

TROI NUMBER PLUG-IN 1.1 USER GUIDE

July 2001



Troi Automatisering
Vuurlaan 18
2408 NB Alphen a/d Rijn
The Netherlands
Fax: +31-172-470539

You can also visit the Troi web site at: <<http://www.troi.com/>> for additional information.

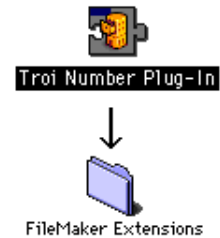
Table of Contents

- Installing plug-ins..... 1**
- If You Have Problems..... 1**
- What can this plug-in do?..... 2**
- Getting started 2**
 - Using external functions..... 2**
 - Where to add the External Functions?..... 2**
 - Simple example..... 3**
- Summary of functions..... 3**
- Using the Balance Functions 4**
 - What the Balance functions can do 4**
 - Defining Balance Functions 5**
- Function Reference 7**
 - Tnum-BalanceCalc 7**
 - Tnum-BalanceResult 8**
 - Tnum-BalanceStart 9**
 - Tnum-Version10**

Installing plug-ins

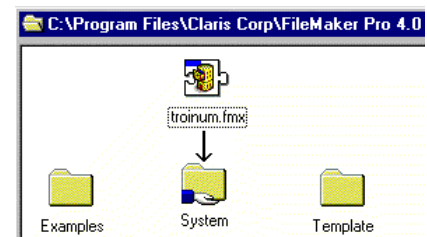
For Macintosh:

- Quit FileMaker Pro.
- Put the file "Troj Number Plug-in" from the folder "Macintosh Plug-in" into the "FileMaker Extensions" folder in the FileMaker Pro folder.
- If you have installed previous versions of this plug-in, you are asked: "An older item named "Troj Number Plug-In" already exists in this location. Do you want to replace it with the one you're moving?" Press the OK button.
- Start FileMaker Pro. The Troj Number Plug-in will display a dialog box, indicating that it is loading and showing the registration status.



For Windows:

- Quit FileMaker Pro.
- Put the file "troinum.fmx" from the directory "Windows Plug-in" into the "SYSTEM" subdirectory in the FileMaker Pro directory.
- If you have installed previous versions of this plug-in, you are asked: "This folder already contains a file called 'troinum.fmx'. Would you like to replace the existing file with this one?" Press the Yes button.
- Start FileMaker Pro. The Troj Number Plug-in will display a dialog box, indicating that it is loading and showing the registration status.



TIP You can check which plug-ins you have loaded by going to the plug-in preferences: Choose **Preferences** from the **Edit** menu, and then choose **Plug-ins**.

You can now open the file "Number Examples.fp5" to see how to use the plug-in's functions. There is also a Function overview in this file.

IMPORTANT There is a problem in FileMaker Pro 4.0v1. Please make sure that all plug-ins that are in the folder "FileMaker Extensions" are enabled in the preferences. (Under Edit/ Preferences/ Application/ Plug-ins). Make sure all plug-ins have a cross before their name. Remove plug-ins you don't use from the "FileMaker Extensions" folder.

If You Have Problems

This user guide tries to give you all the information necessary to use this plug-in. So if you have a problem please read this user guide first. If that doesn't help you can get free support by email. Send your questions to support@troi.com with a full explanation of the problem. Also give as much relevant information (version of the plug-in, which platform, version of the operating system, version of FileMaker Pro) as possible.

If you find any mistake in this manual or have a suggestion please let us know. We appreciate your feedback!

TIP You can get more information on returned error codes from our OSerrrs database on our web site: <http://www.troi.com/software/oserrrs.html>. This free FileMaker database lists all error codes for Windows and Mac OS!

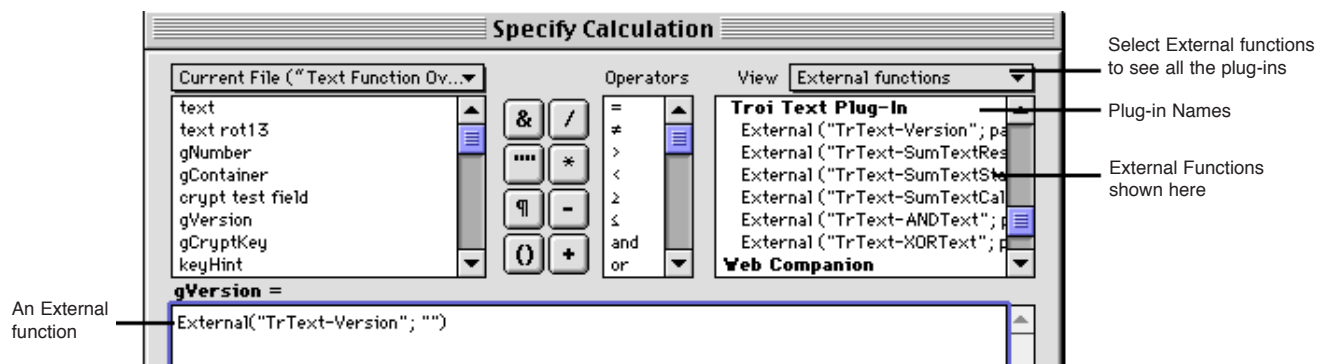
What can this plug-in do?

The Troi Number Plug-in adds the Balance function to FileMaker Pro. With this plug-in you can create a dynamic number field, that shows the running balance of a number of records from a field in a related file. Everytime records get added to the related file the balance is automatically updated.

Getting started

Using external functions

The Troi Number Plug-in adds new functions to the standard functions that are available in FileMaker Pro. The functions added by a plug-in are called external functions. You can see those extra functions for all plug-ins at the top right of the Specify Calculation Box:



You use special syntax with external functions: External("function name", parameter) where function name is the name of an external function. The parameter is required, even if it's only "". Plug-ins don't work directly after installation. To access a plug-in function, you need to add the calls to the function in a calculation for example in a text calculation in Define Fields or in a ScriptMaker Script.

IMPORTANT In the United States, commas act as list separators in functions. In other countries semicolons might be used as list separators. The separator being used depends on the operating system your computer uses, as well as the separator used when the file was created. All examples show the functions with commas. For example: External("Tnum-Version", "") will become External("Tnum-Version"; "") in such a file.

Where to add the External Functions?

External functions for this plug-in can be used in a calculation field when you are defining fields (choose Define from the File menu). Also the some of the plug-in's functions can be used in a script step using a calculation, for example in a Set Field script step.

IMPORTANT The Balance functions have to be used in a specific way, to create the desired effect. See the section on Balance functions for the specifics on this.

Simple example

The core of this plug-in are the 3 balance functions. However these functions are not so simple to start with. So we'll first given an example with the Version function:

Say you have a database Test.fp5, with a global text field called gErrorCode. Create a new ScriptMaker Script called "Show Flash Dialog". In this script add the following step:

```
Set Field[ gErrorCode, External("Tnum-Version", "-ShowFlashDialog")]
```

This will show the flash dialog of the plug-in, and also shows the registration state.

NOTE Function names, like "Tnum-Version" are case sensitive. Be sure to spell them right, or get them from the External Functions list at the top right of the "Specify Calculation" dialog.

Please take a close look at the included example files, as they provide a great starting point. From there you can move on, using the functions of the plug-in as building blocks. Together they give you a new and powerful tool for your database.

Summary of functions

The Troi Number Plug-in adds the following functions:

| <u>function name</u> | <u>short description</u> |
|----------------------|--|
| Tnum-BalanceStart | Begins balance calculation |
| Tnum-BalanceCalc | Performs the actual balance calculation |
| Tnum-BalanceResult | Returns the result of the balance calculation |
| Tnum-Version | Use this function to see which version of the plug-in is loaded and to register the plug-in. |

Using Balance Functions

What the Balance functions can do

The purpose of the 3 Balance functions is to get a dynamic balance in a portal. The 3 functions (Tnum-BalanceStart, Tnum-BalanceCalc and Tnum-BalanceResult) work together to achieve this result. Before describing how to define Balance fields we give you an example of what is possible:

Example

Say you have a file "Customers.fp5" holding customer information and "Transactions.fp5" holding financial transactions. Customers.fp5 has a relation "Customer Transactions" that relates all the transactions of each customer. This shows some data for a customer:

| <u>Date</u> | <u>Transaction description</u> | <u>Amount</u> |
|-------------|--------------------------------|---------------|
| May 1 | Deposit | 100.00 |
| May 29 | Pie | -8.00 |
| July 4th | Ice Cream | -1.00 |

With the Balance functions it is possible to define a field that calculates the running balance. After defining the proper fields with the balance function you can get this data to show in your portal:

| <u>Date</u> | <u>Transaction description</u> | <u>Amount</u> | <u>Balance</u> |
|-------------|--------------------------------|---------------|----------------|
| May 1 | Deposit | 100.00 | 100.00 |
| May 29 | Pie | -8.00 | 92.00 |
| July 4th | Ice Cream | -1.00 | 91.00 |

And if a new transaction is added (or deleted) the balance is recalculated automatically. So this may result in the following new portal data:

| <u>Date</u> | <u>Transaction description</u> | <u>Amount</u> | <u>Balance</u> | |
|-------------|--------------------------------|---------------|----------------|---|
| May 1 | Deposit | 100.00 | 100.00 | |
| May 15 | Chicago Rib Shack | -49.50 | 50.50 | <i>this transaction was added later</i> |
| May 29 | Pie | -8.00 | 42.50 | |
| July 4th | Ice Cream | -1.00 | 41.50 | |

This even works with sorted portals (like in this case the transactions are sorted on date).

Defining Balance Functions

Assume you have a main file "Clients.fp5" and a related file "Transactions.fp5". The Clients.fp5 file has a relation ClientTransactions which shows all the Transactions to a client. These are the steps to make it work:

1- In your related file define a calculation field.

Go to "Define Fields" in Transactions.fp5 and create a new calculation field. Here this field is named balanceCalc. Use the function Tnum-BalanceCalc in this calculation:

```
balanceCalc = TextToNum(External("Tnum-BalanceCalc", TransactionAmount & "|" & NumToText(ID)))
```

Here TransactionAmount is the name of the field you want to create a dynamic balance of and ID is a unique serial number for each transaction record.

IMPORTANT Make sure you make this field both unstored and a number, otherwise the Balance function won't work!

2- In the main file define a field for the total of the balance.

Use the following calculation:

```
totBalanceCalc = TextToNum(
  External("Tnum-BalanceResult", External("Tnum-BalanceStart" ,
  Middle(NumToText(3/2) , 2 , 1) ) &
  Sum(ClientTransactions::balanceCalc))
```

This calculation looks quite complex. You don't need to understand how it works, you can also just create it.

WHAT HAPPENS HERE?

We will go through it step by step: The part `Middle(NumToText(3/2), 2, 1)` is just a clever way to get the decimal separator. On US systems this is a period ".", in Europe this is usually a comma ",". This part will evaluate for US systems as follows:

```
Middle(NumToText(3/2), 2, 1) = Middle(NumToText(1.5), 2, 1) = Middle("1.5", 2, 1) = "."
```

and for European systems (which use the comma as separator):

```
Middle(NumToText(3/2); 2; 1) = Middle(NumToText(1,5); 2; 1) = Middle("1,5"; 2; 1) = ","
```

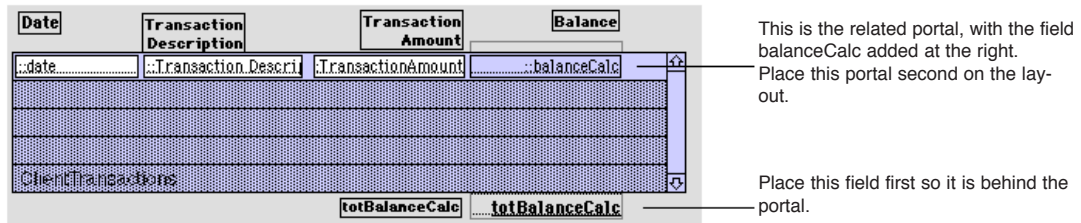
So for US systems the calculation becomes:

```
totBalanceCalc = TextToNum(
  External("Tnum-BalanceResult", External("Tnum-BalanceStart" , "." ) &
  Sum(ClientTransactions::balanceCalc))
```

The function Tnum-BalanceStart is evaluated first, it signals the plug-in to start balancing. It takes the decimal separator as parameter. Then the part `Sum(ClientTransactions::balanceCalc)` is evaluated. This part makes FileMaker call all the related balanceCalc fields. The plug-in can then calculate the proper values of the balance. Tnum-BalanceResult signals to the plug-in that all has been balanced and the result can be returned.

3- In the main file create a layout with a portal and the totBalanceField

Create (or modify) a layout in the main file. First put the `totBalanceCalc` on the layout. Then put a portal on the layout, based one the same relation as the `totBalanceCalc` field uses. In our case this is the `ClientTransactions` relation. In the portal put your fields and also the newly created field `balanceCalc`.



IMPORTANT The balance function only works if the `totBalanceCalc` field is **VISIBLE** on the window and on the layout order it must be arranged **BELOW** the balance portal. This way it is evaluated before the `balanceCalc` field.

Why this way?

You might say, why not make this simply one function like:

```
External ("Tnum-Balance", RelationName::NumberField).
```

This would be easier, but this is not possible with the current Plug-in implementation of FileMaker.

Function Reference

Tnum-BalanceCalc

Syntax Set Field [result, External("Tnum-BalanceCalc", "numberfield")]

See “Using the balance functions” for an overview of how to use Balance.

Parameters

The NumberField is the field you want to balance.

Returned result

the current balance value.

Example usage

Use this function to define a balanceCalc field in a related file like this:

balanceCalc = Calculation, Unstored, Number, = TextToNum(External("Tnum-BalanceCalc"; NumberField)). The NumberField is the field you want to balance. Please make sure you make it both unstored and a number, otherwise this won't work!

Tnum-BalanceResult

Syntax Set Field [result, External("Tnum-BalanceResult", "don't care")]

See "Using the balance functions" for an overview of how to use Balance.
Use this function to stop the balance calculation and get the resulting Balance.

Parameters

Use the calculation as specified in "Using the balance functions" as a parameter. This is not used directly, but ensures that the result is correct.

Returned result

Tnum-BalanceStart

Syntax Set Field [result, External("Tnum-BalanceStart", "decimal separator")]

See "Using the balance functions" for an overview of how to use Balance.
Use this function to start the Balance calculation.

Parameters

The decimal separator is used by the plug-in to know how which separator to return. This depends on where the file was created.

You can do this automatically by using the following formula as parameter:

Middle(NumToText(3/2) , 2 , 1)

3/2 will give 1,5 or 1.5 depending on the setting. So taking the 2nd character will produce the decimal separator.

When no parameter is specified it defaults to ".".

Returned result

Tnum-Version

Syntax Set Field [result, External("Tnum-Version", "")]

Use this function to see which version of the plug-in is loaded.

Note: This function is also used to register the plug-in.

Parameters

switches *determine the behaviour of the function*

switches can be one of this:

-GetString *the version string is returned (default)*

-GetVersionNumber *returns the version number of the plug-in*

-ShowFlashDialog *shows the Flash Dialog of the plug-in (returns 0)*

If you leave the parameter empty the version string is returned.

Returned result

The function returns "" if this plug-in is not loaded. If the plug-in is loaded the result depends on the input parameter. It is either a:

VersionString:

If you asked for the version string it will return for example "Troi Number Plug-in 1.1"

VersionNumber:

If you asked for the version number it returns the version number of the plug-in x1000. For example version 2.0.1 will return number 2010.

ShowFlashDialogResult:

This will return the error code 0.

Special considerations

IMPORTANT Always use this function to determine if the plug-in is loaded. If the plug-in is not loaded use of external functions may result in data loss, as FileMaker will return an empty field to any external function that is not loaded.

Example usage

External(Tnum-Version, "") will return "Troi Number Plug-in 1.1".

Example 2

External("Tnum-Version", "-GetVersionNumber") will return 1100 for version 1.1.

External("Tnum-Version", "-GetVersionNumber") will return 1201 for version 1.2b1

External("Tnum-Version", "-GetVersionNumber") will return 2130 for version 2.1.3

To use a feature introduced with version 1.1 test if the result is bigger than 1100.