess)` only returns information about the privileges defined for accessing records. It ignores access privileges assigned through individual layouts. To fully check access to a record, consider the return values of the Get(LayoutAccess) function and `Get(RecordAccess)`.
- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.
- See Editing record access privileges for more details about limiting access to records.

Examples

Returns **1** when the record access is view-only.

Get(RecordID)

Purpose

Returns the unique ID number of the current record.

Format

`Get(RecordID)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The number returned is a decimal value (an integer) generated by FileMaker Pro when the record is created. It does not change.

Notes

- If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.
- `Get(RecordID)`may not return a consistent value for records in ODBC data sources.

Examples

Returns a unique ID for the current record.

Get(RecordModificationCount)

Purpose

Returns the total number of times changes to the current record have been committed.

Format

`Get(RecordModificationCount)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

To commit changes, you can, for example:

- click out of all fields (exit the record)
- go to a different record
- enter Find mode

If multiple windows are open, clicking in another window does not commit the record.

Notes

- If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- `Get(RecordModificationCount)`returns **NULL** for ODBC data sources.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **0** if the record has not been modified since it was created.

If changes are made to four fields and all four fields are committed together, the result increments by one. If changes are made to four fields and each change is committed separately, the result increments by four.

Get(RecordNumber)

Purpose

Returns the number of the current record in the current found set.

Format

`Get(RecordNumber)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The value returned is determined by the relative place of the record in the found set, and it changes depending on the find criteria and the sort order.

Notes

- To return a value that uniquely and permanently identifies a record in this table, use `Get(RecordID)`.
- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **3** when the current record is the third record in a found set.

Get(RecordOpenCount)

Purpose

Returns the total number of open records in the current found set that haven't been saved.

Format

```
Get(RecordOpenCount)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **4** if there are four open records in the current found set that haven't been saved.

Get(RecordOpenState)

Purpose

Returns a number representing the state of the current record.

Format

```
Get(RecordOpenState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns:

- **0** for a closed or committed record
- **1** for a new record that hasn't been committed
- **2** for a modified record that hasn't been committed

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** if the current record is a new record that hasn't been saved.

Get(RequestCount)

Purpose

Returns the total number of find requests defined for the current table.

Format

`Get(RequestCount)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If there are multiple windows open in the current database file, then results are returned for only the top-most window of the file in which the calculation is defined.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when there are five find requests defined for the current table.

Get(RequestOmitState)

Purpose

Returns a Boolean value representing the state of the **Omit** checkbox in Find mode.

Format

```
Get(RequestOmitState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns **1** if the **Omit** checkbox is selected; otherwise, returns **0**.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** when the **Omit** checkbox is selected in the current find request.

Get(ScreenDepth)

Purpose

Returns the number of bits needed to represent the color or shade of gray of a pixel on the main screen.

Format

`Get(ScreenDepth)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

A value of 8 represents 256 (equal to 2^8) colors or shades of gray.

Notes

- In FileMaker WebDirect, this function is not supported and returns **32**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **32** on a display showing millions (2^{32}) of colors.

Returns **16** on a display showing thousands (2^{16}) of colors.

Returns **4** on a VGA display.

Returns **1** on a black-and-white display.

Get(ScreenHeight)

Purpose

Returns the height, in points, of the screen in which the window of the current file is open.

Format

```
Get(ScreenHeight)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

When the window spans more than one screen, this function uses the screen that contains the largest percentage of the window. If there are multiple windows open in the current database file, each window can have its own screen height value, but results are returned for only the foreground window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **480** when the screen resolution is set to 640 x 480.

Get(ScreenWidth)

Purpose

Returns the width, in points, of the screen in which the window of the current file is open.

Format

```
Get(ScreenWidth)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

When the window spans more than one screen, this function uses the screen that contains the largest percentage of the window. If there are multiple windows open in the current database file, each window can have its own screen width value, but results are returned for only the foreground window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **640** when the screen resolution is set to 640 x 480.

Get(ScriptAnimationState)

Purpose

Indicates whether or not animations are enabled for the current script.

Format

`Get(ScriptAnimationState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 13.0

Description

Returns **0** if animations are disabled for the current script.

Returns **1** if animations are enabled for the current script.

Animations are off by default while a script is running.

Note In FileMaker WebDirect, this function is not supported and returns an empty string.

Examples

Returns **1** when the current script has been set to enable animations using the Set Script Animation script step.

Get(ScriptName)

Purpose

Returns the name of the script currently running (or paused).

Format

```
Get(ScriptName)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Print Report** when the Print Report script is running.

Returns **Update Customer** when the Update Customer script is running.

Get(ScriptParameter)

Purpose

Returns the script parameter passed into the current script.

Format

`Get(ScriptParameter)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Use this function as part of a calculation evaluated within a script.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Print** when "Print" was the value of the parameter passed into the current script.

The following example shows how to pass a return-delimited list as the parameter.

```
ScriptParameter = List ( Customers::First; Customers::Last )
```

`LeftValues ( Get ( ScriptParameter ) ; 1 )` returns **Michael** if Customers::First is "Michael".

The following example shows how to pass named parameters using the `Evaluate`, `Let`, and `Get(ScriptParameter)` functions, allowing access only to variable `"a"` (the example returns **6**):

```
ScriptParameter = "a = 5; b = 10"
```

```
Evaluate("Let ( [" & Get(ScriptParameter) & "]; a + 1 )" )
```

The following example shows how to pass named parameters, allowing access to both variable `"a"` and `"b"`. The simplified first parameter makes the second parameter more complex (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
```

```
Evaluate("Let ( [" & Get(ScriptParameter) & "]; a + 1 & \", \" & b + 2 )" )
```

The following example shows how to pass named parameters, while keeping the ability to check the syntax of the second parameter of the `Let` function (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
```

```
Let( [a = Evaluate("Let( [" & Get(ScriptParameter) & "]; a )");
b = Evaluate("Let( [" & Get(ScriptParameter) & "]; b )")];
a + 1 & ", " & b + 2 )
```

Get(ScriptResult)

Purpose

Returns the script result from a performed subscript.

Format

```
Get(ScriptResult)
```

Parameters

None

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 8.0

Description

Use this function as part of a calculation evaluated within a script. If a subscript doesn't return a result, then the content of the script result will be empty.

Examples

In the following example, the Find Customers script returns the results of a find request when it is called from the Do Reports script. Script Find Customers uses the optional script result of the Exit Script script step. Script Do Reports then uses `Get(ScriptResult)` to determine what other script steps should be performed based on the returned result stored in `Get(ScriptResult)`.

Find Customers

```
Set Error Capture [On]
Perform Find [Restore]
New Record/Request
Exit Script [Result: Get(FoundCount) < 10]
```

Do Reports

```
Perform Script [Find Customers]
If [Get(ScriptResult) = 0]
   Show Custom Dialog [“You have created 10 records already.”]
End If
```

Get(SortState)

Purpose

Returns a value representing the current sort state.

Format

```
Get(SortState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** if the records in the active table are not sorted
- **1** if the records in the active table are sorted
- **2** if the records in the active table are partially sorted (semi-sorted)

Each window has its own sort state.

Notes

- When records are imported from another file to a previously found and sorted set, the records in a sorted set may exist in a semi-sorted state. To include the imported records in the sort order, sort the found set after importing.
- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** when the records in the active table are sorted.

Get(StatusAreaState)

Purpose

Returns a number indicating the current status toolbar state.

Format

`Get(StatusAreaState)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** (zero) if the status toolbar is hidden
- **1** if the status toolbar is visible
- **2** if the status toolbar is visible and locked
- **3** if the status toolbar is hidden and locked

If there are multiple windows open on the currently active database file, then results are returned for only the active window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1**, when the current status toolbar is visible.

Get(SystemDrive)

Purpose

Returns the drive letter (Windows) or volume name (OS X) where the currently running operating system is located.

Format

```
Get(SystemDrive)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In FileMaker WebDirect, this function is not supported and returns an empty string.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **/C:/** in Windows when the operating system is on the C: drive.

Returns **/DriveName/** in OS X when the operating system is on a volume named DriveName.

Get(SystemIPAddress)

Purpose

Returns a list of the IP addresses of all computers connected to an active NIC (Network Interface Controller) card.

Format

`Get(SystemIPAddress)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

IP addresses are separated by carriage returns.

In FileMaker WebDirect, Get(SystemIPAddress) returns the IP address of the interface used to connect to the host.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Suppose a machine has the following active physical interfaces:

- an Ethernet card not connected to a network with an IP address of 10.10.10.10
- a Wi-Fi interface with an IP address of 192.168.1.1
- a VPN connection with an IP address of 172.172.172.172

The function returns:

192.168.1.1

172.172.172.172

Suppose a machine has the following active physical interfaces:

- an Ethernet card not connected to a network with an IP address of 2001::10
- a Wi-Fi interface with an IP address of 3FFE:FFFF:101::230:6EFF:FE04:D9FF/48
- a VPN connection with an IP address of 2001:0DB8:85A3:08D3:1319:8A2E:0370:7334

The function returns:

3FFE:FFFF:101::230:6EFF:FE04:D9FF/48

2001:0DB8:85A3:08D3:1319:8A2E:0370:7334

Get(SystemLanguage)

Purpose

Returns the language currently set on the current system.

Format

`Get(SystemLanguage)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Get(SystemLanguage) is evaluated on the system preference that is set for Region on the Formats tab. The text that is returned is in the English language.

For hosted databases, Get(SystemLanguage) returns the client's current system language.

Note In FileMaker WebDirect, Get(SystemLanguage) returns the web browser's current language.

Examples

Returns **Japanese** when Japanese is the current format for the region.

Get(SystemNICAddress)

Purpose

Returns a list of the hardware addresses of all NIC (Network Interface Controller) cards connected to the computer.

Format

`Get(SystemNICAddress)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Values in the list returned by this function are separated by carriage returns. The address consists of 6 bytes displayed in hexadecimal separated by colons. In Windows, find this address by typing the command "`ipconfig /All`" in a DOS window. In OS X, find this address under **Network Overview** in the **System Profile** tab under Applications/Utilities/Apple System Profiler.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **00:07:34:4e:c2:0d**, for example.

Get(SystemPlatform)

Purpose

Returns a number indicating the current platform.

Format

`Get(SystemPlatform)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **1** if the current platform is Intel-based Macs
- **-2** if the platform is Windows
- **3** if the platform is iOS
- **4** if the platform is FileMaker WebDirect

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

`Get(SystemPlatform)` returns **-2** when the current platform is a Windows platform.

`Abs(Get(SystemPlatform))` returns **1** when the current platform is OS X.

Get(SystemVersion)

Purpose

Returns the version of the current operating system.

Format

`Get(SystemVersion)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **6.1** for Windows 7
- **6.2** for Windows 8
- **10.7** for OS X version 10.7
- **10.8** for OS X version 10.8
- ***<operating system or device> <web browser> <browser version>*** for FileMaker WebDirect

  For operating system or device, returns:

  - **Win** for a Windows operating system
  - **Mac** for an OS X operating system
  - **Linux** for a Linux operating system
  - **iPad** for an iPad
  - **iPhone** for an iPhone
  - **iPod** for an iPod
  - **Other** for an unknown operating system or device

  For web browser, returns:

  - **Safari** for the Safari browser
  - **IE** for the Internet Explorer browser
  - **Chrome** for the Chrome browser
  - **Other** for an unknown browser

  For browser version, returns the version of the web browser accessing FileMaker WebDirect.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **10.8** when the current operating system is OS X version 10.8.

Returns **iPad Safari 6.0** for an iPad using FileMaker WebDirect in Safari version 6.0.

Returns **Win Chrome 25** for a Windows computer using FileMaker WebDirect in Chrome version 25.

Get(TemporaryPath)

Purpose

Returns the path to the temporary folder that FileMaker Pro uses for the current user, or the path that FileMaker Server uses on the system.

Format

`Get(TemporaryPath)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 9.0

Description

The temporary folder name begins with *S*, followed by a number representing the session of the database engine during which the operation took place. Because your operating system controls the location of temporary files, the exact path returned may be different from the examples shown. The actual path returned also depends on which product (FileMaker Pro or FileMaker Server) is executing the function.

In FileMaker Pro, the temporary folder and any files placed in it are deleted when FileMaker Pro is terminated. In FileMaker Server, each schedule runs in its own session; once the schedule is completed, the session terminates and the temporary folder is deleted.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

In Windows, returns:

/%HomeDrive%/Documents and Settings/{user id}/Local Settings/Temp/S<*n*> or

/%UserProfile%/AppData/Local/Temp/S<*n*> or

/%HomeDrive%/WINDOWS/Temp/S<*n*> (Windows XP)

where **%HomeDrive%** is an environment variable that returns the name of your home drive on your hard disk.

%UserProfile% is an environment variable that points to the directory where the profile of the current user is located.

S<*n*> is the name of the folder in which the temporary files are placed (for example, S1); **<*n*>** is a number representing the database engine session during which the operation took place.

In OS X, returns:

/<DriveName>/private/var/folders/<2 characters>/<20 characters>++++TI/Cleanup at Startup/S<*n*>/

where **DriveName** is the name of your hard disk.

S<*n*> is the name of the folder in which the temporary files are placed (for example, S1); **<*n*>** is a number representing the database engine session during which the operation took place.

The location may vary due to different variables on the OS, but should follow a similar pattern.

Get(TextRulerVisible)

Purpose

Returns a Boolean value representing whether or not the text ruler is visible.

Format

```
Get(TextRulerVisible)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns **1** if the text ruler is displayed; otherwise, returns **0**.

Notes

- In FileMaker WebDirect, this function is not supported and returns **0**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** when the text ruler is visible.

Get(TotalRecordCount)

Purpose

Returns the total number of records in the current table.

Format

```
Get(TotalRecordCount)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If the current calculation is stored and you specify its context, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **876** when there are 876 records in the current table.

Get(TriggerCurrentPanel)

Purpose

Returns the index and the object name of the current tab panel or slide panel (the panel to be switched from) when the OnPanelSwitch script trigger is activated.

Format

`Get(TriggerCurrentPanel)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

Use with the Get(TriggerTargetPanel) function. Returns an index value, starting from 1, when running a script triggered by the OnPanelSwitch script trigger, and the object name assigned to the tab or slide panel. Returns 0 if the panel is invalid or if Get(TriggerCurrentPanel) is not used with the OnPanelSwitch script trigger.

You can use the GetValue function to extract the value you want to use from the index value returned by Get(TriggerCurrentPanel).

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

When the tab or slide panel to be switched from is panel number 1, named "Category," `Get(TriggerCurrentPanel)` returns:

```
1
Category
```

Get(TriggerGestureInfo)

Purpose

In FileMaker Go, returns details about the gesture that activated an OnGestureTap trigger.

Format

```
Get(TriggerGestureInfo)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 13.0

Description

In FileMaker Go, returns a list containing these items:

- the string **tap**, indicating the script was started by an OnGestureTap trigger
- a value indicating the tap count
- a value indicating how many fingers were used to make the tap
- the x coordinate in the document where the gesture occurred
- the y coordinate in the document where the gesture occurred

For multi-finger gestures, the coordinates returned indicate the center point of the gesture. The y coordinate has the header (if present) at 0. The records are listed under the header. Then the footer follows the displayed records. In Form View, only one record is displayed. In List View and Table View, all of the records in the current found set are displayed.

This function supports the following gestures:

- Single-tap with one, two, or three fingers
- Double-tap with one finger

Returns an empty string if this function is executed when no OnGestureTap trigger has been activated.

Note In other products in the FileMaker product line, this function is not supported and returns an empty string.

Examples

When a script is triggered by a three-finger single tap, and the gesture occurred at the coordinates (400, 600), this function returns:

tap
1
3
400
600

Get(TriggerKeystroke)

Purpose

Returns a string containing the characters that activated an OnObjectKeystroke or OnLayoutKeystroke script trigger. Multiple characters may be returned when the input comes from an input method editor (IME).

Format

```
Get(TriggerKeystroke)
```

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 10.0

Description

Returns a value when running a script triggered by an OnObjectKeystroke or OnLayoutKeystroke script trigger or running a script called from the triggered script; otherwise returns an empty string.

Examples

The following code displays the text **Processing input...** when a carriage return is entered:

```
If [ Code ( Get(TriggerKeystroke) ) = 13 ]
   Show Custom Dialog [“Processing input...”]
End If
```

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Get(TriggerModifierKeys)

Purpose

Returns the state of the keyboard modifier keys as they were when a script trigger was activated.

Format

```
Get(TriggerModifierKeys)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 10.0

Description

Returns a value only when called from a script activated by a script trigger or from a sub-script called from the triggered script; otherwise returns an empty string.

- See the Get(ActiveModifierKeys) function for a description of the values assigned to the keyboard modifier keys.
- See the Code function for a list of navigational keys and the codes returned to a script activated by this trigger.

Time might elapse between when the keys that activated a script trigger are pressed and the script asks for information on the modifier keys. Use Get(TriggerKeystroke) and Get(TriggerModifierKeys) to capture the keys that were active when the script trigger was activated. Use Get(ActiveModifierKeys) to capture any current keys being pressed.

Notes

- Windows: Alt and Ctrl key combinations do not activate script triggers.
- OS X: Command key combinations do not activate script triggers.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

- The following example will only display a custom dialog box when lowercase “a” is entered:

  ```
  If [Get(TriggerKeystroke)=“a” and Get(TriggerModifierKeys)=0]
     Show Custom Dialog [“You entered \”a\”.”]
  End If
  ```

- The value **9** is returned when Shift-Option is pressed on a computer running OS X. If the Option and Shift keys are pressed on a Mac when a script is triggered, Get(TriggerModifierKeys) returns **9**, regardless of which modifier keys have been pressed between when the trigger was activated and when the script runs.

Get(TriggerTargetPanel)

Purpose

Returns the index and the object name of the target tab panel or slide panel (the panel to be switched to) when the OnPanelSwitch script trigger is activated.

Format

`Get(TriggerTargetPanel)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

Use with the Get(TriggerCurrentPanel) function. Returns an index value, starting from 1, when running a script triggered by the OnPanelSwitch script trigger, and the object name assigned to the tab or slide panel. Returns 0 if the panel is invalid or if Get(TriggerTargetPanel) is not used with the OnPanelSwitch script trigger.

You can use the GetValue function to extract the value you want to use from the index value returned by Get(TriggerTargetPanel).

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

When the tab or slide panel to be switched to is number 2, named “Products,” `Get(TriggerTargetPanel)` returns:

```
2
Products
```

Get(UserCount)

Purpose

Returns the number of clients currently accessing the file.

Format

```
Get(UserCount)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **1** if FileMaker network sharing is turned off
- **1** + the number of clients if FileMaker network sharing is turned on

This function does not count clients accessing the database file via ODBC or JDBC.

Notes

- If you specify the context for the current calculation, this function will be evaluated based on that context; otherwise, it will be evaluated based on the context of the current window.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **5** when there are 4 clients accessing the database file.

Get(UserName)

Purpose

Returns the name of the FileMaker Pro user, as specified in the **General** tab of the Preferences dialog box.

Format

`Get(UserName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

The returned name is user-specified.

Important For greater security, use `Get(AccountName)` to track and manage user access: a user cannot change the account name used to log in to a database file.

Notes

- In FileMaker WebDirect, this function returns the name of the account that logged into the current session.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **Sharon Lloyd** when Sharon Lloyd is the current user.

Get(UseSystemFormatsState)

Purpose

Returns a Boolean value representing the state of the **Use System Formats** command in the Format menu.

Format

```
Get(UseSystemFormatsState)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns **1** if **Use System Formats** is on; otherwise, returns **0**.

Notes

- In FileMaker WebDirect, this function is not supported and returns **0**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** when **Use System Formats** is on.

Get(UUID)

Purpose

Returns text representing a Universally Unique Identifier (UUID).

Format

`Get(UUID)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

Returns a unique 16-byte (128-bit) string. For example, you can use this function to generate a unique ID of a record.

For unstored calculations, returns a new string each time Get(UUID) is evaluated.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Stored calculation: In a calculation field, specify the calculation `Get(UUID)`. Every new record in the calculation field has a unique ID such as E47E7AE0-5CF0-FF45-B3AD-C12B3E765CD5.

Unstored calculation: In a calculation field, specify the calculation `Get(UUID)`. For **Storage Options**, select **Do not store calculation results**. Every time a record is accessed, a new string is generated.

Get(WindowContentHeight)

Purpose

Returns the height, in points, of the FileMaker Pro content area.

Format

`Get(WindowContentHeight)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The content area depends on the current size of the active window but doesn't include the title bar, scroll bars, zoom controls, and page margins. The content area is the space inside these controls. It does not include the status toolbar if it is currently showing.

Notes

- In FileMaker WebDirect, the content area includes the menu bar, status toolbar, scroll bars, and footer area.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **400** in OS X when the current window height is 437 and the status toolbar isn't showing.

The example below combines `Get(WindowContentHeight)` with `Get(WindowHeight)` to determine the height of the title bar and horizontal scroll bar:

`Get(WindowHeight) - Get(WindowContentHeight)` returns **37** in OS X when the window height is 437 and the status toolbar isn't showing.

Get(WindowContentWidth)

Purpose

Returns the width, in points, of the FileMaker Pro content area.

Format

`Get(WindowContentWidth)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The content area depends on the current size of the active window but doesn't include the title bar, scroll bars, zoom controls, or page margins. The content area is the space inside these controls.

Notes

- In FileMaker WebDirect, the content area includes the menu bar, status toolbar, scroll bars, and footer area.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **400** in OS X when the current window width is 415.

The example below combines `Get(WindowContentWidth)` with `Get(WindowWidth)` to determine the width of the vertical scroll bar:

`Get(WindowWidth) - Get(WindowContentWidth)` returns **15** in OS X when the window width is 415.

Get(WindowDesktopHeight)

Purpose

Returns the height, in points, of the desktop space.

Format

```
Get(WindowDesktopHeight)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

In Windows, the desktop space is the area inside the MDI window (sometimes referred to as the client area). This doesn't include any virtual space available through the scrolling of the MDI window.

In OS X, the desktop space is the area on the monitor in which the active window is located excluding menu bars.

In FileMaker WebDirect, the desktop space is the area on the main monitor. In OS X, the main monitor is where the menu bar is located. In Windows, the main monitor is where the taskbar is located.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **956** in Windows when there is a single monitor and its MDI is set to 1280 x 1024.

Returns **1178** in OS X when there is a single monitor and its resolution is set to 1900 x 1200.

Get(WindowDesktopWidth)

Purpose

Returns the width, in points, of the desktop space.

Format

```
Get(WindowDesktopWidth)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

In Windows, the desktop space is the space inside the MDI window (sometimes referred to as the client area).

In OS X, the desktop space is the area on the monitor in which the active window is located excluding menu bars.

In FileMaker WebDirect, the desktop space is the area on the main monitor. In OS X, the main monitor is where the menu bar is located. In Windows, the main monitor is where the taskbar is located.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **450** in Windows when there is a single monitor and its MDI is set to 500 x 450.

Returns **600** in OS X when there is a single monitor and its resolution is set to 800 x 600.

Get(WindowHeight)

Purpose

Returns the height, in points, of the window on which the script is acting (not necessarily the foreground window).

Format

`Get(WindowHeight)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The height of the window is calculated from the top to bottom outer edges of the window. This position doesn’t include shadows or other effects applied to windows.

In FileMaker WebDirect, the window height does not include menus or toolbars that are part of the web browser.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

`Get(WindowHeight)` returns **300** when the current window’s height is 300 points.

Get(WindowLeft)

Purpose

Returns the horizontal distance, in points, of the outer edge of the window on which the script is acting (not necessarily the foreground window) relative to the left-most edge of the screen.

Format

```
Get(WindowLeft)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The origin of the reference coordinate system is at the left-most corner below the menu bar. A negative value indicates the portion of the left side of the window that is hidden.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **52** when the outer edge of the active window is 52 points from the left edge of the screen.

Returns **0** when the active window is 0 points from the left edge of the screen.

Get(WindowMode)

Purpose

Returns a number representing the mode FileMaker Pro is in at the time the function is evaluated.

Format

```
Get(WindowMode)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns:

- **0** for Browse mode
- **1** for Find mode
- **2** for Preview mode
- **3** if printing is in progress
- **4** (FileMaker Pro Advanced) if evaluating the function from the Data Viewer and the current window is in Layout mode

If a script using this function runs while the file is in Layout mode, FileMaker Pro switches to Browse mode and returns **0**. If there are multiple windows open in the current database file, each window can have its own window mode value, but results are returned for only the foreground window.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **2** if the file is in Preview mode when the function is evaluated.

Get(WindowName)

Purpose

Returns the name of the window on which the script is acting (not necessarily the foreground window).

Format

`Get(WindowName)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Returns an empty string if there is no window.

Notes

- You can set the window name with the Set Window Title script step.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

There are two windows, Teachers and Students, displaying the same layout that includes an unstored calculation Calc containing `Get(WindowName)`. **Teachers** is returned when the Teachers window is refreshed, and **Students** is returned when the Students window is refreshed.

Get(WindowOrientation)

Purpose

Returns a value indicating the orientation of the window on which the script is acting (not necessarily the foreground window).

Format

`Get(WindowOrientation)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 13.0

Description

Returns:

- **-2** for landscape left
- **-1** for landscape right
- **0** for square (FileMaker Pro and FileMaker WebDirect only)
- **1** for portrait
- **2** for portrait upside down

Examples

If the window that the current script is acting on is in portrait orientation, `Get(WindowOrientation)` returns **1**.

You have a calculation field named Orientation that uses `Get(WindowOrientation)` to return a value as listed above. You have another calculation field that references the Orientation field and uses the If function, which returns **Portrait** if the Orientation field returns a value greater than 0 and returns **Landscape** if the Orientation field returns a value less than 0:

```
If(Orientation > 0;”Portrait”;”Landscape”)
```

Get(WindowStyle)

Purpose

Returns the style of the window on which the script is acting.

Format

`Get(WindowStyle)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 12.0

Description

Returns:

- **0** (zero) if the window is a document window
- **1** if the window is a floating document window
- **2** if the window is a dialog window

Examples

Returns **0** if the current window is a document window when the function is evaluated.

Returns **1** if the current window is a floating document window when the function is evaluated.

Returns **2** if the current window is a dialog window when the function is evaluated.

Get(WindowTop)

Purpose

Returns the vertical distance, in points, of the outer edge of the window on which the script is acting (not necessarily the foreground window) relative to the bottom edge of the menu bar.

Format

```
Get(WindowTop)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The origin of the reference coordinate system is at the left-most corner below the menu bar. A negative value indicates the portion of the top part of the window that is hidden behind the menu bar.

Notes

- In FileMaker WebDirect, this function is not supported and returns an empty string.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **52** when the outer edge of the active window is 52 points from the menu bar.

Returns **0** when the outer edge of the active window just touches the menu bar.

Get(WindowVisible)

Purpose

Returns a number representing whether or not the current window is visible.

Format

```
Get(WindowVisible)
```

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The current window is the window on which the script is acting (not necessarily the foreground window). Returns **1** if the window is visible. Returns **0** if the window is hidden using the **Hide Window** command. The window can be located outside of the visible screen space and still return **1**.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **1** when the current window is physically visible.

Returns **0** when the current window has been hidden using the Hide Window command in FileMaker Pro.

Get(WindowWidth)

Purpose

Returns the width, in points, of the window on which the script is acting (not necessarily the foreground window).

Format

`Get(WindowWidth)`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

The width of the window is calculated from the left-most to right-most outer edge of the window. This position doesn't include shadows or other effects applied to windows.

Note For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **300** when the current window is 300 points wide.

Get(WindowZoomLevel)

Purpose

Returns the zoom percentage of the current window.

Format

`Get(WindowZoomLevel)`

Parameters

None

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

In Windows, an asterisk appears next to the zoom percentage when **Enlarge window contents to improve readability** is selected in the **General** tab of the Preferences dialog box.

Notes

- In FileMaker WebDirect, this function is not supported and returns **100**.
- For information on how functions evaluate differently on the host versus the client, search the FileMaker Knowledge Base available at http://help.filemaker.com.

Examples

Returns **200** when the current window’s zoom percentage is set to 200.

Returns **200*** in Windows when the current window’s zoom percentage is set to 200 and **Enlarge window contents to improve readability** is selected.

Logical functions

Logical functions test for a condition to evaluate it as true or false. This is known as a Boolean value. If the condition is true, FileMaker Pro returns a **1**; if the condition is false, FileMaker Pro returns a **0**. You can use the keywords `True` and `False` with logical functions and operators when a Boolean value is needed. Keyword `True` returns **1** and keyword `False` returns **0**.

Logical functions can also evaluate parameters such as text or arithmetic operations that do not make a true or false statement, or in the case of the `GetField` function, return the contents of another field.

Click a function name for details.

This function	Returns
Case	One of several possible results based on a series of tests.
Choose	One result value, according to the integer value of a specified test.
Evaluate	Evaluates the specified expression as a calculation.
EvaluationError	An error code, if any, from the specified expression.
ExecuteSQL	Executes a specified SQL SELECT statement within a FileMaker Pro database.
GetAsBoolean	**1** if data converts to a non-zero numeric value; **0** if the specified data has a value of 0 or is empty.
GetField	The contents of the referenced field.
GetFieldName	The fully qualified name of a field reference.
GetLayoutObjectAttribute	The requested layout object attributes from the calculation's active window.
GetNthRecord	The contents of the referenced field from the requested record number.
If	One of two possible results depending on the value of the specified test.
IsEmpty	**1** if the specified field is empty, if the related field, related table, relationship, or file is missing, or if some other error occurs; otherwise, returns **0**.
IsValid	**0** when a record contains an invalid value because of a field type mismatch (text in a date field, for example).
IsValidExpression	**1** if the syntax of the specified expression is correct.
Let	Sets variable to the result of value for the duration of the specified expression.
Lookup	The value specified in the `sourceField` parameter using the relationships in the relationships graph.
LookupNext	The value specified in the `sourceField` parameter using the relationships in the relationships graph.
Self	The content of the object in which the calculation is defined; otherwise, returns an empty string.

Case

Purpose

Returns one of several possible `results` based on a series of `tests`.

Format

`Case(test1;result1{;test2;result2;...;defaultResult})`

Parameters

`test` - any text or numeric expression.

`result` - result corresponding to the expression.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

`Case` evaluates each test expression in order, and when a True expression is found, returns the value specified in `result` for that expression.

You can include a default result at the end of the parameter list. If none of the expressions evaluated return True, the `Case` function returns the value specified for `defaultResult`. If no default result is supplied, `Case` returns an empty result.

Examples

`Case(Score >= 90;“Excellent”;Score > 50;“Satisfactory”;“Needs Improvement”)` displays **Excellent** when the score is 90 or above, **Satisfactory** when the score is between 50 and 90, and **Needs Improvement** for any other score.

`Case(Shipment Method=”Ground”;2;Shipment Method=”Air”;10)` returns **2** when the Shipment Method field contains Ground, and returns **10** when the Shipment Method field contains Air.

Choose

Purpose

Returns one `result` value, according to the integer value of `test`.

Format

```
Choose(test;result0{;result1;result2...})
```

Parameters

`test` - Any integer calculation. The calculation result of test must be a number that indexes into the list that follows. Because the index is a 0-based index, the test result must be 0 to access the first result.

`result` - one or more results.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

FileMaker Pro evaluates `test` to obtain an index number, which is used to choose the corresponding ordinal result.

Because `Choose` is a 0-based list, the first item on the list is indexed 0 and the second item on the list is indexed 1. For example, if test evaluates to 2, then result2 is chosen.

Examples

```
Choose(Rating;“Not Applicable”;“Good”;“Fair”;“Poor”)
```

Rating is a number field that is empty or holds a value. If Rating is empty or 0, the `Choose` function returns **Not Applicable**. If Rating is 1, the result is **Good**. If Rating is 2, the result is **Fair**, and if it is 3, the result is **Poor**. If Rating contains a value that does not map to one of the result parameters, the `Choose` function returns nothing.

Evaluate

Purpose

Evaluates `expression` as a calculation.

Format

`Evaluate(expression{;[field1;field2;field3;...]})`

Parameters

`expression` - any text expression or text field.

`fields` - a list of fields that this function is dependent on. When these fields are modified, the calculation will update its result.

Parameters in curly braces { } are optional. Notice that the optional field list is enclosed in square brackets [ ].

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 7.0

Description

The optional `fields` parameter is a list of fields this calculation is dependent on. If a necessary field isn't listed, modifying that dependent field won't update the result of the calculation.

Examples

`Evaluate(TextField)` returns **4** when TextField contains 2 + 2.

`Evaluate("textfield")` returns **2 + 2** when textfield contains 2 + 2.

`Evaluate(GetField("textfield"))` returns **4** when textfield contains 2 + 2.

`Evaluate(TextField;[Amount])` returns **.80** when TextField contains .08 * Amount and the Amount field contains 10.00.

`Evaluate(“Let(TaxRate=.05;“& Tax Rate Calculation &“)”)` returns **.50** when the field Tax Rate Calculation contains SubTotal * TaxRate where SubTotal is a numeric field that contains 10.00.

The following example shows how to pass named parameters using the `Evaluate`, `Let`, and `Get(ScriptParameter)` functions, allowing access only to variable “a” (the example returns **6**):

```
ScriptParameter = "a = 5; b = 10"
Evaluate("Let ( [" & Get(ScriptParameter) & "]; a + 1 )")
```

The following example shows how to pass named parameters, allowing access to both variable “a” and “b”. The simplified first parameter makes the second parameter more complex (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
Evaluate("Let ( [" & Get(ScriptParameter) & "]; a + 1 & \", \" & b + 2 )")
```

The following example shows how to pass named parameters, while keeping the ability to check the syntax of the second parameter of the `Let` function (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
```

```
Let(  [a = Evaluate("Let( [" & Get(ScriptParameter) & "]; a )"),
         b = Evaluate("Let( [" & Get(ScriptParameter) & "]; b )")]; a + 1
& ", " & b + 2 )
```

Note The `Evaluate` function evaluates an expression, including field values to be evaluated as a calculation formula. It also allows you to specify field dependencies so that a calculation using the evaluation function can be triggered due to changes in other fields of the same record. This function evaluates user-defined formulas. For example, you can create a formula in the Total field that computes state tax:

```
Evaluate(StateTaxFormula) + ShippingCost
```

where the StateTaxFormula field contains:

```
SubTotal * 1.0875
```

and the SubTotal field contains the subtotal before tax and shipping.

The `Evaluate` function has an optional second parameter, which is a field the calculation is dependent on. When the dependent field contents change, FileMaker Pro re-evaluates the calculation. In the following example, the Total calculation will be re-evaluated when SubTotal changes:

```
Evaluate(StateTaxFormula; SubTotal) + ShippingCost
```

The dependent parameter can also be useful in other cases. For example,

```
Evaluate(“Get(CurrentTimeStamp)”; [FieldB; FieldC])
```

will store a timestamp in the calculation field whenever FieldB or FieldC changes.

EvaluationError

Purpose

Returns an error code, if any, from `expression`.

Format

`EvaluationError(expression)`

Parameters

`expression` - any calculation expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

There are two types of errors: syntax and runtime. A syntax error indicates an invalid calculation. A runtime error, such as Field missing or Record missing, occurs when the calculation currently being run is valid but cannot properly execute. See FileMaker Pro error codes for a list of error codes and messages.

Note The `EvaluationError` function must enclose the `Evaluate` function to return any syntax errors.

Examples

`EvaluationError(calculationField)` returns **102** (Field Missing) when `calculationField` contains `total + 1` and the field total has been deleted or renamed.

`EvaluationError(Evaluate(calculationField))` returns **1207** (Unbalanced Parenthesis) when `calculationField` contains `abs(-1` with no closing parenthesis.

ExecuteSQL

Purpose

Executes an SQL query statement for the specified table occurrence within a FileMaker Pro database.

Format

`ExecuteSQL(sqlQuery; fieldSeparator; rowSeparator {;arguments...})`

Parameters

`sqlQuery` - an SQL SELECT statement. The statement can include a Union clause that combines the results of two queries. The statement can contain programmatically generated SQL (dynamic parameters) that indicate where optional arguments are to be used in the query. Use the question mark character (?) to specify a dynamic parameter.

`fieldSeparator` - the character string used as a separator between fields in the result. If an empty string is specified, the separator is a comma. The field separator is not displayed after the last field in the result.

`rowSeparator` - the character string used as a separator between records in the result. If an empty string is specified, the separator is a carriage return. The row separator is not displayed after the last row in the result.

`arguments`- one or more expressions that are evaluated and used as values for the dynamic parameters in the query statement.

Data type returned

text

Originated in

FileMaker Pro 12.0

Description

ExecuteSQL enables you to execute SQL SELECT statements containing dynamic parameters to safely query FileMaker Pro databases in order to avoid security vulnerabilities through injection attacks.

ExecuteSQL does not recognize relationships created in FileMaker Pro, which gives you flexibility to define relationships in SQL statements and retrieve data from any table, independent of the layout context.

ExecuteSQL cannot be used with SQL statements that modify data or the database schema (such as the Insert Into or Delete Table commands).

If an error occurs during query parsing or execution, FileMaker Pro returns **?**.

Notes

- To apply the correct formatting to dates in an SQL query, use the DATE statement. If you do not use the DATE statement, ExecuteSQL treats dates as literal strings.
- FileMaker Pro returns date, time, and number data in Unicode/SQL format, not in the locale of the operating system or the file.

- ExecuteSQL accepts only the SQL-92 syntax ISO date and time formats with no braces. ExecuteSQL does not accept the ODBC/JDBC format date, time, and timestamp constants in braces.
- For more details about SELECT statement syntax, supported SQL statements, expressions, and Catalog functions, see the FILEMAKER ODBC AND JDBC GUIDE and the FILEMAKER SQL REFERENCE.

Examples

Suppose a database contains two tables, Employees and Salaries, which are related through the EmpID field.

Employees

EmpID	Last Name	Department
1	Smith	Development
2	Ogawa	Development
3	Durand	Quality Assurance
4	Garcia	Quality Assurance
5	Mehmet	Documentation
6	Ferrini	Marketing

Salaries

EmpID	Salary
1	98000
2	87000
3	86000
4	90000
5	89000
6	121000

Note The Salaries::Salary field is a number field.

You want to add a field to the Employees table that displays the percentage of an employee's salary relative to the total salaries in a department. Though you could use a calculation in FileMaker Pro to generate this value, you can use the ExecuteSQL function to specify this query using dynamic parameters.

Define a calculation field in the Employees table, then use the ExecuteSQL function to specify the following query statement:

```
100 * Salaries::Salary /  ExecuteSQL ( “select sum ( S.salary ) from
Employees E join Salaries S on E.EmpID = S.EmpID where E.Department = ?”;
“”; “”; Employees::Department )
```

GetAsBoolean

Purpose

Returns **1** if `data` converts to a non-zero numeric value or if a container field contains data; returns **0** if `data` has a numeric value of **0** or is empty.

Format

`GetAsBoolean(data)`

Parameters

`data` - any text, number, date, time, timestamp or container expression, or a field containing text, a number, date, time, timestamp or container

Data type returned

number

Originated in

FileMaker Pro 8.0

Description

Returns a Boolean value.

Examples

`GetAsBoolean("")` returns **0**.

`GetAsBoolean("Some text here.")` returns **0**.

`GetAsBoolean(Container Field)` returns **1** when the field named Container Field contains data, or returns **0** when Container Field is empty.

GetField

Purpose

Returns the contents of `fieldName`.

Format

`GetField(fieldName)`

Parameters

`fieldName` - any text expression or text field that refers to a field's name

Important See Design functions for information about literal text parameters.

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to get the contents of `fieldName`, or in any function that uses a field, such as `NPV`, `GetSummary`, `GetRepetition`, or the aggregate functions.

Examples

Suppose you have the fields Arrow and Target. Arrow contains the text string Target, and Target contains the text string Bullseye.

- `GetField(“Arrow”)` returns **Target**. Notice the use of quotation marks around Arrow to indicate the literal string is the `fieldName` parameter.
- `GetField(Arrow)` returns **Bullseye**. Notice the absence of quotation marks to indicate the value stored in the Arrow field is the `fieldName` parameter.

Suppose you have the fields FirstName and LastName. FirstName contains the text string Jane, and LastName contains the text string Public.

- `GetField(“FirstName”)&“ ”&GetField(“LastName”)` returns the text string **Jane Public**.

`GetSummary(GetField(“Field1”), GetField(“Field” & “2”))` performs a summary on the summary field Field1, using a break field of Field2.

GetFieldName

Purpose

Returns the fully qualified name of a field reference.

Format

`GetFieldName(fieldName)`

Parameters

`fieldName` - any field object or evaluation of a text expression that refers to a field's name

Data type returned

text

Originated in

FileMaker Pro 10.0

Description

Use this function to get the fully qualified name of `fieldName` (tableName::fieldName).

Note If you specify the context for the current calculation, this function will be evaluated based on that context. Otherwise, it will be evaluated based on the context of the current window.

Examples

`GetFieldName(x)` returns the name of a field reference passed into a custom function as parameter `x`.

`GetFieldName(Evaluate(<fieldName>))` returns the name of a field based on the data stored in `<fieldName>`.

`GetFieldName(Evaluate(Get(ActiveFieldName)))` returns the fully qualified name of the field that has the focus when executed.

GetLayoutObjectAttribute

Purpose

Returns the specified attributes of the layout object given by `objectName` that is currently active in the calculation.

Format

```
GetLayoutObjectAttribute(objectName;attributeName{;repetitionNumber;
portalRowNumber})
```

Parameters

`objectName` - the name of a named layout object on the current layout.

`attributeName` - the name of a supported attribute (see below).

`repetitionNumber` - the repetition number (for repeating fields).

`portalRowNumber` - the number of the row in the portal.

Note Parameters in curly braces {} are optional.

Data type returned

text

Originated in

FileMaker Pro 8.5

Attributes

`objectType` - returns the object's type as a text literal, in English. Valid return values are: field, text, graphic, line, rectangle, rounded rectangle, oval, group, button group, button, portal, tab panel, slide panel, chart, web viewer, popover, popover button, and unknown.

`hasFocus` - returns **1** (True) if `objectName` is currently active, otherwise returns **0** (False). Objects that can have the focus are fields, portals, tab panels, slide panels, buttons, popover buttons, charts, and groups. Also returns **1** for a portal when a portal row is selected.

`containsFocus` - returns **1** (True) if `objectName` is currently active or if it contains an active object; otherwise returns **0** (False). Objects that can contain the focus are fields, portals, tab panels, slide panels, buttons, popover buttons, popovers, charts, and groups.

`isFrontPanel` - returns **1** (True) if the target object is the tab or slide panel that is in front.

The following object coordinates are given in points relative to the bottom-left corner of the FileMaker menu bar:

`bounds` - returns a list of numeric values, separated by spaces, that describes the placement of the specified object (left-top to right-bottom).

`left` - returns the left edge coordinate of the specified object.

`right` - returns the right edge coordinate of the specified object.

`top` - returns the top edge coordinate of the specified object.

`bottom` - returns the bottom edge coordinate of the specified object.

`width` - returns a number representing the width (in points) of the specified object.

`height` - returns a number representing the height (in points) of the specified object.

`rotation` - returns a number representing the rotation (in degrees) of the specified object.

`startPoint,endPoint` - returns a pair of numeric values (horizontal vertical), separated by spaces, that represent the start point or end point of a line object. Other objects return the top-left point for `startPoint` and the bottom-right point for `endPoint`.

`source` - returns the source description of the specified object as follows. For:

web viewers - returns current URL.

fields - returns the fully qualified field name (table name::field name).

text objects - returns the text (does not return merge fields).

portals - returns the related table name.

graphics - returns image data such as the image filename.

charts - returns the XML description of a chart object.

For all other objects, returns an empty string.

`content` - returns the content of the specified object as follows. For:

web viewers - returns the current content (such as HTML code).

fields - returns the field data formatted using the specified object's properties.

text objects - returns the text (including text from merge fields).

graphics - returns image data such as the name of a file in a container field if the image is stored (in the field or externally), or the reference to the file if the image is unstored.

charts - returns the bitmap representation of a chart object.

For all other objects, returns an empty string.

`enclosingObject` - returns `objectName` of the enclosing layout object. Otherwise, returns an empty string. Only groups, tab panels, slide panels, popover buttons, popovers, and portals can contain other objects.

`containedObjects` - returns a list of named objects contained within `objectName`. Only groups, tab panels, slide panels, popover buttons, popovers, and portals can contain other objects.

`isobjecthidden` - returns **1** (True) if `objectName` is hidden for the current record. Otherwise, returns **0** (False). Returns **1** for objects other than popovers that are to the right of the layout boundary.

Notes

- If objects are set to auto-resize, attributes returned are based on the resized bounds of the object in its current state.
- If objects are located above the status toolbar, negative coordinate values are returned.
- When `repetitionNumber` or `portalRowNumber` is 0, the function behaves as if the parameter was not specified. For `portalRowNumber`, the function returns data from the first portal row. For `repetitionNumber`, the function acts on the first repetition (for returning `content` or `source`) or acts on the entire field as a whole (for returning `bounds`). Both parameters are necessary because you must be able to reference a particular field repetition within a particular portal row.

Examples

`GetLayoutObjectAttribute("CancelButton";"objectType")` returns **button** (if the button was created in a new file or a file after it was converted), returns **button group** (if the button is associated with a button action or script and was converted in a file from a previous version of FileMaker Pro), or returns **text** (if the button isn’t associated with a button action or script and was converted in a file from a previous version of FileMaker Pro).

`GetLayoutObjectAttribute("CancelButton";"bounds")` returns **138 24 391 38 0**.

GetNthRecord

Purpose

Returns the contents of `fieldName` from the provided `recordNumber`.

Format

`GetNthRecord(fieldName;recordNumber)`

Parameters

`fieldName` - any related field or repeating field, or an expression that returns a field or a repeating field

`recordNumber` - the record number from which you want data

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 8.0

Description

The result of `GetNthRecord()` will not be updated when the record referred to by GetNthRecord() is a record other than the one in which the calculation is currently being evaluated.

`GetNthRecord` of the current table returns the Nth record of the found set according to how the current table is sorted.

`GetNthRecord` of a related table returns the Nth record of the related set (relative to the current record), regardless of how the related table (or portal) is sorted.

Examples

`GetNthRecord(First Name;2)` returns the contents of the First Name field for record 2 in the current table.

`GetNthRecord(First Name;Get(RecordNumber)+ 1)` returns the contents of the First Name field for the next record in the current table.

`GetNthRecord(Contacts::First Name;2)` returns the contents of the First Name field for record 2 in the Contacts table.

`GetNthRecord(Contacts::Has Repetitions[2];2)` returns the contents of the second repetition of the Has Repetitions field for record 2 in the Contacts table.

If

Purpose

Returns one of two possible results (`result1` or `result2`) depending on the value of `test`.

Format

`If(test;result1;result2)`

Parameters

`test` - any numeric value or logical expression

`result1` - expression or field name

`result2` - expression or field name

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

If `test` is True (any non-zero numeric result), FileMaker Pro returns `result1`. If `test` is False (0), `result2` is returned. `Test` must be an expression that returns either a numeric or Boolean (True, False) result.

Notes

- If you have more than two possible results, consider using the `Case` function.
- By default, if `test` refers to a field that doesn't yet contain a value, `If` returns an empty result. To override this functionality, deselect the **Do not evaluate if all referenced fields are empty** checkbox.

Examples

`If(Country = "USA";"US Tech Support";"International Tech Support")` returns **International Tech Support**, if the Country field contains France or Japan. Returns **US Tech Support** if the Country field contains USA.

`If(State ="CA";Subtotal * CA Tax Rate;0)` returns the tax if the purchaser is a resident of California; otherwise returns **0**.

IsEmpty

Purpose

Returns True(1) if `field` is empty, if a related field, related table, relationship, or file is missing, or if some other error occurs; otherwise, returns False(0).

Format

`IsEmpty(field)`

Parameters

`field` - any field name, text expression, or numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`IsEmpty(OrderNum)` returns **1** if the OrderNum field is empty.

`If(IsEmpty(LastName);“Invalid record”;“”)` displays **Invalid Record** if the LastName field is blank, but displays nothing if there is an entry in LastName.

`IsEmpty(Payments::DatePaid)` returns **1** if, for example, the Payments table has been moved or renamed.

`IsEmpty(“text”)` returns **0**.

IsValid

Purpose

Returns **0** (False) if the data is invalid and **1** (True) if the data is valid.

Format

`IsValid(field)`

Parameters

`field` - any field name

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns **0** (False) if:

- A record contains an invalid value because of a field type mismatch (text in a date field, for example)
- FileMaker Pro cannot locate (temporarily or permanently) the related table in which the referenced field is defined
- A field has been deleted from a related table, and therefore the references to that field in the source table are invalid

Otherwise, it returns **1** (the data is valid).

Examples

`IsValid(Datefield)` returns **0** if there is non-date data in Datefield, for example if text was imported into it.

`IsValid(Amount)` returns **0** if there is only text in the number field Amount.

`IsValid(table::field)` returns **0** if the related table was renamed and the relationship isn't updated with the new filename.

IsValidExpression

Purpose

Returns **1** (True) if `expression` syntax is correct. Returns **0** (False) if `expression` has a syntax error.

Format

`IsValidExpression(expression)`

Parameters

`expression` - any calculation expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Examples

`IsValidExpression(calculationField)` returns **1** (true) if `calculationField` contains `total + 1.`

`IsValidExpression(calculationField)` returns **0** (false) if `calculationField` contains `abs(-1` with no closing parenthesis.

Let

Purpose

Sets `varX` to the result of `expressionX` for the duration of calculation, until the script exits (local variables), or until the file is closed (global variables).

Format

```
Let({[}var1=expression1{;var2=expression2...]};calculation)
```

Parameters

`var` - any variable name, local variable name, or global variable name (see About naming fields for guidelines on naming variables).

`expression` - any calculation expression, field, or constant.

`calculation` - any calculation expression, field, or constant.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 7.0

Description

Multiple variables are allowed when using a list syntax that is enclosed in square brackets [ ] and is separated by semicolons. For example:

```
Let([variable=value;variable2=value2];calculation)
```

The $ symbol references a local variable and two $$ symbols reference a global variable. An optional repetition number appears in square brackets [ ] immediately after the variable name. For example:

```
Let([$variable[repetition]=value;$$variable2=value2]{;calculation} )
```

The `Let` function sets the variables from left to right. You can use previously defined variables (for example, variables that you defined with the Set Variable script step) to define new variable values, and you can nest one `Let` function within another. If you use a previously defined variable within a nested `Let` function, the variable has scope only within the nested function (as if you had defined a completely unique variable). See the City example shown below.

Once defined, local and global variables can be referenced in any calculation within their scope. The scope of global variables is limited to the current file. The scope of local variables is the current script. Local variables defined in a calculation are scoped to the file but are only available when scripts are not running. A local and global variable (or even two local variables in different scripts) can have the same name but they are treated as different variables and store different values.

Examples

`Let(x=5;x*x)` returns **25**.

`Let([x=5;squared=x*x;cubed=squared*x];cubed)` returns **125**.

`Let(City=“Paris”;Let(City=“San Francisco”;City&“-“)&City)` returns **San Francisco - Paris**.

The following example sets a local variable `counter` at repetition `50` with a value of `120`:

`Let($counter[50]=120;$counter[50]*2)` returns **240**.

The following example shows how to pass named parameters using the `Evaluate`, `Let`, and `Get(ScriptParameter)` functions, allowing access only to variable `“a”` (the example returns **6**):

```
ScriptParameter = "a = 5; b = 10"
Evaluate("Let([" & Get(ScriptParameter) & "]; a+1 )" )
```

The following example shows how to pass named parameters, allowing access to both variable `“a”` and variable `“b”`. The simplified first parameter makes the second parameter more complex (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
Evaluate("Let( [" & Get(ScriptParameter) & "]; a+1 & \", \" & b+2 )" )
```

The following example shows how to pass named parameters, while keeping the ability to check the syntax of the second parameter of the `Let` function (the example returns **6, 12**):

```
ScriptParameter = "a = 5; b = 10"
Let([a = Evaluate("Let( [" & Get(ScriptParameter) & "]; a )"),b =
Evaluate("Let( [" & Get(ScriptParameter) & "]; b )")]; a+1 & ", " & b+2 )
```

Lookup

Purpose

Returns the value specified in `sourceField` using the relationships in the relationships graph. The result of the optional `failExpression` will be returned if the lookup fails.

Format

`Lookup(sourceField{;failExpression})`

Parameters

`sourceField` - the field from which the lookup value is taken.

`failExpression` - any expression.

Parameters in curly braces { } are optional.

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 7.0

Description

For this function to access the contents of the source field, the tables containing the source field and calculation field need to be related. Calculations using the `Lookup` function won't be forced to be unstored calculations.

Note `Lookup` returns ? when the related table is an ODBC data source.

Examples

There are two tables, People and Company, in a database file containing the data shown below.

People table

CompanyID	Employee
100	John Smith
200	Peter Wong
300	Sally Anderson

Company table

CompanyID	CompanyName	Code
100	Apple	91234
100	Apple	82345
200	FileMaker	95054

The People and Company tables are related using the number field CompanyID. The calculation `CompanyName = Lookup(Company::CompanyName;“Not found”)` defined in the People table will return **Apple** for the first record, **FileMaker** for the second record, and **Not found** for the third record.

LookupNext

Purpose

Returns the next lower or higher value in `sourcefield` when there isn't a matching related value.

Format

`LookupNext(sourceField;lower/higherFlag)`

Parameters

`sourceField` - the field from which the lookup value is taken

`lower/higherFlag` - the keywords `lower` or `higher` denote whether the value from the next lower/higher matching record must be taken if no related record is found

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 7.0

Description

Returns the value specified in `sourceField` using the relationships in the relationships graph. LookupNext is similar to the Lookup function, except that when the lookup fails, the value from `sourceField` in the lower or higher matching record will be returned, as specified by `lower/higherFlag`.

For this function to access the value in `sourceField`, the tables containing the source field and calculation field need to be related. Calculations using the `LookupNext` function won't be forced to be unstored calculations.

Note `LookupNext` returns ? when the related table is an ODBC data source.

Examples

In this example, you are shipping several items and the cost of shipping is based on weight ranges. Use the LookupNext function to find which shipping rate applies for an item. Use `LookupNext` with the `higher` flag instead of `Lookup` because the weight of an item may not exactly match the maximum weight, therefore we want to find the *next* highest value.

There are two tables, Items and Shipping Costs, in a database file containing data as shown below.

Items table

Item	Weight	Rate Lookup
Lamp	8	
Chair	22	
Desk	60	
Bed	120	

Shipping Costs table

Rate Code	Maximum Weight
A	25
B	50
C	100
D	150

The two tables are related by `Weight` and `Max Weight`. The calculation field `Rate Lookup` is defined as `Rate Lookup = LookupNext(Shipping Costs::Max Weight; Higher)`.

The `Rate Lookup` calculation field will return **25**, **25**, **100**, and **150** for records 1 to 4. `Rate Lookup` can then be used to get the correct rate code (A, A, C, and D respectively).

Item	Weight	Rate Lookup
Lamp	8	25
Chair	22	25
Desk	60	100
Bed	120	150

Self

Purpose

Returns the content of the object in which the calculation is defined.

Format

`Self`

Parameters

`None`

Data type returned

text, number, date, time, timestamp

Originated in

FileMaker Pro 9.0

Description

The `Self` function provides a way for a calculation to reference the object with which it is associated without having to explicitly reference the object.

Use `Self` to create a single calculation formula that can be applied to different objects. The `Self` function is helpful for conditional formatting calculations and tooltip calculations because it returns the content of the layout object when that object has a value. You can also use the `Self` function in field definition calculations (including auto-enter and validation calculations) to return the value of the corresponding field.

Examples

This example formula can be used in an object's conditional formatting panel to set text formatting when the number entered in the field is greater than 10.

`self > 10` returns **1** (True) when applied to a layout field object whose value is greater than 10.

Use the following example in a layout object's tooltip calculation to display different tooltip text according to whether or not a value less than zero was entered.

```
if(self < 0; "Value is less than zero"; "Value is zero or greater")
```

Mobile functions

Mobile functions are used with FileMaker Go.

Click a function name for details.

This function	Returns
Location	The current latitude and longitude of an iOS device running FileMaker Go and the horizontal accuracy of the values returned.
LocationValues	The current latitude, longitude, and altitude of an iOS device running FileMaker Go; the horizontal and vertical accuracy of the values returned; and the number of minutes since the values were returned.

Location

Purpose

Returns the current latitude and longitude on an iOS device running FileMaker Go. Also returns the horizontal accuracy of the values returned. The location is obtained via GPS, cellular network, or Wi-Fi.

Format

```
Location (accuracy {; timeout})
```

Parameters

`accuracy` - any numeric expression or field containing a number that represents a distance in meters.

`timeout` - any numeric expression or field containing a number that represents the most time it will take to fetch the location. Measured in seconds, the default value is 60.

Parameters in curly braces { } are optional.

Data type returned

Text

Originated in

FileMaker Pro 12.0

Description

Returns and caches the location of a device in the format `latitude, longitude`. You can use the values that are returned to query map services. `Location` fetches the location values until `timeout`. If you cancel the process, FileMaker Go returns the most accurate location in the cache (if any).

If no location is received, FileMaker Go returns an empty string. In FileMaker Pro, `Location` returns an empty string.

Note To avoid excessive battery consumption and repeat fetches, specify a smaller number for timeout.

Examples

`Location ( 100; 40 )` takes up to 40 seconds to return the latitude and longitude with an accuracy of 100 meters.

```
+110.230000, -131.340000, +65.000000
```

LocationValues

Purpose

Returns the current latitude, longitude, and altitude on an iOS device running FileMaker Go. Returns the horizontal and vertical accuracy of the values returned and the number of minutes since the values were returned. The location is obtained via GPS, cellular network, or Wi-Fi.

Format

```
LocationValues (accuracy {; timeout})
```

Parameters

`accuracy` - any numeric expression or field containing a number that represents a distance in meters.

`timeout` - any numeric expression or field containing a number that represents the most time it will take to fetch the location. Measured in seconds, the default value is 60.

Parameters in curly braces { } are optional.

Data type returned

Text

Originated in

FileMaker Pro 12.0

Description

Returns and caches the current location of a device in the format:

```
latitude¶longitude¶altitude¶horizontal accuracy (+/- accuracy in meters)¶
vertical accuracy (+/- accuracy in meters)¶age of value in minutes (0.2
would represent 0.2 minutes or 12 seconds ago)
```

You can use the GetValue function to retrieve any of the six carriage return-delimited values above.

`LocationValues` fetches the location values until the requested accuracy is met or until `timeout`. If you cancel the process, FileMaker Go returns the most accurate location in the cache (if any). If no location is received, FileMaker Go returns an empty string.

In FileMaker Pro, `LocationValues` returns an empty string.

Note To avoid excessive battery consumption and repeat fetches, specify a larger number for accuracy and a smaller number for timeout.

Examples

`LocationValues` returns the following location for a device:

```
37.406489
-121.983428
0.000000
65
-1
0.001236
```

Number functions

Number functions are used to manipulate numeric data.

Click a function name for details.

This function	Returns
Abs	The absolute value (a positive number) of a number.
Ceiling	A number rounded up to the next integer.
Combination	The number of ways to uniquely choose a specified number of items from a set of specified size.
Div	An integer of the specified number divided by the divisor.
Exp	The value of the constant **e** (the base of the natural logarithm, equal to 2.7182818) raised to the power of a specified number.
Factorial	The factorial of a specified number stopping at 1, or at a specified number factorial.
Floor	A number rounded down to the next lower integer.
Int	The whole number (integer) part of the value you specify, without rounding.
Lg	The base 2 logarithm of the specified number, which can be any positive value.
Ln	The base-e (natural) logarithm of the specified number.
Log	The common logarithm (base 10) of the specified number, which can be any positive value.
Mod	The remainder after a specified number is divided by divisor.
Random	A random number between zero and one.
Round	A number rounded off to the specified precision (number of decimal places).
SetPrecision	Any math functions contained within the specified expression to the specified digits of precision, if the math function supports extended precision.
Sign	One of three possible values: **-1** when the specified number is negative, **0** when it's zero, and **1** when it's positive.
Sqrt	The square root of a number.
Truncate	A number truncated to the specified precision (number of decimal places), without evaluating the value of the discarded digits.

Abs

Purpose

Returns the absolute value of `number`.

Format

`Abs(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number, time

Originated in

FileMaker Pro 6.0 or earlier

Description

The absolute value of a number is positive. For example, if a negative number appears in a field, the `Abs` function removes the minus sign and changes it to a positive value.

Examples

`Abs(-123)` returns **123**.

`Abs(PriceDifference)` returns the positive value of the number in the PriceDifference field.

`Abs(TargetDate - ActualDate)` returns a positive value for the number of days difference between the values in TargetDate and ActualDate.

Ceiling

Purpose

Returns `number` rounded up to the next integer.

Format

```
Ceiling(number)
```

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Examples

`Ceiling(1.25)` returns **2**.

`Ceiling(-1.25)` returns **-1**.

Combination

Purpose

Returns the number of ways to uniquely choose `numberOfChoices` items from a set of size `setSize`.

Format

```
Combination(setSize;numberOfChoices)
```

Parameters

`setSize` - any numeric expression or field containing a non-negative numeric expression

`numberOfChoices` - any numeric expression or field containing a non-negative numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

This function is useful in statistics, combinatorics, and polynomial expansions. The values returned by this function are referred to as combination coefficients. They form Pascal's triangle.

$$\textbf{Combination} = \frac{\textbf{Factorial}(\textbf{setSize, numberOfChoices})}{\textbf{Factorial}(\textbf{numberOfChoices})}$$

Examples

`Combination(5;2)` returns **10** for a set consisting of {a, b, c, d, e} because the unique choices when choosing two at a time are {ab, ac, ad, ae, bc, bd, be, cd, ce, de}.

`(13 * 12 * Combination(4;2) * Combination(4;3)) / Combination(52;5)` returns **0.00144057...**, which is the probability of being dealt a full-house in 5-card poker (less than a 1% chance).

Div

Purpose

Returns the next lowest integer value after dividing `number` by `divisor`. Equivalent to `Floor(number/divisor)`.

Format

```
Div(number;divisor)
```

Parameters

`number` - any numeric expression or field containing a numeric expression

`divisor` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Examples

`Div(2.5;2)` returns **1**.

`Div(-2.5;2)` returns **-2**.

Exp

Purpose

Returns the value of the constant **e** (the base of the natural logarithm, equal to 2.7182818) raised to the power of `number`.

Format

`Exp(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The `Exp` function is the inverse of the `Ln` function.

Examples

`Exp(1)` returns **2.71828182...**.

`Exp(Ln(2))` returns **2**.

`Exp(0)` returns **1**.

Factorial

Purpose

Returns the factorial of `number` stopping at 1 or stopping at the optional `numberOfFactors`.

Format

`Factorial(number{;numberOfFactors})`

Parameters

`number` - numeric expression or field containing a positive integer.

`numberOfFactors` - any numeric expression or field containing a number that represents how many factors to include in the multiplication.

Parameters in curly braces { } are optional.

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

This function is useful in statistics and combinatorics.

Where `n = number` and `i = numberOfFactors`:

$$\textbf{Factorial}(\mathbf{n}) = \mathbf{n}(\mathbf{n}-\mathbf{1})(\mathbf{n}-\mathbf{2})\ldots(\mathbf{1})$$

$$\textbf{Factorial}(\mathbf{n;i}) = \mathbf{n}(\mathbf{n}-\mathbf{1})(\mathbf{n}-\mathbf{2})\ldots(\mathbf{n}-\mathbf{i}+\mathbf{1})$$

Examples

`Factorial(3)` returns **6**, which = 3 * 2 * 1.

`Factorial(10;3)` returns **720**, which = 10 * 9 * 8.

Floor

Purpose

Returns `number` rounded down to the next lower integer.

Format

`Floor(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Examples

`Floor(1.25)` returns **1**.

`Floor(-1.25)` returns **-2**.

Int

Purpose

Drops digits to the right of the decimal point and returns the integer part of `number` without rounding.

Format

`Int(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Int(1.45)` returns **1**.

`Int(-3.9)` returns -**3**.

`Int(123.9)` returns **123**.

`Int(Players/3)` returns **4**, if Players contains **13**.

Lg

Purpose

Returns the base-2 logarithm of `number`.

Format

`Lg(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

`Number` can be any positive value. Negative values return an error. For 0, the `Lg` function returns nothing because this value is out of the acceptable range.

$$\mathbf{Lg} = \frac{\mathbf{Ln}(\mathbf{number})}{\mathbf{Ln}(\mathbf{2})}$$

Examples

`Lg(1)` = **0**

`Lg(2)` = **1**

`Lg(32)` = **5**

Ln

Purpose

Returns the base-e (natural) logarithm of `number`.

Format

`Ln(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Number` can be any positive value. Negative values and 0 return an error. The `Exp` function is the inverse of the `Ln` function.

Examples

`Ln(2.7182818)` returns **.99999998...**.

`Ln(Exp(5))` returns **5**.

Log

Purpose

Returns the base-10 (common) logarithm of `number`.

Format

`Log(number)`

Parameters

`number` - any positive numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

`Number` can be any positive value. Negative values return an error. For 0, the `Log` function returns nothing because this value is out of the acceptable range.

$$\textbf{Log} = \frac{\textbf{Ln}(\textbf{number})}{\textbf{Ln}(\textbf{10})}$$

Examples

`Log(1)` returns **0**.

`Log(100)` returns **2**.

Mod

Purpose

Returns the remainder after `number` is divided by `divisor`.

Format

`Mod(number;divisor)`

Parameters

`number` - any numeric expression or field containing a numeric expression

`divisor` - numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the `Mod` function to test whether a number is even or odd by specifying a divisor of 2. If the result is zero, then the number is even; otherwise, it's odd. The result has the same sign as `divisor`.

$$\textbf{Mod} = \textbf{number} - (\textbf{Div}(\textbf{number;divisor}) \bullet \textbf{divisor})$$

Examples

`Mod(13;4)` returns **1**.

`Mod(7;5)` returns **2**.

`Mod(7;-5)` returns **-3**.

`Mod(-7;5)` returns **3**.

`Mod(-7;-5)` returns **-2**.

`Mod(Participants;TeamSize)` returns **4** if Participants contains **40** and TeamSize contains **9**.

`If(Mod(Get(RecordNumber);2) = 0;“even”;“odd”)` labels a record even or odd using the `Get(RecordNumber)` function.

Random

Purpose

Returns a number between zero and one, including zero, but not including one.

Format

Random

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Returns a pseudo-random number in the range (0,1). FileMaker Pro generates a new random number when you:

- insert the `Random` function into a formula
- cause a formula containing the `Random` function to be reevaluated (by changing data in any of the fields the formula uses)
- display or access a calculation field defined to have an unstored result

Examples

`Int ( Dice::NumSides * Random ) + 1` returns a randomly chosen side of a single die.

The following script calculates multiple die rolls, adds the rolls to a single variable, then displays the results in a custom dialog.

```
Loop
   Set Variable [$ROLL; Value:$ROLL + ( Int(Test::NumSides * Random) + 1)]
   Set Variable [$COUNTER; Value:$COUNTER + 1]
   Exit Loop If [$COUNTER = Dice::NumDice]
End Loop
Show Custom Dialog [$ROLL]
```

Round

Purpose

Returns `number` rounded off to the specified `precision` (number of decimal places).

Format

`Round(number;precision)`

Parameters

`number` - any numeric expression or field containing a numeric expression

`precision` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

If you round a negative number of decimal places, all digits to the right of the decimal point are dropped, and the number is rounded to the nearest tens, hundreds, and so on. The `Round` function always rounds up at 0.5.

Examples

`Round(123.456;2)` returns **123.46**.

`Round(14.5;0)` returns **15**.

`Round(29343.98;-3)` returns **29000**.

`Round(123.456;-1)` returns **120**.

SetPrecision

Purpose

Computes any math function with a precision of 16 to 400 digits to the right of the decimal point.

Format

`SetPrecision(expression;precision)`

Parameters

`expression` - any numeric expression

`precision` - any number or numeric expression

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

All functions except trigonometric functions support extended precision. This function doesn't perform a truncation.

Examples

`SetPrecision(5/9;30)` returns **0.555555555555555555555555555556**.

`SetPrecision(1.321321321321321321321321321;0)` returns **1.3213213213213213**.

`SetPrecision(If(field1>5;Exp(50);Average(5/9;1/7;5/7));25)` returns either **5184705528587072464087.4533229334853848274691006** if field1 > **5**, or **0.4708994708994708994708995** if field1 <= **5**.

Sign

Purpose

Returns one of three possible values: **-1** when `number` is negative, **0** when it's zero, and **1** when it's positive.

Format

`Sign(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Sign(15.12)` returns **1**.

`Sign(-175)` returns **-1**.

`Sign(BalanceDue)` returns **0**, if BalanceDue is a number field containing **0**.

Sqrt

Purpose

Calculates the square root of `number`.

Format

`Sqrt(number)`

Parameters

`number` - any positive number, numeric expression, or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate `Sqrt`.

$$\textbf{Sqrt} = \sqrt{\textbf{number}}$$

Examples

`Sqrt(4)` returns **2**.

`Sqrt(SquareFeet)` returns **6** if the SquareFeet number field contains **36**.

Truncate

Purpose

Returns `number` truncated to the specified `precision` (number of decimal places).

Format

`Truncate(number;precision)`

Parameters

`number` - any numeric expression or field containing a numeric expression

`precision` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

This function doesn't evaluate digits beyond the specified `precision`. Use the `Round` function to round up or down to the required precision.

Examples

`Truncate(123.456;2)` returns **123.45**.

`Truncate(-14.6;0)` returns **-14**.

`Truncate(29343.98;-3)` returns **29000**.

`Truncate(123.456;4)` returns **123.456**.

`Truncate(29343.98;5)` returns **29343.98**.

Repeating functions

Repeating functions perform calculations on repeating fields.

Click a function name for details.

This function	Returns
Extend	In a calculation involving both repeating and non-repeating fields, allows a value in a non-repeating field to be used with every repetition in a repeating field.
GetRepetition	The contents of the specified repetition of a repeating field.
Last	The last valid, non-blank value in the specified field.

Extend

Purpose

Allows a value in `non-repeatingField` to be used with every repetition in a repeating field.

Format

`Extend(non-repeatingField)`

Parameters

`non-repeatingField` - any non-repeating field (a field defined to contain only one value), or an expression that returns a reference to one

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the `Extend` function with calculations involving both repeating and non-repeating fields. Without the `Extend` function, the value in `non-repeatingField` is used only with the first repetition in the repeating field.

Examples

`Extend(TaxRate) * Quantity * ItemPrice` returns **1.197**, **.6606**, and **1.497** when `TaxRate` contains **.06**; the repeating field Quantity contains **1**, **3**, and **5**; and the repeating field ItemPrice contains **19.95**, **3.67**, and **4.99**.

`Item Count * Extend(if(Company Size > 100; Reduced Price; Price))` returns **$1250**, **$500**, and **$750** when `Reduced Price` contains $50; the repeating field `Item Count` contains 25, 10, and 15; and `Company Size` is greater than 100. If `Company Size` is less than 100 and `Price` contains $100, this calculation returns **$2500**, **$1000**, and **$1500**.

GetRepetition

Purpose

Returns the contents of the repeating field specified by `number`.

Format

`GetRepetition(repeatingField;number)`

Parameters

`repeatingField` - any repeating field, or an expression that returns a reference to a repeating field

`number` - the field repetition number

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Examples

ParcelBids is a field defined to repeat with ten values and contains the values 2500, 1200, and 1500.

`GetRepetition(ParcelBids;2)` returns **1200**.

`GetRepetition(if(IsEmpty(ParcelBids) ≠ true, ParcelBids, HouseBids);2)` returns **1200**.

`GetRepetition(ParcelBids;5)` returns nothing.

Note You can also find the contents of a particular repetition in a repeating field using square brackets [ ] as array operators. For example, `ParcelBids[2]` returns **1200**. See Getting the contents of a repetition in a repeating field.

Last

Purpose

Returns the last valid, non-blank value in `field`.

Format

`Last(field)`

Parameters

`field` - any repeating field or related field, or an expression that returns a reference to a repeating field or related field

Data type returned

text, number, date, time, timestamp, container

Originated in

FileMaker Pro 6.0 or earlier

Description

If `field` specifies a repeating field then it returns the last non-blank repetition. If `field` specifies a related field, then it returns the last non-blank value in the related set.

Note The last related value will depend on the way related records are sorted. If the related records are not sorted, then the `Last` function returns a value based on the creation order of the records.

Examples

`Last(ParcelBids)` returns **1500** if ParcelBids is a number field defined to repeat with ten values and contains the values 2500, 1200, and 1500.

`Last(Payments::PaymentDate)` returns the payment date in the last matching record in the Payments table.

`Last(if(IsEmpty(Company);PersonalPhone;WorkPhone))` returns the last non-empty phone number from the repeating field PersonalPhone when the Company field is empty. If the Company field is not empty, the function returns the last non-empty phone number from the repeating field WorkPhone.

Summary functions

Summary functions produce a summary of all records in the found set, or subsummary values for records in different groups. Formulas can contain more than one summary function. Summary functions calculate more slowly than other functions because they generate values for a range of records.

An alternative way to generate similar calculated results is to use Aggregate functions to summarize data in related records (whether or not they appear in a portal). See Aggregate functions and information about summarizing data in portals.

Click the function name for details.

This function	Returns
GetSummary	The value of the summary field for the current range of records when the database file is sorted by the break field.

GetSummary

Purpose

Returns the value of `summaryField` for the current range of records when the database file is sorted by `breakField`.

Format

`GetSummary(summaryField;breakField)`

Parameters

`summaryField` - field of type summary, or an expression that returns a reference to one.

`breakField` - field, or an expression that returns a reference to one. To calculate a grand summary value, use the same summary field for both the summary field and the break field parameters.

`GetSummary` must be set up in the same table as the break field.

Data type returned

number, date, time, timestamp

Originated in

FileMaker Pro 6.0 or earlier

Description

This function produces subsummary values. If the database file isn't sorted by the break field, the result is blank.

When a summary field is also used as the break field, returns the summary field value for the entire found set of records (a grand summary value).

Use `GetSummary` to capture summary values when you want to:

- use summary values in a calculation
- display subsummary values in Browse mode or in a body part

Calculations using the `GetSummary` function are unstored.

Note You can get similar results using a self-join relationship and Aggregate functions. For more information, see Summarizing data in portals.

Examples

`GetSummary(Total Sales;Country)` returns a summary of all records pertaining to the value in the Country field.

`GetSummary(Total Sales, if(Number of Countries > 1, Country, Sales Zone))` returns a summary of Total Sales by Country if Number of Countries is greater than 1. Otherwise, it returns a summary of Total Sales by Sales Zone.

`GetSummary(Total Sales;Total Sales)` produces a summary of all records (similar to using a summary field, which is a total of total sales).

`If(ThisCharge > 3 * GetSummary(AvgCharge;Customer), “Verify this charge”, “ “)` displays **Verify this charge** if the current charge is greater than three times the average charge.

Text functions

Text functions can be used to analyze, rearrange, extract, and build text strings. For example, you could use the `MiddleWords` function to extract specific words from supplied text.

Text functions operate on these parameters:

- fields of type text
- text constants (in quotation marks)
- expressions having a text result

Click a function name for details.

This function	Returns
Char	Returns the characters for the Unicode code points in the number.
Code	Returns the Unicode code points for the characters in the text. If zero characters are in the text, returns an empty string.
Exact	**1** (True) for an exact match, or **0** (False) for a mismatch between two text strings or container fields.
Filter	Only the specified characters, in the order that they were originally entered in the text.
FilterValues	Only the specified values, in the order that they were originally entered in the text.
GetAsCSS	The specified text, converted to the CSS (Cascading Style Sheets) format.
GetAsDate	Dates in the specified text as field type date, for use in formulas involving dates or date functions.
GetAsNumber	Numbers in the specified text as field type number, for use with formulas involving numbers or numeric functions. If zero numeric characters are in the text, returns an empty string.
GetAsSVG	The specified text, converted to the SVG (Scalable Vector Graphics) format.
GetAsText	The specified number, date, time or timestamp as field type text, for use with formulas involving text or text functions.
GetAsTime	Times or timestamps in the specified text as field type time, for use with formulas involving the time or timestamp functions.
GetAsTimestamp	The specified data as field type timestamp, for use with formulas involving timestamps.
GetAsURLEncoded	The specified text, converted with URL (Uniform Resource Locators) encoding.
GetValue	A specific value from a list of values.
Hiragana	Hiragana converted from Katakana (Hankaku and Zenkaku).
KanaHankaku	Hankaku Katakana converted from Zenkaku Katakana.
KanaZenkaku	Zenkaku Katakana converted from Hankaku Katakana.
KanjiNumeral	Kanji numerals converted from Arabic numerals.
Katakana	Zenkaku Katakana converted from Hiragana.
Left	The specified number of characters in the text, counting from the left.
LeftValues	The specified number of values in the text, counting from the left.
LeftWords	The specified number of words in the text, counting from the left.

This function	Returns
Length	The number of characters in the specified text, including all spaces, numbers, and special characters.
Lower	All letters in the specified text as lowercase.
Middle	The specified number of characters in the text, starting at a specified character position.
MiddleValues	The specified number of values in the text, starting with a specified value.
MiddleWords	The specified number of words in the text, starting with a specified word.
NumToJText	Roman numbers converted to Japanese text.
PatternCount	The number of occurrences of a search string in the specified text.
Position	The specified occurrence of a search string, starting from a specified position.
Proper	The first letter of each word in the specified text as uppercase, and all other letters as lowercase.
Quote	The specified text surrounded by quotation marks (“ ”).
Replace	A new string of characters consisting of the specified text as modified by the specified replacement text.
Right	The specified number of characters in the text, counting from the right.
RightValues	The specified number of values in the text, counting from the right.
RightWords	The specified number of words in the text, counting from the right.
RomanHankaku	Hankaku (alphanumeric & symbols) converted from Zenkaku (alphanumeric & symbols).
RomanZenkaku	Zenkaku (alphanumeric & symbols) converted from Hankaku (alphanumeric & symbols).
SerialIncrement	The combined text and numbers in a specified value, with the numbers incremented by the specified amount.
Substitute	A text string with every occurrence of a specified search string in the text replaced by a specified replacement string.
Trim	Text stripped of all leading and trailing spaces.
TrimAll	Text with full width spaces between non-Roman and Roman characters removed.
Upper	All letters in the specified text as uppercase.
ValueCount	A count of the total number of values in the specified text.
WordCount	A count of the total number of words in the specified text.

Char

Purpose

Returns the characters for the Unicode code points in the number.

Format

`Char(number)`

Parameters

`number` - a decimal number representing one or more Unicode code points

Data type returned

text

Originated in

FileMaker Pro 10.0

Description

Each group of five digits in the number is treated as a Unicode code point, and the character for each five-digit group is returned in the text.

If the number is 0, the function returns an empty string.

If the number is between 1 and 99,999, the function returns a single character.

If the number contains more than five digits, the function returns the string of characters represented by those code points.

Note Some Unicode characters can be represented by a single code point or multiple code points. For example, the character *ä* can be represented by the letter **a** plus ¨ (dieresis) or by the single character **ä**. The single code point version of this kind of character is called a precomposed character or a composite character.

Examples

`Char(0)` returns **an empty string (“”)**.

`Char(97)` returns **a**.

`Char(98)` returns **b**.

`Char(9800097)` returns **ab**.

`Char(228)` returns **ä**.

`Char(77600097)` returns **ä**. In this case the number represents two Unicode characters: the letter **a** and the dieresis character. When these two characters appear together in a string they are displayed as a single character.

Code

Purpose

Returns the Unicode code points for the characters in the text.

Format

`Code(text)`

Parameters

`text` - one or more characters

Data type returned

number

Originated in

FileMaker Pro 10.0

Description

Returns the Unicode code points for the characters in `text`. If zero characters are in `text`, returns an empty string.

If one character is in the text, the function returns the code point for that character. If the text contains multiple characters, the Unicode code point for each character is returned as a group of five digits where the code point for the first character is represented by the low five digits, the code point for the second character in the next higher (to the left) five digits, and so forth.

When converting a composite character such as **ä**, the function returns the Unicode code point for the composite character.

The following table shows how navigational characters are reported to a script activated by this trigger:

Key Pressed	Is reported as	Notes
backspace	8	Corresponds to Unicode/ASCII code for BS (backspace)
tab	9	Corresponds to Unicode/ASCII code for HT (horizontal tab)
shift-tab	9	The shift can be detected using the value returned from the Get(TriggerModifierKeys) function
enter	10	Corresponds to Unicode/ASCII code for LF (linefeed)
return	13	Corresponds to Unicode/ASCII code for CR (carriage return)
escape	27	Corresponds to Unicode/ASCII code for ESC (escape)
left arrow	28	Corresponds to Unicode/ASCII code for FS (file separator)
up arrow	29	Corresponds to Unicode/ASCII code for GS (group separator)
right arrow	30	Corresponds to Unicode/ASCII code for RS (record separator)
down arrow	31	Corresponds to Unicode/ASCII code for US (unit separator)
space	32	Corresponds to Unicode/ASCII code for Space

Key Pressed	**Is reported as**	**Notes**
forward delete	127	Corresponds to Unicode/ASCII code for Delete

Note If there are too many characters to be represented in the FileMaker number field type, the function returns a NaN (Not a Number) value.

Examples

`Code(“”)` returns an empty string.

`Code(“a”)` returns **97**.

`Code(“b”)` returns **98**.

`Code(“ab”)` returns **9800097**.

`Code(“ä”)` returns **228**.

`Code(“ä”)`(an **a** followed by a dieresis character entered in a separate keystroke) returns **77600097**.

Exact

Purpose

Compares the contents of any two fields. If the fields match, the result is **1** (True); otherwise, the result is **0** (False).

Format

`Exact(originalText;comparisonText)`

Parameters

`originalText` - any text expression, text field, or container field

`comparisonText` - any text expression, text field, or container field

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

For text to match exactly, the uppercase and lowercase usage must be the same. For container fields, the data must be stored in the same manner (either embedded, or stored by reference).

Note When evaluating values, text attributes such as font, styles, and sizes are not considered.

Tip If case isn't important, use the `Lower` or `Upper` function on both parameters to process data before checking for an exact match.

Examples

`Exact(“McDonald”;“McDonald”)` returns **1** (True).

`Exact(“McDonald”;“MCDONALD”)` returns **0** (False).

`Exact(Upper(“McDonald”);Upper(“MCDONALD”))` returns **1** (True).

`Exact(“John”;“John ”)` returns **0** (False).

`Exact(BillTo;ShipTo)` returns **1** (True) when the value in BillTo is the same as the value in ShipTo.

`Exact(Recipient;Upper(Recipient))` returns **1** (True), when Recipient contains JOHNSON.

`Exact(Country;“Spain”)` returns 1 (True) when the Country field contains Spain.

Filter

Purpose

Returns from `textToFilter` only those characters specified in `filterText`, in the order they were originally entered in `textToFilter`.

Format

`Filter(textToFilter;filterText)`

Parameters

`textToFilter` - any text expression or text field

`filterText` - the characters to preserve in the specified text

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

If `filterText` doesn't have any characters, an empty string is returned. The `Filter` function is case-sensitive.

Examples

`Filter(“(408)555-1212”;“0123456789”)` returns **4085551212**.

`Filter(“AaBb”;“AB”)` returns **AB**.

The following example removes all text from the provided data, then formats the remaining numbers in the preferred phone number formatting:

```
Let(phone = filter(theField;“0123456789”);“(“ & left(phone;3) & “)” &
middle(phone;4;3) & “-” & middle(phone;7;4))
```

If `theField` contains **Work: 408.555.1212** this calculation returns **(408)555-1212**.

FilterValues

Purpose

Returns a text result containing only the values that were provided in `filterValues`, in the order they were originally entered in `textToFilter`.

Format

`FilterValues(textToFilter; filterValues)`

Parameters

`textToFilter` - any text expression or text field

`filterValues` - values that you want to preserve in the specified text

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

If `filterValues` doesn't have any values, an empty string is returned.

Values are text items separated by carriage returns. A value can be empty, a single character, a word, a sentence, or a paragraph. When you press Enter or Return, you start creating a new value. The last value will be recognized with or without a carriage return.

When the `textToFilter` or the `filterValues` parameter is a literal string, you must insert a paragraph character (¶) between each item in the string. To insert a carriage return character, click the ¶ button in the Specify Calculation dialog box.

The `FilterValues` function is not case-sensitive.

Examples

`FilterValues(“Plaid¶Canvas¶Suitcase”;“Plaid¶Canvas”)` returns

Plaid

Canvas

`FilterValues(ValueListItems(“Database”;“Sizes”);“Medium¶Small”)` returns

Small

Medium

when a database file named Database has a value list Sizes that contains `Small¶Medium¶Large.`

GetAsCSS

Purpose

Returns `text` converted to the CSS (Cascading Style Sheets) format.

Format

`GetAsCSS(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

CSS format is an internet text format similar to HTML. CSS supports more text formats than HTML, so CSS will represent what you have typed more accurately.

Note The `GetAsCSS` function does not return formats that are set in the Conditional Formatting dialog box.

Examples

`GetAsCSS(text)` returns the example result shown below when the field text contains the word “Frank” and the word Frank has the following text attributes: Font = Helvetica, Font Size = 12 points, Font Color = red, Font Style = bold.

Example result:

<SPAN STYLE = “font-family: 'Helvetica';font-size: 12px;color:

#FF0000;font-weight: bold;text-align: left;”>Frank</SPAN>

GetAsDate

Purpose

Returns dates in `text` as field type date, for use in formulas involving dates or date functions.

Format

`GetAsDate(text)`

Parameters

`text` - any text expression or text field containing text in the same format as the date on the system where the file was created

Data type returned

date

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the `GetAsDate` or `Date` function to enter a date constant into a formula. The format of `text` date must be the same as the date format on the system where the file was created.

You can also use this function to convert the number of days to a date. If you specify a number as the parameter, it has to be between 1 (for 1/1/0001) and 1460970 (for 12/31/4000).

Note If the function returns a number instead of a date, go to the Specify Calculation dialog box and make sure the **Calculation result is** date.

Important To avoid errors when using dates, always use four-digit years. For more information about how FileMaker Pro handles two-digit dates, see Conversion of dates with two-digit years.

Examples

`GetAsDate("03/03/2014")` returns **3/3/2014**. You can perform date operations on this result using the Date function.

`GetAsDate(735516)` returns **10/10/2014**. The number 735516 specifies the number of days since 1/1/0001.

Use the following formula to determine the number of days elapsed between values in two timestamp fields:

`GetAsDate(EndDate) - GetAsDate(StartDate)` returns **90** if the value in the field `EndDate` is `4/1/2010 1:00 AM` and the value in the field `StartDate` is `1/1/2010 11:15 PM`.

GetAsNumber

Purpose

Returns only the numbers in `text`, as field type number, for use with formulas involving numbers or numeric functions.

Format

`GetAsNumber(text)`

Parameters

`text` - any text expression or text field containing numbers

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

This function drops all non-numeric characters from `text`. If zero numeric characters are in `text`, returns an empty string.

You can also use this function to convert a date to the number of days. The returned number is the number of days since 1/1/0001.

Examples

`GetAsNumber(“FY98”)` returns **98**.

`GetAsNumber(“$1,254.50”)` returns **1254.5**.

`GetAsNumber(“2 + 2”)` returns **22**.

`GetAsNumber(SerialNumber)` returns **35684**, when the value of SerialNumber is TKV35FRG6HH84.

`GetAsNumber(DateOfBirth)` returns **735516**, when the DateOfBirth field contains 10/10/2014.

`GetAsNumber(Passcode)`returns an empty string, when the Passcode field contains QTjPLeRMaCV.

GetAsSVG

Purpose

Returns `text` converted to the SVG (Scalable Vector Graphics) format.

Format

`GetAsSVG(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

SVG is an internet text format similar to HTML or CSS. SVG supports more text formats than HTML, so SVG will represent what you have typed more accurately.

Note The GetAsSVG function does not return formats that are set in the Conditional Formatting dialog box.

Examples

`GetAsSVG(text)` returns the example result (below) when the field text contains the word "Frank" and the word Frank has the following text attributes: Font = Helvetica, Font Size = 12 points, Font Color = red, Font Style = bold.

Example result:

```
<StyleList>
<Style#0>"font-family: 'Helvetica';font-size: 12px;color:
 #FF0000;font-weight: bold;text-align: left;",
 Begin: 1, End: 5</Style>
</StyleList>
<Data>
<Span style="0">Frank</Span>
</Data>
```

GetAsText

Purpose

Returns `data` as field type text, for use with formulas involving text or text functions.

Format

`GetAsText(data)`

Parameters

`data` - any number, date, time or timestamp expression, or a field containing a number, date, time, timestamp, or container

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

The `data` returned can be a field type number, date, time, timestamp, or container.

For a container field, `GetAsText` returns external path information, text (when the container contains text that does not resolve into a valid path), or a question mark (?) if the container data is embedded in the database. For container data stored externally, data is returned in the format shown in the following example:

`GetAsText(Container)`returns

remote:cat.jpg
size:320,240
JPEG:Images/Animals/cat.jpg

Examples

`GetAsText(45)` returns **45**.

`“You are ” & GetAsText(DaysDelinquent) & “ days late.”` returns **You are 3 days late.** when the value of DaysDelinquent is 3.

`“FY” & GetAsText(FiscalYear)` returns **FY98**, if the FiscalYear number field contains 98.

GetAsTime

Purpose

Returns times or timestamps in `text` as field type time, for use with formulas involving the `Time` or `Timestamp` functions.

Format

`GetAsTime(text)`

Parameters

`text` - any text expression or text field containing a time

Data type returned

time

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the `GetAsTime` or the `Time` function to enter a time constant into a formula. The format of the supplied time must be the same as the time format on the system where the file was created.

Examples

`GetAsTime(“02:47:35”)` returns **2:47:35** when you select time as the calculation result. You can perform time calculations on this result.

`GetAsTime(“02:47:35”)` returns **1/1/0001 2:47:35** when you select timestamp as the calculation result.

`Abs(GetAsTime(“12:15 pm”) - CheckOut)` returns **3:00:00** when the CheckOut time field contains 3:15 PM.

GetAsTimestamp

Purpose

Returns `text` as field type timestamp, for use with formulas involving timestamps.

Format

`GetAsTimestamp(text)`

Parameters

`text` - any text expression, or text, number, date, or time field

Data type returned

timestamp

Originated in

FileMaker Pro 7.0

Description

Text strings must be in the form of a date followed by a time. A number is considered to be the number of seconds since 1/1/0001. There are 86400 seconds in each day.

Examples

`GetAsTimestamp("4/5/2014 4:05:06")` returns **4/5/2014 4:05:06 AM**.

`GetAsTimestamp(50000)` returns **1/1/0001 1:53:20 PM**.

GetAsURLEncoded

Purpose

Returns `text` as URL (Uniform Resource Locator) encoding, for use as a URL.

Format

`GetAsURLEncoded(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 8.5

Description

This function removes all styles from `text`. All characters are first converted to UTF-8 format. Characters that are neither letters nor digits, or digits that are in the upper ASCII range, are converted to %HH format (a percent sign followed by the character's hexadecimal value).

See the following website for more information on URL encoding:

http://www.w3.org

Examples

`GetAsURLEncoded("Hello")` returns **Hello**.

`GetAsURLEncoded("San Francisco")` returns **San%20Francisco**.

`GetAsURLEncoded("français")` returns **fran%c3%a7ais**.

GetValue

Purpose

Returns the requested value given by `valueNumber` from `listOfValues`.

Format

`GetValue(listOfValues;valueNumber)`

Parameters

`listOfValues` - a list of carriage return-delimited values

`valueNumber` - the value to return from the list

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

This function is useful in looping scripts or recursive custom calculations.

Values are text items separated by carriage returns. You can place several values together to create a carriage return-delimited list of values. A value can be empty, a single character, a word, a sentence, or a paragraph. When you press Enter or Return, you start creating a new value. The last value will be recognized with or without a carriage return.

When the `listOfValues` parameter is a literal string, you must insert a literal carriage return character (¶) between each item in the string. To insert a literal carriage return character, click the ¶ button in the Specify Calculation dialog box.

Examples

`GetValue(“London¶Paris¶Hong Kong”;2)` returns

Paris

Hiragana

Purpose

Converts Katakana (Hankaku and Zenkaku) in `text` to Hiragana.

Format

`Hiragana(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Hiragana("アイウエオ")` returns あいうえお

KanaHankaku

Purpose

Converts Zenkaku Katakana to Hankaku Katakana.

Format

`KanaHankaku(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`KanaHankaku(“データベース”)` returns ﾃﾞｰﾀﾍﾞｰｽ

KanaZenkaku

Purpose

Converts Hankaku Katakana to Zenkaku Katakana.

Format

`KanaZenkaku(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`KanaZenkaku(“` "ﾃﾞｰﾀﾍﾞｰｽ" `“)` returns データベース

KanjiNumeral

Purpose

Converts Arabic numerals to Kanji numeral.

Format

`KanjiNumeral(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`KanjiNumeral(123)` returns 一二三

`KanjiNumeral(“富士見台２の３の２５“)` returns 富士見台二の三の二五

Katakana

Purpose

Converts from Hiragana to Zenkaku Katakana.

Format

`Katakana(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Katakana(“あいうえお“)` returns アイウエオ

Left

Purpose

Returns `numberOfCharacters` in `text`, counting from the left.

Format

`Left(text;numberOfCharacters)`

Parameters

`text` - any text expression or text field

`numberOfCharacters` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Left(“Manufacturing”;4)` returns **Manu**.

`Left(Name;Position(Name;“ “;1;1))` returns **Sophie**, when the Name field contains Sophie Tang.

`Left(PostalCode;3) & Upper(Left(LastName;4))` returns **481JOHN** when the PostalCode field contains 48187 and LastName contains Johnson.

LeftValues

Purpose

Returns a text result containing `numberOfValues` from the list of values in `text`, counting from the left.

Format

`LeftValues(text;numberOfValues)`

Parameters

`text` - any text expression or text field

`numberOfValues` - any numeric expression or field containing a number

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Values are text items separated by carriage returns. A value can be empty, a single character, a word, a sentence, or a paragraph. When you press Return you start creating a new value. The last value will be recognized with or without a carriage return.

Each returned value ends with a carriage return, allowing lists to be easily concatenated.

Examples

`LeftValues(“Plaid¶Canvas¶Suitcase”;2)` returns

Plaid

Canvas

`LeftValues(list;1)` returns

Sophie

when the text being evaluated contains

- Sophie
- Bill

LeftWords

Purpose

Returns a text result containing `numberOfWords` in `text`, counting from the left.

Format

`LeftWords(text;numberOfWords)`

Parameters

`text` - any text expression or text field

`numberOfWords` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`LeftWords(“Plaid Canvas Suitcase”;2)` returns **Plaid Canvas**.

`LeftWords(Name;1)` returns **Sophie**, when the Name field contains Sophie Tang.

Note Characters such as the ampersand (&) and hyphen (-) can be used to identify the beginning of a new word.

Length

Purpose

Returns the number of characters in `field`, including all spaces, numbers, and special characters.

Format

`Length(field)`

Parameters

`field` - any text, number, date, time, timestamp, or container field, or any text expression or numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

This function returns the number of characters in a specified field. For a container field, `Length` returns the size of the original file in bytes.

Examples

`Length(“John”)` returns **4**.

`Length(Description)` returns **12** when the value in Description is Modem for PC.

`Length(“M1” & Left(Product;5))` returns **7**, when the Product field contains Canvas Backpack.

Lower

Purpose

Returns all letters in `text` as lowercase.

Format

`Lower(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Note

To change how a field displays without modifying its contents, see Text formatting functions.

Examples

`Lower("ABCD")` returns **abcd**.

`Lower(Course)` returns **history**, when the Course field contains History.

`Lower("YOUR BILL IS OVERDUE")` returns **your bill is overdue**.

Middle

Purpose

Extracts the `numberOfCharacters` from `text`, starting at the character position specified by `start`.

Format

`Middle(text;start;numberOfCharacters)`

Parameters

`text` - any text expression or text field

`start` - any numeric expression or field containing a number

`numberOfCharacters` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Middle(“(408)555-9054”;2;3)` returns **408**.

`Middle(PhoneNumber;2;3)` returns **408** when the PhoneNumber field contains (408) 555-9054.

`Middle(“abcdefghij”;5;2)` returns **ef**.

`Middle(Name;Position(Name;“ ”;1;1)+1;3)` returns **Smi**, when the text field Name contains John Smith.

MiddleValues

Purpose

Returns a text result containing the specified `numberOfValues` in `text`, starting with `startingValue`.

Format

`MiddleValues(text;startingValue;numberOfValues)`

Parameters

`text` - any text expression or text field

`startingValue` - any numeric expression or field containing a number

`numberOfValues` - any numeric expression or field containing a number

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Values are text items separated by carriage returns. A value can be empty, a single character, a word, a sentence or a paragraph. When you press Return you start creating a new value. The last value will be recognized with or without a carriage return.

Each value that is returned ends with a carriage return, allowing lists to be easily concatenated.

Examples

`MiddleValues(“Plaid¶Canvas¶Suitcase”;2;1)` returns

Canvas

`MiddleValues(list;2;2)` returns

Bill

John

when the list field contains

- Sophie
- Bill
- John

MiddleWords

Purpose

Returns a text result containing the `numberOfWords` from `text`, beginning at `startingWord`.

Format

`MiddleWords(text;startingWord;numberOfWords)`

Parameters

`text` - any text expression or text field

`startingWord` - any numeric expression or field containing a number

`numberOfWords` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`MiddleWords(“Plaid Canvas Suitcase”;2;2)` returns **Canvas Suitcase**.

`MiddleWords(Name;1;2)` returns **Brigitte Erika**, when the Name field contains Brigitte Erika Durand.

Note Characters such as the ampersand (&) and hyphen (-) can be used to identify the beginning of a new word.

NumToJText

Purpose

Converts Roman numbers in `number` to Japanese text.

Format

`NumToJText(number;separator;characterType)`

Parameters

`number` - any numeric expression or field containing a number

`separator` - a number from 0 - 3 representing a separator

`characterType` - a number from 0 - 3 representing a type

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

If the value for `separator` and `characterType` are blank or other than 0 to 3, then 0 is used.

Separator:

- 0 - no separator
- 1 - every 3 digits (thousands)
- 2 - ten thousands(万) and millions(億) unit
- 3 - tens(十), hundreds(百), thousands(千), ten thousands(万) and millions(億) unit

Type:

- 0 - half width (Hankaku) number
- 1 - full width (Zenkaku) number
- 2 - Kanji character number 一二三
- 3 - Traditional-old-style Kanji character number 壱弐参

Examples

`NumToJText(123456789;2;0)` returns 1億2345万6789

`NumToJText(123456789;3;2)` returns 一億二千三百四十五万六千七百八十九

PatternCount

Purpose

Returns the number of occurrences of `searchString` in `text`.

Format

`PatternCount(text;searchString)`

Parameters

`text` - any text expression or text field

`searchString` - any text expression or text field representing the set of characters you want to find

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

This function is not case-sensitive.

Examples

`PatternCount(“Mississippi”;“is”)` returns **2**.

`PatternCount(“Mississippi”;“issi”)` returns **1** (the function isn’t inclusive).

`PatternCount(Attending;“Guest”)` returns **1** if the Guest checkbox is one of the items selected in the Attending field.

Position

Purpose

Returns the starting position of the specified occurrence of `searchString` in `text`.

Format

`Position(text;searchString;start;occurrence)`

Parameters

`text` - any text expression or text field

`searchString` - any text expression or text field representing the set of characters you want to find.

`start` - any numeric expression, or field containing a number, representing the number of characters from the start of the text string at which to begin the search.

`occurrence` - any numeric expression or field containing a number, representing which instance of the text string you want to find. A negative occurrence value causes the scan to go in the opposite direction from start. A zero value for occurrence is invalid and returns a result of zero.

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

This function is not case-sensitive. If `searchString` isn't contained in text or if there was no specified occurrence, zero is returned.

Examples

`Position(“Mississippi”;“iss”;1;1)` returns **2**.

`Position(“Mississippi”;“iss”;1;2)` returns **5**.

`Position(“Mississippi”;“iss”;3;1)` returns **5**.

`Left(Name;Position(Name;“ “;1;1)-1)` returns **William**, when Name is a text field that contains William Smith.

`Right(Name;Length(Name) - Position(Name;“ “;Length(Name);-1))` returns **Smith**.

Proper

Purpose

Returns the first letter of each word in `text` as uppercase and all other letters as lowercase.

Format

`Proper(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Proper(“ABCD”)` returns **Abcd**.

`Proper(Name)` returns **Yumiko Kitagawa**, when the Name field contains YUMIKO KITAGAWA.

Quote

Purpose

Returns the text form of `text` enclosed in quotation marks.

Format

`Quote(text)`

Parameters

`text` - any text expression or field

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

This function protects text from being evaluated by the `Evaluate` function. Special characters within `text` are escaped appropriately.

Examples

`Quote(“hello”)` returns **“hello”**.

`Quote(“abc\¶”)` returns **“abc\¶”**.

`Quote(“say \”hello\” fred”)` returns **“say \”hello\” fred”**.

`Evaluate(Quote(“1 + 2”))` returns **1 + 2**.

`Evaluate(“1 + 2&” & Quote(“ - 1 + 2”))` returns **3 - 1 + 2**.

Replace

Purpose

Replaces a string of characters in `text` with `replacementText`.

Format

`Replace(text;start;numberOfCharacters;replacementText)`

Parameters

`text` - any text expression or text field

`start` - any numeric expression or field containing a number representing the starting position in `text`

`numberOfCharacters` - any numeric expression or field containing a number representing the number of characters to remove from text

`replacementText` - any text expression or field containing the text to replace in the original string

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Character replacement in `text` begins at the `start` character position and continues for `numberOfCharacters` characters. Compare to the `Substitute` function.

Examples

`Replace(“1234567”;5;1;“X”)` returns **1234X67**.

`Replace(“1234567”;5;1;“XX”)` returns **1234XX67**.

`Replace(“1234567”;5;2;“X”)` returns **1234X7**.

`Replace(“William”;3;4;”NEW TEXT”)` returns **WiNEW TEXTm**.

`Replace(PhoneNumber;1;3;”415”)` returns **415-555-9054**, when the PhoneNumber field contains 408-555-9054.

Right

Purpose

Returns the specified `numberOfCharacters` in `text`, counting from the right.

Format

`Right(text;numberOfCharacters)`

Parameters

`text` - any text expression or text field

`numberOfCharacters` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Right(“Manufacturing”;4)` returns **ring**.

`Right(Name;Length(Name) - Position(Name;“ “;1;1))` returns **Cannon**, when the Name field contains Michelle Cannon.

`Right(SerialNumber;3) & Upper(Left(LastName;4))` returns **187FERR** when the SerialNumber text field contains 00-48-187 and LastName contains Ferrini.

RightValues

Purpose

Returns a text result containing the specified `numberOfValues` in `text`, counting from the right.

Format

`RightValues(text;numberOfValues)`

Parameters

`text` - any text expression or text field

`numberOfValues` - any numeric expression or field containing a number

Important See Design functions for information about literal text parameters.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Values are text items separated by carriage returns. You can place several items together to create a carriage return-delimited list of values. A value can be empty, a single character, a word, a sentence, or a paragraph. When you press Return you start creating a new value. The last value will be recognized with or without a carriage return.

When the `text` parameter is a literal string as in the example below, you must insert a literal carriage return character between each item in the list. In the Specify Calculation dialog box, click the ¶ button to insert a literal carriage return character.

Each value that is returned ends with a carriage return, allowing lists to be easily concatenated.

Examples

`RightValues(“Plaid¶Canvas¶Suitcase”;2)` returns

Canvas

Suitcase

`RightValues(names;1)` returns

John

when the names field contains

- Sophie
- Bill
- John

RightWords

Purpose

Returns a text result containing the `numberOfWords` in `text`, counting from the right.

Format

`RightWords(text;numberOfWords)`

Parameters

`text` - any text expression or text field

`numberOfWords` - any numeric expression or field containing a number

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`RightWords("Plaid Canvas Suitcase";2)` returns **Canvas Suitcase**.

`RightWords(Name;1)` returns **Virtanen**, when the Name field contains Matti Virtanen.

Note Characters such as the ampersand (&) and hyphen (-) can be used to identify the beginning of a new word.

RomanHankaku

Purpose

Converts from Zenkaku alphanumeric and symbols to Hankaku alphanumeric and symbols.

Format

`RomanHankaku(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`RomanHankaku(“Ｍａｃｉｎｔｏｓｈ”)` returns **Macintosh**.

RomanZenkaku

Purpose

Converts from Hankaku alphanumeric and symbols to Zenkaku alphanumeric and symbols.

Format

`RomanZenkaku(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Examples

`RomanZenkaku`(“Macintosh”) returns **Ｍａｃｉｎｔｏｓｈ**.

SerialIncrement

Purpose

Returns the combined text and numbers specified by `text`, with the numbers in `text` incremented by the specified amount.

Format

`SerialIncrement(text;incrementBy)`

Parameters

`text` - any text that also contains a number

`incrementBy` - any numeric expression to increment the text by

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

This function doesn’t remove the text in `text`, which normally happens when performing standard math against a value that contains text.

If the `incrementBy` value is a decimal number, then only the integer portion of `incrementBy` value is added to the last number in `text`. Any character other than a number is considered a separator. You can use both positive and negative `incrementBy` values.

Examples

`SerialIncrement(“abc12”;1)` returns **abc13**.

`SerialIncrement(“abc12”;7)` returns **abc19**.

`SerialIncrement(“abc12”;-1)` returns **abc11**.

`SerialIncrement(“abc12”;1.2)` returns **abc13**.

`SerialIncrement(“abc1.2”;1.2)` returns **abc1.3**.

In the example below any character other than a number is considered as a separator and the number on the far right is incremented.

`SerialIncrement(“abc123;999”;1)` returns **abc123;1000**.

Substitute

Purpose

Returns a text string with every occurrence of `searchString` in `text` replaced by `replaceString` in `text`.

Format

```
Substitute(text;searchString;replaceString)
```

Parameters

`text` - any text expression or text field

`searchString` - any text expression or text field

`replaceString` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

This function is case-sensitive. Compare to the `Replace` function.

Multiple substitutions are allowed when you enclose each pair of `searchString` and `replaceString` parameters within square brackets [ ] and separate them with semicolons. FileMaker supports up to 999 nested substitute conditions. Each search and replace list item is also separated by semicolons. For example:

```
Substitute(text; [search1; replace1]; [search2; replace2]; ... [searchN;
replaceN])
```

Examples

`Substitute(Description;“WYSIWYG.”;“What you see is what you get.”)` replaces every occurrence of the acronym “WYSIWYG.” in the Description field with the phrase **What you see is what you get.**

`Substitute(text;[“a”;“A”];[“b”;“B”])` replaces every lowercase a or b with **A** or **B**.

Trim

Purpose

Returns `text` stripped of all leading and trailing spaces.

Format

`Trim(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to remove unneeded spaces when you convert files from other programs or systems that require a fixed number of characters per field, or to remove spaces accidentally typed during data entry.

Examples

`Trim(“ Tom ”)` returns **Tom**.

`Trim(Middle(“00230013 William 1234”;9;9))` returns **William**.

TrimAll

Purpose

Returns a copy of `text` with specified spaces removed or inserted. Use to work with spaces between text or non-Roman spaces such as full- and half-width spaces; otherwise, use Trim.

Format

`TrimAll(text;trimSpaces;trimType)`

Parameters

`text` - any text expression or text field

`trimSpaces` - 0 or False, 1 or True

`trimType` - 0 through 3 depending on the trim style that you wish to use

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Set trimSpaces to True (1) if you want to remove full-width spaces. Set `trimSpaces` to False (0) if you want to keep full-width spaces.

Note Full-width spaces are only present in some non-Roman languages like Japanese. If you only use Roman languages, set trimSpaces to False (0).

A character is considered Roman if its Unicode value is less than U+2F00. Any character whose Unicode value is greater than or equal to U+2F00 is considered non-Roman.

Characters within the Roman range are those belonging to the following character blocks: Latin, Latin-1 Supplement, Latin Extended-A & B, IPA Extensions, Spacing Modifier Letters, Combining Diacritical Marks, Greek, Cyrillic, Armenian, Hebrew, Arabic, Devanagari, Bengali, Gurmukhi, Gujarati, Oriya, Tamil, Telugu, Kannada, Malayalam, Thai, Lao, Tibetan, Georgian, Hangul Jamo, and additional Latin and Greek extended blocks.

Symbols within the Roman range include punctuation characters, superscripts, subscripts, currency symbols, combining marks for symbols, letter-like symbols, number forms, arrows, math operators, control pictures, geometric shapes, dingbats, and so on.

Characters within the non-Roman range are those belonging to the CJK symbols/punctuations area, Hiragana, Katakana, Bopomofo, Hangul compatibility Jamo, Kanbun, CJK unified ideographs, and so on.

Spaces are removed or inserted depending on the value of `trimType,` as given in the following tables:

This trimType value	Does this
0	Removes all spaces between non-Roman and Roman characters (always leave one space between Roman words).

This trimType value	Does this
1	Always includes a half-width space between non-Roman and Roman characters (always leave one space between Roman words).
2	Removes spaces between non-Roman characters (reduce multiple space between non-Roman and Roman words to 1 space; do not add spaces if there are none; always leave one space between Roman words).
3	Removes all spaces everywhere.

In all cases, spaces between non-Roman characters are removed.

Type	Non-Roman - Non-Roman	Non-Roman - Roman	Roman - Roman
0	Remove	Remove	1 space
1	Remove	1 space*	1 space
2	Remove	1 space	1 space
3	Remove	Remove	Remove

* = insert space between non-Roman and Roman text if there isn’t one.

Examples

TrimAll(" Julian   Scott Dunn ";0;0) returns **Julian Scott Dunn**.

TrimAll(名前,1,0) returns 山田太郎 if the value of 名前 field is 山田　太郎

TrimAll("ファイルメーカーPro　は高品質",1,0) returns ファイルメーカーProは高品質

Upper

Purpose

Returns all letters in `text` as uppercase.

Format

`Upper(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 6.0 or earlier

Description

Use the `Upper` function to ensure consistent data entry of such things as state abbreviations or postal codes.

Note To change how a field displays without modifying its contents, see Text formatting functions.

Examples

`Upper(“Ca”)` returns **CA**.

`Upper(“12n34p”)` returns **12N34P**.

ValueCount

Purpose

Returns a count of the total number of values in `text`.

Format

`ValueCount(text)`

Parameters

`text` - any text expression or text field

Important See Design functions for information about literal text parameters.

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

Values are text items separated by carriage returns. You can place several items together to create a carriage-return-delimited list of values. A value can be empty, a single character, a word, a sentence, or a paragraph. When you press Return you start creating a new value. The last value will be recognized with or without a carriage return.

When the `text` parameter is a literal string as in the example below, you must insert a literal carriage return character between each item in the list. In the Specify Calculation dialog box, click the ¶ button to insert a literal carriage return character.

Examples

`ValueCount(“Item 1¶Item 2¶Item 3”)` returns **3**.

`ValueCount(ValueListItems(“Employees”;“Employee Names”))` returns the total number of values in the Employee Names value list in the Employees database file.

WordCount

Purpose

Returns a count of the total number of words in `text`.

Format

`WordCount(text)`

Parameters

`text` - any text expression or text field

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`WordCount("The sun is rising.")` returns **4**.

`WordCount(Letter)` returns the total number of words in the Letter field.

Note Characters such as the ampersand (&) and hyphen (-) can be used to identify the beginning of a new word.

Text formatting functions

Text formatting functions can be used to change the color, font, size, and style of the specified text. For example, you could use the `TextFont` function to change the font of the specified text from Arial to Courier. You can use these functions together to change the appearance of text on your layouts.

Text formatting functions operate on these parameters:

- fields of type text
- text constants (in quotations)
- expressions having a text result

Click a function name for details.

This function	Returns
RGB	A number obtained by combining the red, green, and blue values to represent a color.
TextColor	The color of text to the color specified by the `RGB` function.
TextColorRemove	Text with the font colors reverted to the default font color for the field.
TextFont	Text in the specified font and character set.
TextFontRemove	Text with the fonts reverted to the default font for the field.
TextFormatRemove	Text with the formatting reverted to the default text format for the field.
TextSize	Text in the specified font size.
TextSizeRemove	Text with the font sizes reverted to the default font size for the field.
TextStyleAdd	Text with the specified styles added in a single action.
TextStyleRemove	Text with the specified styles removed in a single action.

RGB

Purpose

Returns an integer number from 0 to 16777215 obtained by combining the `red`, `green,` and `blue` values (each ranging from 0 to 255) to represent a color.

Format

`RGB(red;green;blue)`

Parameters

`red` - any numeric expression or numeric field containing a value ranging from 0 to 255

`green` - any numeric expression or numeric field containing a value ranging from 0 to 255

`blue` - any numeric expression or numeric field containing a value ranging from 0 to 255

Data type returned

number

Originated in

FileMaker Pro 7.0

Description

Numbers returned by this function can be passed as the color parameter in the `TextColor` or `TextColorRemove` functions. The `RGB` function uses the following formula to calculate the result:

`red * 256`^{2} `+ green * 256 + blue`

where $256^2 = 65536$

Tip To determine the RGB value of a color, in Layout mode, click the Fill color palette in the formatting bar and choose **Other Color**. In OS X, select the Color Sliders tab. Values are shown for each of the basic colors.

Examples

`RGB(255;0;0)` returns **16711680** representing red.

`RGB(0;255;0)` returns **65280** representing green.

`RGB(0;0;255)` returns **255** representing blue.

`RGB(0;0;0)` returns **0** representing black.

`RGB(255;255;255)` returns **16777215** representing white.

Using a table with text fields FirstName and LastName, specify the following auto-enter calculation for a third field called FullName that displays FirstName in orange and LastName in purple:

```
TextColor(FirstName;RGB(255;165;0)) &" "&
TextColor(LastName;RGB(160;32;240))
```

TextColor

Purpose

Changes the color of `text` to the color specified by the `RGB` function.

Format

`TextColor(text;RGB(red;green;blue))`

Parameters

`text` - any text expression or text field

`RGB(red;green;blue)` - any integer from 0 to 16777215 obtained by combining the `red`, `green`, and `blue` values (each ranging from 0 to 255) to represent a color

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Use this function to change the color of `text`.

Note Text formatting options will be lost if the field type that is returned is something other than text.

Tip To determine the RGB value of a color, in Layout mode, click the Fill color palette in the formatting bar and choose **Other Color**. In OS X, select the Color Sliders tab. Values are shown for each of the basic colors.

Examples

`TextColor(“Plaid”;RGB(255;0;0))` returns the word **Plaid** in red.

`TextColor(“Plaid”;RGB(0;255;0))` returns the word **Plaid** in green.

`TextColor(“Plaid”;RGB(0;0;255))` returns the word **Plaid** in blue.

`TextColor(“Plaid”;RGB(0;0;0))` returns the word **Plaid** in black.

`TextSize( TextFont( TextColor( MyTable::MyText; RGB( 0 ; 125 ; 125 ) ); "Courier" ) ; 12)` returns the text contained in **MyTable::MyText** formatted as 12pt. green text with the Courier font.

TextColorRemove

Purpose

Removes all font colors in `text`, or removes the font color specified by the `RGB` function.

Format

`TextColorRemove(text{;RGB(red;green;blue)})`

Parameters

`text` - any text expression or text field.

`RGB(red;green;blue)` - any integer number from 0 to 16777215 obtained by combining the `red`, `green`, and `blue` values (each ranging from 0 to 255) to represent a color.

Parameters in curly braces { } are optional.

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

Use this function to revert `text` to the default font color for the field. If you don't use the `RGB` function to specify a color, all of the text displays in the default font color that was set in Layout mode for the field. When the font color is specified by the `RGB` function, only the specified font color is removed from every portion of the text displayed in that color and these same portions of the text are then displayed in the field's default font color.

Note Text formatting options will be lost if the field type that is returned is something other than text.

Examples

`TextColorRemove("Red Text and Green Text")` returns **Red Text and Green Text** displayed in the field's default font color.

`TextColorRemove("Red Text and Green Text";RGB(255;0;0))` returns **Red Text and Green Text** with only the pure red font color removed from the words **Red Text**.

TextFont

Purpose

Changes the font of `text` to the specified `fontName` or optional `{fontScript}`.

Format

```
TextFont(text;fontName{;fontScript})
```

Parameters

`text` - any text expression or text field.

`fontName` - any font name expressed in text.

`{fontScript}` - the name of a character set that contains characters required for writing in the specified language.

Parameters in curly braces { } are optional.

Note The `fontScript` parameter is not enclosed in quotation marks (" "), and can have any of the values listed below in Description.

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

Spellings for font names must be correct. Text formatting options will be lost if the field type that is returned is something other than text.

FileMaker Pro looks for a font that matches the specified font name and font script character set. If no matches exist, FileMaker Pro looks for a default font with the font script specified in the **Fonts** tab of the Preferences dialog box. If this fails, then the `TextFont` function uses the default font for the system script specified in the **Fonts** tab of the Preferences dialog box. This font might not be the same as the font script provided.

The following font scripts are available:

- Roman
- Greek
- Cyrillic
- CentralEurope
- ShiftJIS
- TraditionalChinese
- SimplifiedChinese
- OEM
- Symbol
- Other

Examples

`TextFont(“Plaid”;“Courier”)` returns the word **Plaid** in the Courier font.

`TextFont(“Plaid”;“Arial”)` returns the word **Plaid** in the Arial font.

`TextFont(“Plaid”;“Arial”;Cyrillic)` returns the word **Plaid** in the Arial font in the font script of Cyrillic.

`TextSize( TextFont( TextColor( MyTable::MyText; RGB( 0 ; 125 ; 125 ) ); "Courier" ) ; 12)` returns the text contained in **MyTable::MyText** formatted as 12pt. green text with the Courier font.

TextFontRemove

Purpose

Removes all fonts in `text`, or removes the font specified by `fontToRemove` or the combination of `fontToRemove` and `fontScript`.

Format

```
TextFontRemove(text{;fontToRemove;fontScript})
```

Parameters

`text` - any text expression or text field.

`fontToRemove` - any font name expressed in `text`.

`fontScript` - the name of a character set that contains characters required for writing in the specified language.

Parameters in curly braces { } are optional.

Note The `fontScript` parameter is not enclosed in quotation marks (“ “), and can have any of the values listed below in Description.

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

Use this function to revert `text` to the default for the field. If you don’t specify a font, all of the text displays in the default font that was set in Layout mode for the field. When the font is specified by `fontToRemove` or the combination of `fontToRemove` and `fontScript`, only the specified font is removed from every portion of the text displayed in that font and these same portions of the text are then displayed in the field's default font.

Spellings for font names must be correct. Text formatting options will be lost if the field type that is returned is something other than text.

FileMaker Pro looks for a font that matches the specified font name and font script character set. If no matches exist, FileMaker Pro looks for a default font with the font script specified in the **Fonts** tab of the Preferences dialog box. If this fails, then the `TextFontRemove` function uses the default font for the system script specified in the **Fonts** tab of the Preferences dialog box. This font might not be the same as the font script provided.

The following font scripts are available:

- Roman
- Greek
- Cyrillic
- CentralEurope
- ShiftJIS

- TraditionalChinese
- SimplifiedChinese
- OEM
- Symbol
- Other

Examples

`TextFontRemove(“Arial Text and Courier Text”)` returns **Arial Text and Courier Text** displayed in the field’s default font.

`TextFontRemove(“Arial Text and Courier Text”;“Arial”)` returns **Arial Text and Courier Text** with the Arial font removed from the words **Arial Text** for all `fontScripts` that use the Arial font.

`TextFontRemove(“Arial Text and Courier Text”;“Arial”;Cyrillic)` returns **Arial Text and Courier Text** with the Arial font removed from Cyrillic character sets.

TextFormatRemove

Purpose

Removes all text formatting from `text` in a single action.

Format

`TextFormatRemove(text)`

Parameters

`text` - any text expression or text field

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

Use this function to remove all fonts, styles, font sizes, and font colors from the specified text.

Examples

`TextFormatRemove(“Plaid”)` returns the word **Plaid** without any text formatting applied.

TextSize

Purpose

Changes the font size of `text` to `fontSize`.

Format

`TextSize(text;fontSize)`

Parameters

`text` - any text expression or text or number field

`fontSize` - any font size expressed as an integer

Data type returned

text, number

Originated in

FileMaker Pro 7.0

Description

The font size is described in points (72 points to the inch). Text formatting options will be lost if the data type that is returned is something other than text or number.

Examples

`TextSize(“Plaid”;18)` returns the word **Plaid** in 18 point text.

`TextSize(“Plaid”;24)` returns the word **Plaid** in 24 point text.

`TextSize( TextFont( TextColor( MyTable::MyText; RGB( 0 ; 125 ; 125 ) ); "Courier" ) ; 12)` returns the text contained in **MyTable::MyText** formatted as 12pt. green text with the Courier font.

TextSizeRemove

Purpose

Removes all font sizes in `text`, or removes the font size specified by `sizeToRemove`.

Format

`TextSizeRemove(text{;sizeToRemove})`

Parameters

`text` - any text expression or text field.

`sizeToRemove` - any font size expressed as an integer.

Parameters in curly braces { } are optional.

Data type returned

text

Originated in

FileMaker Pro 8.0

Description

Use this function to revert `text` to the default font size for the field. If you don't specify a size, all of the text displays in the default font size that was set in Layout mode for the field. When the font size is specified by `sizeToRemove`, only the specified font size is removed from every portion of the text displayed in that size and these same portions of the text are then displayed in the field's default font size.

The font size is described in points (72 points to the inch). Text formatting options will be lost if the field type that is returned is something other than text.

Examples

`TextSizeRemove("10 Point Text and 18 Point Text")` returns **10 Point Text and 18 Point Text** displayed in the field's default font size.

`TextSizeRemove("10 Point Text and 18 Point Text";18)` returns **10 Point Text and 18 Point Text** with the 18 point font size removed from the words **18 Point Text**.

TextStyleAdd

Purpose

Adds the specified `styles` to `text` in a single action.

Format

`TextStyleAdd(text;styles)`

Parameters

`text` - any text expression or text field

`styles` - any named style listed below in Description

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

You can add multiple styles by using the + operator between style names. Negative values are not valid. All styles will be removed, if the only style specified is Plain. Plain is ignored if mixed with other styles. Styles are not case-sensitive and do not contain spaces.

Text formatting options will be lost if the field type that is returned is something other than text.

The styles that are available are:

- Plain
- Bold
- Italic
- Underline
- HighlightYellow
- Condense
- Extend
- Strikethrough
- SmallCaps
- Superscript
- Subscript
- Uppercase
- Lowercase
- Titlecase
- WordUnderline
- DoubleUnderline

- AllStyles (all available styles)

Note To format or change the case of text, use the `Lower`, `Upper`, or `Proper` function (see Text functions).

Examples

`TextStyleAdd(“Plaid”;Italic)` returns the word **Plaid** in italics.

`TextStyleAdd(FirstName;Bold+Underline)` returns **Sophie** in bold, underlined text when the FirstName field contains Sophie.

The following calculation removes all styles from the text, then italicizes the entire phrase.

```
TextStyleAdd(TextStyleAdd(FirstName;Plain);Italic)
```

The following calculation creates two descriptions of styles, then concatenates two phrases using these styles. Using the `Let` function is an effective way to avoid creating a long and complex `TextStyleAdd` statement.

```
Let([TitleStyle=Smallcaps+Titlecase;BodyStyle=Plain];
TextStyleAdd(titleField;titleStyle)&"¶¶" &
TextStyleAdd(bodyField;BodyStyle))
```

In the following example, to find every occurrence of several words and change their style, use the `Substitute` function combined with the `TextStyleAdd` function.

Substitute(ArticleBody;[“Phrase1”;TextStyleAdd(“Phrase 1”;Italic)];[“Phrase 2”;TextStyleAdd(“Phrase 2”;Bold)];)

TextStyleRemove

Purpose

Removes the specified `styles` from `text` in a single action.

Format

`TextStyleRemove(text;styles)`

Parameters

`text` - any text expression or text field

`styles` - any named style from the list of available styles

Data type returned

text

Originated in

FileMaker Pro 7.0

Description

You can remove multiple styles by using the + operator between style names. Negative values are not valid. The Plain styles cannot be used for this function. Plain is ignored if intermingled with other styles. Styles are not case-sensitive and do not contain spaces.

An additional style called AllStyles has been provided to make it easier to remove all styles. Text formatting options will be lost if the field type that is returned is something other than text.

The styles that are available are:

- Bold
- Italic
- Underline
- HighlightYellow
- Condense
- Extend
- Strikethrough
- SmallCaps
- Superscript
- Subscript
- Uppercase
- Lowercase
- Titlecase
- WordUnderline
- DoubleUnderline
- AllStyles (all available styles)

Examples

`TextStyleRemove(“Plaid”;Italic)` returns the word **Plaid** with the italics style removed.

`TextStyleRemove(FirstName;Bold + Underline)` returns **Sophie** with the bold and underlined styles removed when the FirstName field contains Sophie.

`TextStyleRemove(FirstName;AllStyles)` returns **Sophie** without any styles.

`TextStyleRemove(MyTable::MyText;HighlightYellow)`returns the text contained in **MyTable::MyText** with the HighlightYellow style removed.

Time functions

Time functions calculate times and manipulate time information.

Click a function name for details.

This function	Returns
Hour	A number representing the hour portion (0-23) of a specified time value.
Minute	A number representing the minute portion (0-59) of a specified time value.
Seconds	A number representing the seconds portion (0-59) of a specified time value.
Time	A time result with the specified number of hours, minutes, and seconds.

Hour

Purpose

Returns a number representing the hour portion (0-23) of a specified `time`.

Format

`Hour(time)`

Parameters

`time` - any time value or field of type time

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Hour(“12:15:23”)` returns **12**.

`Hour(Duration) + (Minute(Duration)/60)` returns **2.5**, when the Duration time field contains 2:30:15.

`If(Hour(HoursWorked) > 8;“Overtime Pay”;“ “)` returns **Overtime Pay** when the number of hours in HoursWorked is greater than 8.

`Hour(CheckIn)` returns **3** when the value of CheckIn is 3:24.

Minute

Purpose

Returns a number representing the minute portion (0-59) of a specified `time`.

Format

`Minute(time)`

Parameters

`time` - any time value or field of type time

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Minute(“12:15:23”)` returns **15**.

`Hour(Duration) + (Minute(Duration)/60)` returns **2.5**, if the Duration time field contains 2:30:15.

Note If no minute value is specified, 0 is returned.

Seconds

Purpose

Returns a number representing the seconds portion (0-59) of a specified `time`.

Format

`Seconds(time)`

Parameters

`time` - any time value or field of type time

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Seconds(“12:15:23”)` returns **23**.

Note If no seconds value is specified, 0 is returned.

Time

Purpose

Returns a time result with the specified number of `hours`, `minutes`, and `seconds`.

Format

`Time(hours;minutes;seconds)`

Parameters

`hours` - the hour value of a time

`minutes` - the minutes value of a time

`seconds` - the seconds value of a time

Data type returned

time

Originated in

FileMaker Pro 6.0 or earlier

Description

FileMaker Pro compensates when you supply fractional hours or minutes. The result is the time, formatted according to the time format of the field in the current layout.

Use the `Time` function or the `GetAsTime` function to enter a time constant into a formula.

Examples

`Time(4;14;32)` returns **4:14:32**.

`Time(4.5;10;30)` returns **4:40:30**.

`Time(4;15;70)` returns **4:16:10**.

Timestamp functions

Timestamps are used for a wide variety of synchronization purposes, such as marking the exact date and time at which a particular event occurred.

This function	Returns
Timestamp	A timestamp containing a calendar date and time of day.

Timestamp

Purpose

Returns a timestamp containing `date` as a calendar date and `time` as a time of day.

Format

`Timestamp(date;time)`

Parameters

`date` - any calendar date or date field

`time` - any time value or time field

Data type returned

timestamp

Originated in

FileMaker Pro 7.0

Description

The format of the result depends on the date and time formats that were in use when the database file was created. You can change the date and time formats in your operating system.

Examples

`Timestamp(Date(10;11;2014);Time(9;10;30))` returns **10/11/2014 9:10:30 AM**.

`Timestamp(Date(10;11;2014);Time(13;10;30))` returns **10/11/2014 1:10:30 PM**.

`Timestamp(Date(10;11;2014);Time(10;65;5))` returns **10/11/2014 11:05:05 AM**.

`Timestamp(Date(10;35;2014);Time(4;5;6))` returns **11/4/2014 4:05:06 AM**.

Trigonometric functions

Trigonometric functions are used to calculate degrees, angles, and other geometric data.

Note All trigonometric functions use radians as the unit of measure. Once you have a result, you can convert the radians into degrees using the `Degrees` function.

Click a function name for details.

This function	Returns
Acos	The arccosine, or inverse cosine, of a number.
Asin	The arcsine, or inverse sine, of a number.
Atan	The trigonometric arc tangent, or inverse tangent, of a number.
Cos	The cosine of the specified angle.
Degrees	Degrees, converted from the specified radians.
Pi	The value of the constant Pi, which is approximately 3.14159.
Radians	Radians, converted from the specified degrees.
Sin	The sine of the specified angle.
Tan	The tangent of the specified angle.

Acos

Purpose

Returns the arccosine (`Acos`), or inverse cosine, of `number`.

Format

`Acos (number)`

Parameters

`number` - any numeric expression or field containing a numeric expression in the range -1 to 1

Data type returned

number

Originated in

FileMaker Pro 9.0

Description

The arccosine is the angle whose cosine is `number`. The returned angle is given in radians in the range 0 (zero) to Pi. The input number parameter must be between -1 and 1.

If you want to convert the result from radians to degrees, multiply it by 180/Pi or use the Degrees function.

Examples

`Acos(-0.5)` returns **2.0943951**.

`Acos(-0.5)*180/Pi` returns **120**.

`Degrees(Acos(-0.5))` returns **120**.

`Acos(2.0)`returns ? (not a number).

Asin

Purpose

Returns the arcsine (`Asin`), or inverse sine, of `number`.

Format

`Asin (number)`

Parameters

`number` - any numeric expression or field containing a numeric expression in the range -1 to 1

Data type returned

number

Originated in

FileMaker Pro 9.0

Description

The arcsine is the angle whose sine is `number`. The returned angle is given in radians in the range -Pi/2 to Pi/2. The input number parameter must be between -1 and 1.

To express the arcsine in degrees, multiply the result by 180/Pi or use the Degrees function.

Examples

`Asin(-0.5)` returns **-0.523598776**.

`Asin(-0.5)*180/Pi` returns **-30**.

`Degrees(Asin(-0.5))` returns **-30**.

`Asin(2)`returns ? (not a number).

Atan

Purpose

Returns the trigonometric arc tangent (`Atan`), or inverse tangent, of `number`.

Format

`Atan(number)`

Parameters

`number` - any numeric expression or field containing a numeric expression

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The arc tangent is the angle, in radians, whose tangent is equal to the specified number.

Examples

`Atan(1)` returns **.78539816...**.

`Degrees(Atan(1))` returns **45**.

Cos

Purpose

Returns the cosine (`Cos`) of `angleInRadians`.

Format

`Cos(angleInRadians)`

Parameters

`angleInRadians` - any numeric expression or field containing a numeric expression, in radians

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Cos(1.047)` returns **.50017107...**.

`Cos(Radians(60))` returns **.5**.

Degrees

Purpose

Converts `angleInRadians` to degrees.

Format

`Degrees(angleInRadians)`

Parameters

`angleInRadians` - any numeric expression or field containing a numeric expression, in radians

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to translate results of trigonometric functions from radians to degrees.

$$\textbf{Degrees} = \frac{\textbf{180} \bullet \textbf{angleInRadians}}{\pi}$$

Examples

`Degrees(Atan(1))` returns **45**.

`Degrees(1.0472)` returns **60.00014030...**.

Pi

Purpose

Calculates the value of the constant Pi (π), which is approximately 3.14159.

Format

`Pi`

Parameters

None

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Pi * 15` returns **47.124**.

Radians

Purpose

Converts `angleInDegrees` to radians.

Format

`Radians(angleInDegrees)`

Parameters

`angleInDegrees` - any numeric expression or field containing a numeric expression, in degrees

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

The parameters for FileMaker Pro trigonometric functions must be expressed in radians. If the values you want to use as parameters in a trigonometric equation are in degrees, use this function to convert them to radians first. A degree is equal to Pi/180 radians.

$$\textbf{Radians} = \frac{\pi \bullet \textbf{angleInDegrees}}{\textbf{180}}$$

Examples

`Radians(45)` returns **.78539816...**.

`Sin(Radians(30))` returns **.5**.

Sin

Purpose

Returns the sine (`Sin`) of `angleInRadians` expressed in radians.

Format

`Sin(angleInRadians)`

Parameters

`angleInRadians` - any numeric expression or field containing a numeric expression, in radians

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Examples

`Sin(Radians(60))` returns **.86602**.

`Sin(.610865)` returns **.57357624...**.

Tan

Purpose

Returns the tangent (`Tan`) of `angleInRadians`.

Format

`Tan(angleInRadians)`

Parameters

`angleInRadians` - any numeric expression or field containing a numeric expression, in radians

Data type returned

number

Originated in

FileMaker Pro 6.0 or earlier

Description

Use this function to calculate the `Tan` of `angleInRadians`.

Note With the `Tan` function, you cannot use values exactly equal to 90 degrees (Pi/2 radians), or multiples of 90 degrees.

$$\textbf{Tan} = \frac{\textbf{Sin}(\textbf{angleInRadians})}{\textbf{Cos}(\textbf{angleInRadians})}$$

Examples

`Tan(.13)` returns **.13073731...**.

`Tan(Radians(34))` returns **.6745085**.

Glossary

A

Access key (Windows)

A key that activates a menu, menu item, or control when used with the ALT key. In Windows, this key corresponds to the underlined letter on a menu, command, or dialog box option.

Access privileges

Permission to view and work with certain records, fields, layouts, value lists, and scripts and to perform selected activities in a file.

Account

A username and (usually) password that accesses a file with a defined level of privileges. There are two predefined accounts: Admin and Guest. Admin is a Full Access account that can be renamed or deleted. At least one Full Access account that is authenticated via FileMaker must be defined for each database file. Guest account is a special account that cannot be renamed or deleted, but can be made active or inactive.

ActiveX Automation

A Windows programming and scripting protocol that allows external control of specific commands and actions in FileMaker Pro, including opening and closing FileMaker Pro databases, toggling the application's visibility, and performing FileMaker Pro scripts.

Alpha channel

The alpha channel of an image stores transparency information and controls the transparency of the red, green, and blue color channels.

Animation

A visual effect that provides feedback while you are performing actions in FileMaker Pro and FileMaker Go such as switching between slide panels.

API (Application Programming Interface)

A set of software application building blocks, such as data structures, variables, procedures, and functions, used by programmers.

AppleScript

A scripting language you can use to control functions of OS X and of applications that support AppleScript (often called scriptable applications).

Apple events

An OS X technology that lets applications communicate with one another. FileMaker Pro can send and receive Apple events to and from applications that support them.

Ascending sort order

Alphabetical sequence (A to Z) for words, lowest to highest order for numbers, and earliest to latest for dates and times.

ASCII character set

American Standard Code for Information Interchange. A standard character set used by computer systems worldwide (often extended for different alphabets).

Authentication

The process of checking the validity of an account and password (if one is defined) before assigning privileges and allowing access to a system or a database file. An account authenticated via FileMaker Pro or FileMaker Server is referred to as a FileMaker Account.

(FileMaker Server can also authenticate an account via External Server — an external authentication system such as Apple Open Directory, or a Windows Domain.)

Authorization

Allowing a file to access a protected file’s schema (including its tables, layouts, scripts, and value lists). Such a file is an *authorized file*.

Auxiliary files

In a FileMaker Pro Advanced runtime solution, files that are bundled with a primary file.

B

Badge

An icon indicating that conditional formatting, script triggers, or tooltips have been applied to a field, object, or layout; that find is available for a field; or that an object is a button or a popover button.

Base directory

For container fields that store an external reference, the directory in which the referenced data is stored. You can also specify a file system path or a calculation for each open storage container field.

Binding key

In FileMaker Pro Advanced, a case-sensitive key between 1 and 24 characters long that links the components of a runtime solution.

Blank layout

A layout that contains empty body, header, and footer parts. (In previous versions of FileMaker Pro, a Blank layout was a predefined layout type.)

Body part

A layout part that contains individual records from a database file.

Book

In the status toolbar, the navigation control for moving from one record to another in Browse mode, from one layout to another in Layout mode, from one request to another in Find mode, and from one page to another in Preview mode.

If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Boolean value

A Boolean value is either True or False. A field containing any number except zero evaluates as True. A field containing zero, no data, or content that does not resolve into a number evaluates as False. For example, a field containing “ABC,” “ABC0,” or an empty field is False. A field containing “1” or “ABC2” is True.

Break field

In a subsummary part, records are grouped (sorted) by values in another field, called the break field. Whenever the value of the break field changes, the report “breaks” and FileMaker Pro inserts the subsummary part.

Browse mode

The FileMaker mode in which you enter and edit information in fields. Groups of fields make up the records of your database. You can view one record at a time (click **Form View** in the layout bar), view your records in a list (click **List View**), or view records arranged in a spreadsheet-like table (click **Table View**).

(Use Browse mode to enter and edit your information; use Layout mode to design how your information is displayed. Use Find mode to find records that match search criteria; use Preview mode to display how your records will print.)

Button

A layout object that performs a specified script in Browse or Find modes.

C

Cache

The amount of memory assigned to FileMaker Pro. A larger cache size increases performance. A smaller cache size saves data to the hard disk more frequently, offering greater protection in case of a system crash.

Calculation field

A field that returns the result of a calculation of values. You can create a formula for the calculation using functions, constants, operators, and information from other fields in the same record.

Cascading style sheets

A system of codes or tags that define how a web browser displays information in a web page. Cascading style sheets provide more control over the layout and appearance of web pages than HTML. Cascading style sheets work like templates for web pages. If a web page contains cascading style sheets, users must view it in a browser that supports cascading style sheets.

CGI (Common Gateway Interface)

The specification for communication between an HTTP server and server gateway programs, which is supported by most servers.

Character encoding

The character set or code page of a file. If necessary, you can specify a character set to be used when importing, exporting, indexing, sorting, and spell-checking files. FileMaker supports ASCII, Windows ANSI, Macintosh, Japanese (Shift-JIS), Unicode UTF-8, Unicode UTF-16, and Unicode UTF-16 Windows.

Chart

A graphical depiction that makes it easy to compare data and see patterns and trends.

Chart legend

A key that identifies the colors assigned to a data series in a chart.

Client

A user that opens a database file that is shared on a network, published in a browser, or shared via ODBC/JDBC. FileMaker Network settings and privileges determine how FileMaker Pro and FileMaker Go clients interact with databases hosted through FileMaker Pro, and FileMaker Server.

Client application

The application that requests data (using SQL) from a data source (using ODBC or JDBC). Also, FileMaker Pro is a client application when it accesses a database hosted by FileMaker Server.

Client/server architecture

The relationship between two networked computers that share resources. The client requests services from the server, and the server provides services to the client.

Clipboard

A temporary storage area in computer memory where FileMaker Pro places the most recent selection you've cut or copied.

Clone

A copy of a FileMaker Pro file that contains all the field definitions, tables, layouts, scripts, and page setup options, but none of the data.

Column

When a database file is viewed as a table, a column corresponds to a field.

Combo box

A type of drop-down list you can set up in Layout mode. In the **Data** tab of the Inspector, select **Include arrow to show and hide list**. The list will only drop when users click the arrow, not when they enter the field.

Commit

To save changes to a database file. Certain actions such as navigating between records, finding, and sorting do not change the file's modification date. Other actions, such as changing data in a record or changing a layout do change the file's modification date.

Consistency check

A consistency check examines a file that may have been damaged due to, for example, an unexpected quit. FileMaker Pro reads every file block and verifies that the internal structure of the block is valid and the block is properly linked to other blocks in the file. (A consistency check does not read all the data within each file block or check the schema or higher-level structures in the file; these tasks are performed by file recovery.)

Constant

In a formula, an unchanging value. For example, a constant can be a field name, a text literal ("Total:"), or a number. The value of the constant doesn't change from record to record as the formula is evaluated. Text constants in formulas can be up to 253 characters long.

Container data type

Pictures, movies, and documents, such as Microsoft Word files and PDF files can be inserted in a container field. Data in container fields can be embedded in the field, stored by file reference, or stored externally.

Context

The table occurrence from which calculations and scripts are begun, and from which a relationship is evaluated in the relationships graph.

Convert

Opening a data file from another application, which creates a new FileMaker Pro file containing the data.

Also refers to opening a file created with a previous version of FileMaker Pro.

Custom function

A function that is not one of the default FileMaker Pro functions. In FileMaker Pro Advanced, you can create custom functions that can be reused anywhere in the database.

Custom menu

A menu that is not one of the default FileMaker Pro menus. In FileMaker Pro Advanced, you can create custom menus, menu items, and menu sets.

D

Data Entry Only privilege set

One of the three predefined privilege sets that appear in every file. The Data Entry Only privilege set allows read/write access to the records in a file, but not design access (for example, the ability to create layouts and value lists).

Data point

In a chart, the value of data plotted on the x or y axis displayed as a column, bar, point, bubble, or slice.

Data series

Data points that are plotted in a chart. When multiple data series are charted, each data series is displayed in a unique color and is defined in the chart's legend, if included.

Data source

A named reference that provides access to another FileMaker database file or to an ODBC database. ODBC data sources are also referred to as external SQL sources (ESS).

In charting, the origin of the data to be charted (current found set, current record (delimited data), or related records).

Data source name (DSN)

A data structure that contains the information about a specific database that an ODBC driver needs in order to connect to it.

Data Viewer

A FileMaker Pro Advanced feature that lets you monitor expressions such as field values, local and global variables, and calculations. You can monitor expressions while running scripts or while testing them in the Script Debugger. You can also monitor field values and variables in the database file.

Database Design Report

A FileMaker Pro Advanced tool that creates a report of your database schema.

Database Encryption

A form of encryption that protects "data at rest." For FileMaker, data at rest is a FileMaker Pro database file, and its temporary files, while it is being stored on disk (and not open). An encrypted database is protected from unauthorized access with an encryption password. FileMaker Pro Advanced is required to encrypt database files. Encrypted database files can be decrypted and re-encrypted as needed.

Database file

A collection of information in a file containing one or more tables pertaining to a subject, such as customers or invoices. (A large database can also comprise many database files.)

Database Management System (DBMS)

An application that allows users to store, process, and retrieve information in a database.

Descending sort order

Reverse alphabetical sequence (Z to A) for words, highest to lowest order for numbers, and latest to earliest dates and times.

Developer Utilities

A FileMaker Pro Advanced feature that lets you bind files into a runtime solution, display files in Kiosk mode, prevent users from modifying the design or structure of databases, and automatically rename sets of files and update links to related files.

Dialog window

A document window in a modal state. Its behavior is similar to the behavior of a window when a running script is paused. You can create and customize a dialog window using the New Window script step.

Document window

A standard modeless FileMaker window. You can create and customize a document window using the New Window script step.

Domain name

The primary subdivision of internet addresses, which is indicated by the last part of an internet address after the final period (or dot). In the United States, the standard domains are .com, .edu, .gov, .mil, .org, and .net. In other countries, the top-level domain is usually the country domain.

Domain name server (DNS)

A server that matches up the URL of a website (for example http://www.filemaker.com) with its proper numeric IP address (for example 12.34.56.78).

Driver

The ODBC or JDBC driver translates SQL queries into commands that a DBMS can understand. It processes ODBC/JDBC calls, submits SQL requests to the data source, and returns the data back to the driver manager, which then routes it to the requesting application (for example, FileMaker Pro).

Driver manager

The control panel that manages communication between requesting applications and data sources. When an application makes a request via ODBC/JDBC, the driver manager routes the request through the correct driver to the correct data source and returns the data to the requesting application. All ODBC/JDBC drivers and data sources to be used on that computer are registered with the driver manager.

Drop-down calendar

A field set up to display an interactive monthly calendar when a user enters the field in Browse or Find mode.

DTD (Document Type Definition)

A formal description of a particular type of XML. It defines a document structure, including the names of data elements and where they may occur within the structure. Valid XML conforms to the rules established in its DTD.

Dynamic guides

In Layout mode, guidelines that extend horizontally and vertically to help you move, resize, or align objects. Dynamic guides also “snap-to” the upper and lower boundaries and the centers of objects as you move, resize, or align the objects.

E

Email

Electronic mail. A system for transmitting messages from one computer or terminal to another. A message sent from one computer user to another is stored in the recipient's account mailbox until that person logs onto the system and reads the message.

Embedded

For container fields, a file that is stored in the container field. Container data can also be stored by reference or stored externally.

Encryption password

The password required to open an encrypted file. For FileMaker Pro, the encryption password is required for the FileMaker Pro user who opens the file, but not for the client who opens a file hosted by FileMaker Pro. FileMaker Pro Advanced is required to enable, disable, or change an encryption password.

Envelope layout

A predefined layout with fields arranged for printing on standard business envelopes.

Ethernet

A type of fast local area network used for connecting computers and peripherals within the same building or campus.

Export

To save data from one file so that it can be used in another file or in another application.

Expression

A value or any computation that produces a value. Expressions can contain functions, field values, and constants and can be combined to produce other expressions.

Extended privilege

Data sharing permissions that determine if a privilege set allows users to open a shared file using FileMaker Pro or FileMaker Server (fmapp); view a database as an ODBC or JDBC data source (fmxdbc); view a database using a web browser via FileMaker WebDirect (fmwebdirect), XML web publishing (fmxml), or PHP web publishing (fmphp); or access a file in FileMaker Go without having to re-login (fmreauthenticate10). Plug-ins from third-party developers may provide additional extended privileges.

External function

A function written in C or C++ as part of a third-party plug-in that extends the feature set of FileMaker Pro or FileMaker Pro Advanced.

External script

A script used by a database file, but defined in a different database file. Use the Perform Script script step to select a defined script from a related file, or to select a file reference to a database file on your hard drive or network.

F

Field

The basic unit of data in a record. You define a field to hold a specific, discrete category of data, such as Last Name, Employee Photo, or to display the result of a calculation. You can define text, number, date, time, timestamp, container, calculation, and summary fields. Field can also refer to the object on a layout that displays the data, such as an edit box, checkbox set, or pop-up menu.

Field boundary

In Layout mode, an outline that shows the size of a field. To see field boundaries, choose **View** menu > **Show** > **Field Boundaries**. These boundaries only appear in Layout mode. However, you can use the **Appearance** tab of the Inspector to format objects to have borders that do appear in Browse mode and when you print the layout.

Field label

Text on a layout that identifies a field. When you place a field on a layout, you can have FileMaker Pro add a field label that matches the field name. You can change or delete this field label if you want.

Field name

The name you assign to a field when you define the field. When you place a field onto a layout, you can have FileMaker Pro also place an editable field label that matches the field name. Fully qualified fields are displayed in `tablename::fieldname` format.

Field type

The part of a field definition that determines what kind of data you can enter in the field and the kinds of operations FileMaker Pro can perform with the data. FileMaker Pro can create text, number, date, time, timestamp, container, calculation, and summary fields. (Global fields contain the same value for all records in the database and can be of any type except summary.)

File path

The location of a file in an operating system as identified by the drive, folders, filename, and file extension.

FileMaker Network

A communications method built into FileMaker Pro that allows you to share files hosted by FileMaker Pro or FileMaker Server with others over a network or in FileMaker Go. The FileMaker Network settings and privileges you set up determine how other users (called “clients”) can open and use the shared file.

FileMaker WebDirect

A web application for accessing layouts from database files in a web browser. Web clients use FileMaker WebDirect to access layouts from database files hosted by FileMaker Server.

Find mode

The FileMaker mode in which you specify criteria for finding a subset of records.

(Use Browse mode to enter and edit your information; use Layout mode to design how your information is displayed. Use Find mode to find records that match search criteria; use Preview mode to display how your records will print.)

Find request

In Find mode, a blank form based on the current layout. Enter search criteria into one or more fields of the find request.

Firewall

A security system used to prevent unauthorized users from gaining access to a LAN. A firewall usually has a single computer that is connected to the internet and all internet traffic must pass through that computer.

Floating document window

A standard modeless FileMaker window that stays on top of other windows while users work in FileMaker Pro. You can create and customize a floating window using the New Window script step.

Footer part

Use the footer part for page numbers or dates. This part appears at the bottom of every screen or page (unless you add a Title Footer). You can have only one footer in a layout. A field in the footer displays data from the last record on that page.

Form View

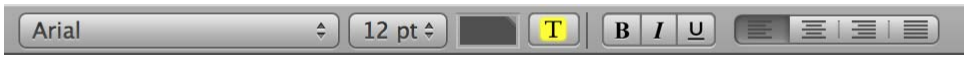
Displays one record at a time. By default, fields appear on separate lines. To select this view, click **Form View** in the layout bar.

If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Formatting bar

In Layout and Browse modes, the area that displays options for formatting text and objects. To show or hide the formatting bar, click **Formatting** in the layout bar.

Formatting bar in Layout mode



If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Formula

A set of instructions that FileMaker Pro follows to calculate a value used in a field or as the criteria for matching database records.

Found set

The set of records in a table that are made active when you search for data. When you find all records, the found set is the entire table.

Full Access privilege set

One of the three predefined privilege sets that appear in every file. The Full Access privilege set allows complete read/write access to a file, including making changes to privileges for the file.

Fully qualified name

The complete name of a field or layout, expressed using the format `tableName::[field or layout name]`, where “tableName” is the name of the underlying table occurrence in the relationships graph upon which the field or layout is based. A fully qualified name identifies an exact instance of a field or layout. Because fields and layouts with common names can be based on different tables, FileMaker Pro uses fully qualified names to avoid errors in calculations and scripts.

Function

A predefined, named formula that performs a specific calculation and returns a single, specific value.

Function list separator

The punctuation character ; (a semicolon) that separates parameters in a function definition. If you type a comma (,), FileMaker Pro automatically changes it to a semicolon after you close the Specify Calculation dialog box.

G

GIF (Graphics Interchange Format)

A platform-independent file format often used to distribute graphics on the internet.

Global field

A field defined with the global storage option can contain one value that's used for all records in a file. Use the value of a global field as a fixed value in calculations, to declare variables in If or Loop script steps, or for fields that rarely need to be updated (for example, a company logo in a container field). A global field can be any field type except summary. A global field can't be indexed.

Global variable

A global variable can be used in a calculation or script anywhere in a file, for example, other scripts or file paths. The value of a global variable is not cleared until the file is closed.

Grammar

A precise description of a formal language, such as XML, consisting of sets of rules for how strings (words) in the language can be generated, and how the strings can be recognized as part of the language.

Grand summary

Total or other aggregate value for all records in the found set.

Grand summary part

Use grand summary parts to view and display summary information (totals, averages, and so on) in summary fields for all records in the found set. You can add one grand summary part at the top (leading) and one grand summary part at the bottom (trailing) of a layout.

Grid

In Layout mode, a series of nonprinting intersecting horizontal and vertical lines. The grid aligns objects you create, resize, move, or position. Objects “snap to” the grid to help you create and edit objects more precisely. Gridlines adjust when you change the unit of measure.

Grouped object

A collection of objects that behaves as one object in Layout mode.

Guest

A user who opens a protected file without specifying an account name and password. The Guest account is assigned a privilege set that determines what guests can do in the file. Guest access may be disabled for a file.

Guides

Nonprinting, movable horizontal and vertical guidelines to help you position and align objects in Layout mode. An object's left or right boundary, top or bottom boundary, or center “snaps to” a guide.

H

Handle

One of the small squares surrounding a selected object, and used to resize and reshape the object.

Header part

Use a header part for column headings, titles, and other information that appears only at the top of every page on a layout. FileMaker Pro displays the header in Browse mode and prints it on every page, except the first page if you add a title header. Fields added to a header are printed on every page, using data from the first record on that page.

Homepage

The starting page for a website. It often has some form of a table of contents that allows viewers to link to other parts of the website.

Host

After a file has been opened and enabled for sharing, the host is either the first FileMaker Pro user to share the file, or the host is FileMaker Server. Once the host opens the file, other users (clients) can access and change the file. All changes are stored in the file on the computer or device where the file resides. FileMaker Network settings and privileges determine how FileMaker Pro and FileMaker Go clients interact with databases hosted through FileMaker Pro or FileMaker Server.

HTML (Hypertext Markup Language)

A language that is used for displaying and accessing information on the World Wide Web.

HTTP (Hypertext Transfer Protocol)

The internet protocol that defines how a web server responds to requests for files.

I

Import

To bring (copy) data from a table, another file, or another application into the current table. You can also import scripts from one FileMaker Pro file into another.

Indexing

An option that can be enabled when defining (or changing) the definition of a field. When indexing is enabled, FileMaker Pro builds a list of all the values that occur in the field in the table. This improves the performance of tasks such as finding data, but it increases the size of the database file on disk.

Inspector

In Layout mode, a tool that allows you to view and edit the settings for objects. You can open multiple inspectors to view and format settings in different tabs at the same time. To open the Inspector, choose **View** menu > **Inspector**. To open another Inspector window, choose **View** menu > **New Inspector**.

Interactive container

A container field for which the **Interactive content** option in the Inspector is selected. Interactive containers are rendered using the web browser technologies that web viewer uses.

Internet

An international network of many other networks that are linked using the TCP/IP network protocol.

Internet Service Provider (ISP)

The company from which you purchase your connection to the internet.

Intranet

A private TCP/IP network of linked computers within a company or organization.

IP (Internet Protocol) address

For IPv4, a four-part number, usually formatted as 12.34.56.78, that uniquely identifies a computer on the internet; for IPv6, an eight-part number, usually formatted as [2001:0DB8:85A3:08D3:1319:8A2E:0370:7334], that uniquely identifies a computer on the internet. When referenced in an application, IPv6 addresses must be contained in square brackets, for example, [2001:0DB8:85A3:08D3:1319:8A2E:0370:7334].

J, K

JDBC

A Java API that uses SQL statements to access data from, and exchange data with, many database management systems. The JDBC driver communicates between your Java applet and the FileMaker Pro or FileMaker Server data source.

JPEG (Joint Photographic Experts Group)

A platform-independent file format often used to distribute graphics on the internet.

Key

A column (or columns) that makes a particular row unique (corresponds to a match field).

Kiosk

A FileMaker database that runs full screen, without toolbars or menus. Users click buttons to navigate. In FileMaker Pro Advanced, use the Developer Utilities to create Kiosk solutions. You can bind Kiosk solutions into stand-alone runtime solutions.

L

Labels layout

A predefined layout with fields arranged for printing horizontally or vertically on mailing label stock, and media and index sheets.

LAN (local area network)

A connection between computers within a location using cable or a wireless system.

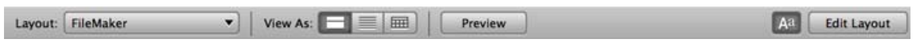
Layout

An arrangement of fields, objects, pictures, and layout parts that represents the way information is organized and presented when you browse, preview, or print records. You can design different layouts for entering data, printing reports and mailing labels, displaying web pages, and so on.

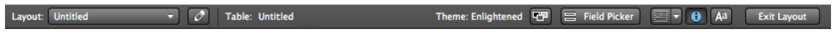
Layout bar

In the status toolbar, the area that displays options for working with layouts, such as the Layout pop-up menu.

Layout bar in Browse mode



Layout bar in Layout mode



If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Layout mode

The FileMaker mode in which you determine how information in fields is presented on the screen and in printed reports.

(Use Browse mode to enter and edit your information; use Layout mode to design how your information is displayed. Use Find mode to find records that match search criteria; use Preview mode to display how your records will print.)

Layout part

A section of a layout that organizes or summarizes information. Layout parts include Body, Header, Footer, Title Header, Title Footer, leading and trailing Grand Summary, and leading and trailing Subsummary.

Layout pop-up menu

In the layout bar, a pop-up menu from which you can choose **Manage Layouts** (Layout mode) or a defined layout (all modes).

If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Layout theme

A collection of coordinated styles that determine the color, object, part, and background attributes, and the fonts used on a layout. Themes also enhance the appearance of a layout or report and give all your layouts a consistent look. A theme does not control the placement or behavior of fields or objects on a layout. A theme is assigned when you create a new layout, but you can change the theme in Layout mode. You can also create a theme by changing the styles used for the layout, and then saving the theme with a new name.

Layout tools

In the status toolbar in Layout mode, a collection of tools that includes the Selection tool (pointer), Text tool, Line tool, Rectangle tool, Rounded Rectangle tool, Oval tool, Field/Control tool, Button tool, Popover Button tool, Tab Control tool, Slide Control tool, Portal tool, Chart tool, Web Viewer tool, Field tool, Part tool, and Format Painter tool.

If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Layout types

FileMaker Pro includes several predefined types of layouts and reports to display on different types of devices (such as laptop computer screens or touch device screens), for different purposes (such as browsing records, entering data, or printing reports, mailing labels, or envelopes). After you choose a layout type, you can make additional choices.

To use a predefined layout type, in Layout mode, click **New Layout/Report** in the status toolbar. The assistant guides you through creating the type of layout or report you want. After you finish the assistant, use the tools and commands in Layout mode to tailor the layout for your needs.

LDAP (lightweight directory access protocol)

A protocol for accessing online directory services.

Link

On a web page, text or a graphic which — when you click it — displays an associated web page or a specific element within a page.

Also, the HTML code that creates a link to another web page or to a specific element within a page.

List View

Displays records one record at a time in a list format. To select this view, click **List View** in the layout bar.

If you don't see the status toolbar, click the status toolbar control button ▭ at the bottom of the document window.

List view layout

A type of Report layout in which fields that you specify appear in columns across the screen or page in one line. Field names are in the header part and the footer part is blank. (In previous versions of FileMaker Pro, a List view layout was a predefined layout type.)

Locked object

An object on a layout that cannot be edited or deleted. To lock or unlock an object, select it in Layout mode. In the Inspector, click **Position**, then choose **Lock** or **Unlock** in the Arrange & Align area. If an object is locked, its selection handles dim.

If you don't see the Inspector, choose **View** menu > **Inspector**.

Lookup

A lookup matches records and copies data from a related table into a field in the current table. After data is copied, it becomes part of the current table (as well as existing in the table it was copied from). Data copied to a table doesn't automatically change when the values in the related table change.

Lookup target field

The field that you want data copied to during a lookup.

Lookup source field

The field in the related table that contains the data you want copied during a lookup.

M

Many-to-many relationship

A correspondence between data in database tables in which more than one record in the first table is related to more than one record in another table, and more than one record in that table is related to more than one record in the first table.

Match field

For relational databases and lookups, a field in a source table and a field in a related table that contain values you want to use to find matching records. (A match field is sometimes called a key field or trigger field.) In the relationships graph, match fields appear in italics.

For importing records, values in the match fields determine which records in the source table update which records in the target table.

Menu

A list of menu items. Each menu has a title that appears on the menu bar.

Menu bar

The area at the top of the window (Windows) or screen (OS X) that displays the installed menu set.

Menu item

One item listed in a menu on the menu bar. A menu item corresponds with one command, submenu, or separator.

Menu item properties

All the settings for a menu item, including platform, display title, shortcut, and action.

Menu set

The collection of menus that installs on the menu bar.

Merge field

A placeholder on a layout for the contents of a database field. A merge field expands or contracts in Browse and Preview modes, or when printed, to fit the amount of data in the database field for each record.

Merge fields are useful for mail merge form letters; FileMaker Pro uses merge fields in predefined Labels and Envelope layouts.

Merge variable

A variable inserted onto a layout that displays values from a local or global variable onto the layout of the current record. You see the merge variable value in Browse, Find, and Preview modes, and when you print records.

Modal window

A window that requires user action before users can continue working in FileMaker Pro. Use a modal window to display alert messages or to collect information from users.

Mode

In FileMaker Pro, the four different environments (Browse, Find, Layout, and Preview) that you use to work with your database file.

Mode pop-up menu

A pop-up menu at the bottom of the document window from which you can choose a mode (Browse, Find, Layout, or Preview). This menu is available in all modes.

Multi-key field

A match field that contains more than one value, each on a separate line. A multi-key field can be used in one table involved in a relationship, to match several possible values in the match field of the other table.

Multimedia

Files that combine media, like text, graphics, sound, animation, and video.

N

Network protocol

A network protocol (for example, TCP/IP) is a set of rules that govern how computers exchange messages on a network.

New Layout/Report assistant

The New Layout/Report assistant guides you through creating a layout or report according to options you choose. In Layout mode, click **New Layout/Report** in the status toolbar, or choose **Layouts** menu > **New Layout/Report**.

O

Object

On a FileMaker Pro layout, an object is a discrete entity or shape that you can select, move, modify, delete, or name. Lines, fields, buttons, popovers, panel controls, portals, imported graphics, blocks of text, tab controls, and web viewers are objects.

ODBC

An API that uses SQL statements to access data from, and exchange data with, many database management systems. FileMaker Pro uses ODBC drivers to share data (as a data source) and to interact with data from other applications (as a client application).

1-away relationship

A correspondence between database tables in which two tables are directly related to each other, with no other tables between them.

One-to-many relationship

A correspondence between data in database tables in which one record in the first table is related to more than one record in another table.

One-to-one relationship

A correspondence between data in database tables in which one record in the first table is related to one record in another table.

Operands

Components of a formula. For example, in the formula Quantity*Price, Quantity and Price are the operands.

Operators

In calculations, symbols that indicate how to combine two or more expressions. These include the standard arithmetic operators (+, -, /, *), logical operators that set up conditions that must be met to make a value True or False (AND, OR, XOR, and NOT), and find operators (<, =, @) that help you limit the records defined in a find request.

In the relationships graph, symbols that define the match criteria between one or more pairs of fields in two tables. These include: != (not equal), > (greater than), < (less than), = (equal), <= (less than or equal to), >= (greater than or equal to) and x (all rows, or cartesian product).

P

Panel control

Panel controls include the tab control and slide control. Individual panels of these controls are tab panels and slide panels, respectively.

Parent script

A script that defines script parameters and can call other scripts.

Part label

In Layout mode, the label that appears at the left or side of the bottom dividing line of each layout part. By dragging it up or down, you can use the part label to resize a part. You can also open the Part Definitions dialog box by double-clicking the label.

PHP (PHP: Hypertext Preprocessor)

An open-source programming language primarily used in server-side application software to create dynamic web pages. FileMaker Server lets you publish data from FileMaker Pro databases on customized web pages created with PHP.

Plug-in

Software that extends the capabilities of an application in a specific way.

Popover

A layout object that can contain other layout objects. Popovers include a content area (where objects are placed) and can include a title. You open a popover by clicking a popover button. Popovers reposition on the screen as needed to stay in view.

Popover button

A layout object that opens a popover.

Port

A pre-assigned number that indicates a “logical connection place” where a client (such as a web browser) can connect to a particular server application on a networked computer. Port numbers range from 0 to 65536. Port 80 is the default port for HTTP services such as FileMaker Pro web publishing, but you can use another port number if 80 is already in use by another server application.

Portal

For relational databases, a layout object in one table where you place one or more related fields to display in rows the data from one or more related records.

Preview mode

The FileMaker mode in which you see how layouts will look when they're printed.

(Use Browse mode to enter and edit your information; use Layout mode to design how your information is displayed. Use Find mode to find records that match search criteria; use Preview mode to display how your records will print.)

Primary file

In a FileMaker Pro Advanced runtime solution, the file that connects all of the auxiliary files and opens when you start the runtime application. From the Developer Utilities in FileMaker Pro Advanced, you can select a primary file for solutions that have more than one file.

Privilege set

A defined set of permissions that determines a level of access to a database file. You can define as many privilege sets as you like for a file. There are three predefined privilege sets: Full Access, Data Entry Only, and Read-Only Access.

Q

Query

Retrieving, manipulating, or modifying data from a data source by sending SQL statements. Also, requesting, and then receiving, data from a DBMS. You can also add, edit, format, sort, and perform calculations on your data using queries.

Quick find

In Browse mode, searches records across multiple fields on a layout.

QuickTime

An application from Apple Inc. that compresses, stores, and plays files combining text, sound, animation, and video.

QuickTime VR

A type of QuickTime movie. QuickTime VR movies let you view panoramic images or objects from many angles.

R

Read-Only Access privilege set

One of the three predefined privilege sets that appear in every file. The Read-Only Access privilege set allows read access to the records in a file, but not write or design access.

Record

One set of fields in a database table. Each record contains data about a single activity, individual, subject, or transaction.

Recover

If a file is damaged (for example, from an unexpected quit during a hard-drive update), you can attempt to recover the file. FileMaker Pro preserves as much data (the file's schema and structure and its tables, records, layouts, scripts, and field definitions) as possible.

Recurring import

A method of setting up a file to automatically import read-only data from another file. With recurring import, imported data refreshes when you open the file, view the layout that contains the imported data for the first time during a FileMaker Pro session, or run a data update script.

Recursive script

A script that calls itself.

Related field

For relational databases, a field in one table that is related to a field in another table (or to a different field within the same table). If a relationship is defined between two tables (even through another table), data in fields in one table can be accessed from the other table.

Related record

A record in the related table whose match field (according to the relationship used) contains a value that's equal to the value in the match field of another table.

Related table

For relational databases, the table that contains the data you want to access and work with in the current table. For lookups, the table that contains the data to copy.

Relational database

A group of one or more database files that, when used together, contain all the data you need. Each occurrence of data is stored in only one table at a time, but can be accessed in any table, either in the same file or from a related file. Data from another table or file is displayed in the current table without being copied, and the data changes whenever the values in the other table or file change.

Relationship

Relationships provide access to data from one table to another. Relationships can join one record in one table to one record in another table, one record to many other records, or all records in one table to all records in another table, depending on the criteria you specify when you create the relationship in the relationships graph.

Relationships graph

In the Relationships tab of the Manage Database dialog box, you can see the occurrences of tables both in the current file and from any external, related database files. In this relationships graph, you join tables and change relationships between fields in different tables.

When you create a new table, a visual representation, or occurrence, of the table appears in the relationships graph. You can specify multiple occurrences (with unique names) of the same table in order to work with complex relationships in the graph.

Repeating field

A field containing multiple, separate values.

Report layout

A predefined layout type for setting up reports. You can create a layout with simple rows and columns of data (as in a list), or a complex report with grouped data (subsummary reports). The fields that you specify appear in columns across the screen or page in one line. Field names are in the header part and the footer part is blank.

Report with grouped data

A subsummary report that you create using the Report layout type. Reports with grouped data can include totals and subtotals.

Row

When a database file is viewed as a table, a row corresponds to a record.

Runtime solution

A database that does not require FileMaker Pro or FileMaker Pro Advanced in order to be used. In FileMaker Pro Advanced, use the Developer Utilities to bind a primary file and any auxiliary files to produce a stand-alone runtime solution.

S

Schema

In database terminology, a schema is the organization of the tables, the fields in each table, and the relationships between fields and tables.

Screen stencils

In Layout mode, nonprinting guides that help you design layouts for iPad, iPhone, iPod touch, or computers with different screen resolutions.

Script

One or more instructions (script steps) that you define to automate repetitive or difficult tasks. You manage scripts using the Manage Scripts feature. You run a script by clicking its button, choosing its menu command, calling it from another script or a plug-in, or running it at startup or when a file closes.

Script Debugger

A FileMaker Pro Advanced tool for debugging FileMaker Pro scripts.

Script step

A command that you include in a script.

Script trigger

A mechanism that causes a specified script to run when a particular event occurs.

Search criteria

In Find mode, the values and operators you specify to locate records. For example, if you type ABC Travel in the Vendor field, FileMaker Pro looks for and returns all records that have this name in the Vendor field.

Security

The protection that's placed on a file. Security includes various types of accounts to authenticate users, levels of privilege sets to determine what can be done with a file, and authorization of other files to create references to the current file (including its tables, layouts, scripts, and value lists). Security also includes extended privileges, which determine the data sharing options that are permitted by a privilege set.

Self-join

A relationship between fields in the same table. This creates another occurrence of the table in the relationships graph.

Separator

A line within a menu that separates or groups menu items.

Serial number

A unique number entered by FileMaker Pro for each record. You can tell FileMaker Pro to automatically enter a serial number for each record by setting the Auto-Enter options in the Options for Field dialog box. You can also serialize records in Browse mode by choosing **Records** menu > **Replace Field Contents**.

Shared database

A database file for which sharing has been enabled, which permits users to access the database file over a network. FileMaker Pro, FileMaker Pro Advanced, and FileMaker Server each support one or more of the following ways to share databases: FileMaker Network sharing, which permits multiple FileMaker Pro or FileMaker Go users to use a database file simultaneously; publishing of databases to web browser users via FileMaker WebDirect or Custom Web Publishing; and sharing of data with other applications via ODBC/JDBC.

Shared ID

In FileMaker Pro Advanced, a case-sensitive ID between 1 and 32 characters long that links encrypted database files in a multi-file solution.

Shortcut

Also known as keyboard shortcut. One or more keys that users can press to perform tasks.

Shortcut menu

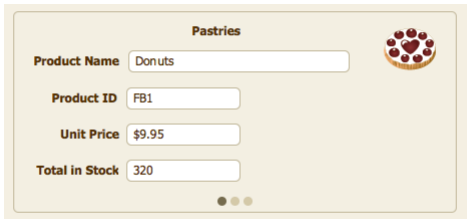
Use to edit objects or data quickly by choosing commands from a shortcut, or context, menu. Commands vary depending on the mode you're using, the item the cursor is over, and whether an item is selected.

To display a shortcut menu, Right-click (Windows) or Control-click (OS X) the item.

Slide control

A layout object made up of one or more slide panels, allowing you to organize fields and other objects within each slide panel's borders.

Slide control in Browse mode



Pastries

Product Name	Donuts
Product ID	FB1
Unit Price	$9.95
Total in Stock	320

Slide panel

A component of a slide control. The slide panel is the area displayed when a dot in a slide control is selected. You can place objects such as lines, fields, buttons, portals, imported graphics, blocks of text, tab controls, slide controls, and web viewers in slide panels.

Slider

In the status toolbar, the navigation control for quickly moving to a record in your database file based on its location in the file. For example, in Browse mode, move the slider to the left to go to the first record and to the right to go to the last record.

In Browse mode, moving the slider changes the current record. In Find mode, moving the slider changes the current find request. In Layout mode, moving the slider changes the current layout. In Preview mode, moving the slider changes the current page.

Sliding objects

Objects that move together to close gaps left by entries in adjacent fields.

Set sliding in Layout mode, in the Sliding & Visibility area of the Inspector.

If you don't see the Inspector, choose **View** menu > **Inspector**.

Snapshot link

A found set of records that is saved in the FileMaker Pro Snapshot Link (FMPSL) format, with the filename extension .fmpsl. A snapshot link captures and preserves the found set as it was when you performed the find request. You can also email an FMPSL file to another person.

Solution

A database file or files.

Sort order

The sequence for rearranging records. Records are sorted by the first field in the sort order list, then the second, and so on. Values within each field are sorted by the order specified (ascending, descending, or custom).

Source file

The file from which you bring data during importing or exporting, or the file from which you add a table to the relationships graph.

Source table

The table upon which one or more table occurrences in the relationships graph are based. The source table is the table defined in the Tables tab of the Manage Database dialog box.

SQL

A structured programming query language that controls and interacts with a DBMS.

Stacking order

The order in which objects overlap on a layout. In Layout mode, you can change this order by cutting and pasting objects or by clicking **Bring to front**, **Bring forward**, **Send to back**, or **Send backward** in the Arrange & Align area of the Inspector.

If you don't see the Inspector, choose **View** menu > **Inspector**.

Standard form layout

The default layout, with all fields arranged on separate lines in the order they were defined. The body part is only as tall as it needs to be to include all the fields in the database. This layout includes header and footer parts. (In previous versions of FileMaker Pro, a Standard Form layout was a predefined layout type.)

Starter Solution

Or template. A predesigned and formatted FileMaker Pro file, or web page, that you can copy and change for your own use.

Startup script

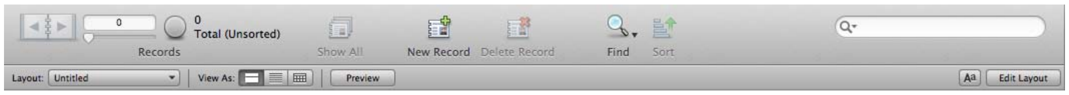
A script that automatically runs when a file is opened. You can script such things as setting system formats to the user's formats or setting a database to be shared in a startup script.

You specify a startup script in the File Options dialog box.

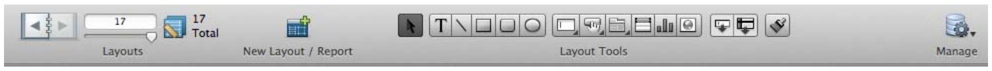
Status toolbar

The area across the top of the document window that displays navigation controls, customizable buttons, and a layout bar for working with layouts. In Layout mode, it includes layout tools.

Status toolbar in Browse mode (Windows)



Status toolbar in Layout mode (OS X)



If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

Structure

In FileMaker Pro, the organization of file elements such as scripts, layouts, value lists, privileges, and so on. You interact with a file's schema through its structure.

Submenu

A menu that extends from another menu item.

Sub-script

A script that is called from another script.

Subsummary parts

Use summary parts to view and display information from one or more records. You place a summary field in a summary part to display a summary of information for each group of records sorted on the break field. You can add one or more subsummaries above (leading) or below (trailing) the body.

Subsummary value

Aggregate values for different categories of data within a field. For example, a subsummary value can be the total of employees for each department.

Summary field

A field that contains the result of a summary calculation of values across a group of records.

Supplemental field

A FileMaker calculation field or summary field that you can append to ODBC tables in order to do calculations on the external data while working in FileMaker. The calculations are not stored and you are not changing the schema of the ODBC table.

System formats

Settings you control with control panels to determine how dates, times, currency, and numbers display and sort on your computer. (See the documentation that came with your computer for information on using these control panels.)

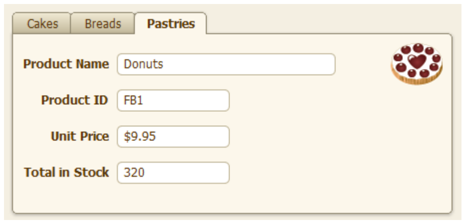
If the system formats are different on your computer from the ones on the computer where the database file was created, the first time you open the file, FileMaker Pro will ask you if you want to use the system's settings or the file's settings. Using the Inspector, you can also format number fields to display decimals and thousands separators according to the current system formats.

T

Tab control

A layout object made up of one or more tab panels that allows you to organize fields and other objects within each tab panel's borders.

Tab control and tab panels in Browse mode



Product Name	Donuts
Product ID	FB1
Unit Price	$9.95
Total in Stock	320

Tab order

The order in which you move from field to field in a record. In Layout mode, you can define a custom tab order and include buttons, tab controls, and web viewers in the tab order.

Tab panel

A component of a tab control. The tab panel is the area displayed when a tab in a tab control is selected. You can place objects such as lines, fields, buttons, portals, imported graphics, blocks of text, tab controls, slide controls, and web viewers in tab panels.

Table

A collection of data pertaining to a subject, such as customers or stock prices. A database file contains one or more tables, which consist of fields and records. When you create a new table, a visual representation, or occurrence, of the table appears in the relationships graph. You can specify multiple table occurrences (with unique names) of the same table in order to work with complex relationships in the graph.

Table View

Displays multiple records in a tabular format like a spreadsheet. Each record appears in a row, and each field appears in a column. To select this view, click **Table View** in the layout bar.

If you don't see the status toolbar, click the status toolbar control button at the bottom of the document window.

In Browse mode, you can use Table View to create, modify, and delete fields; choose field types; add, delete, and sort records; or create a chart or dynamic report.

Target file

The file into which you bring data during importing.

TCP/IP (Transmission Control Protocol/Internet Protocol)

The basic communication protocol that is the foundation of the internet.

Template

A predefined website that you can select in the Web Viewer Setup dialog box to help you create a web viewer quickly.

Text baseline

In Layout mode, the guideline that appears at the base of the text in a field or text block. Text baselines can be solid, dotted, or dashed. If you want text baselines to also appear in Browse and Find modes, select **Text baselines** in the **Appearance** tab of the Inspector.

If you don't see the Inspector, choose **View** menu > **Inspector**.

Text expression

Any expression that returns a text result. For example, a text expression can be a constant ("London"), a field reference (Status), or a calculated value (Rightwords(Lastname;1)).

Timestamp

A field type combining date and time that is compatible with the ODBC requirement for the SQL format [yyyy.mm.dd hh:mm:ss.sss].

Tooltip

A small box that displays text when a user moves the cursor over a layout object. Tooltips display in Browse, Find, and Layout modes.

U

Unicode

A worldwide standard that, in one code page, provides a unique number for every character in human languages, no matter what the platform, software program, or operating system.

Unit of measure

In Browse and Layout modes, you can set the unit of measure to points, inches, or centimeters.

Unstored calculation

A calculation field with a result that is only calculated when the value is needed, for example, to browse or print. In most cases, FileMaker Pro makes a field stored when you define it, but you can change the storage type to unstored.

URL (Uniform Resource Locator)

A web address, which consists of a protocol, a host name, and optionally a port, a directory, and a filename. For example, http://www.filemaker.com/ , ftp://12.34.56.78:80/myfiles/, or fmp://mywebsite.com/sample.fmp12.

V

Value list

To save time and ensure accuracy during data entry, define frequently used text, number, date, or time values as a value list. When you enter data, you can choose from the list of defined values.

You can format value lists to display in a drop-down list or pop-up menu, or as checkboxes or option (radio) buttons. The values in a value list can be user-defined or based on the values in a field in the same file or in a different file. You can also define relationships for use with value lists, to access and display particular related values. Another option is to use a value list from another file.

Variable

In a calculation, a symbol or name that represents a value. Use the Set Variable script step to specify the name, value, and repetition of the variable. Names prefixed by $ are local variables available only within the current script. Prefix the name with $$ to make the variable available throughout the current file (global). Local and global variables can have the same name but they are treated as different variables.

View

An arrangement of your data primarily useful for onscreen manipulation. In Browse mode, Find mode, or Preview mode, Form View displays individual records, List View displays records in a list, and Table View displays records in a spreadsheet-like table format.

W, X, Y, Z

Web address

The calculated expression that you enter in the Web Viewer Setup dialog box. A web address is not equivalent to a URL that a web user could enter in a web browser.

Web browser

An application that you can use to view web pages/sites on the World Wide Web or an intranet. Browsers download the web pages onto the viewer's computer.

Web page

An HTML document displayed on the internet or on an intranet.

Web server

A computer that is connected to the internet or an intranet and has a web server application installed on it. Web server applications deliver web pages and associated files to web browsers.

Website

One or more web pages connected by links and displayed on the internet or on an intranet.

Web user

Someone using a web browser to access a FileMaker Pro database published on the World Wide Web or an intranet.

Web viewer

A layout object that allows you to display information from websites based on data in your database.

World Wide Web

An interlinked collection of web pages residing on web servers, and other documents, menus, and databases, which are available via URLs.

X-axis data

In a column, stacked column, line, and area chart, the data series you are comparing (for example, company name).

In a bar or stacked bar chart, the data series you are measuring (for example, annual sales).

XSLT (Extensible Stylesheet Language Transformations)

XSLT (XSL Transformations) is a subset of XSL (Extensible Stylesheet Language) that is used to transform, or change, the structure of an XML document into a different document format. For example, you can use an XSLT style sheet to transform an XML document into an HTML or TXT document.

XML (Extensible Markup Language)

Instead of being a rigid file format, XML is a language for defining agreed-upon formats that groups can use for exchanging data. Many organizations and businesses are using XML to transfer product information, transactions, inventory, and other business data.

FileMaker Pro can export XML data that can then be used, for example, by spreadsheet applications, data charting applications, and enterprise SQL databases. FileMaker Pro can also import XML data.

Y-axis data

In a column, stacked column, line, and area chart, the data series you are measuring (for example, annual sales).

In a bar or stacked bar chart, the data series you are comparing (for example, company name).