Formulas

User Guide

Note: This documentation is an app that is still a work in progress, so it may document features that have not been incorporated yet, or features that have been changed.

If you come across a part this is not clear, or needs more information, please let me know through the feedback form on http://www.donkeyengineering.com/formulasapp/.

For Formulas, Version 0.2
November 18, 2008
http://www.donkeyengineering.com/formulasapp/

Table of Contents

Formulas Overview

The goal of Formulas is to be simple but powerful calculator for OS X. It is the second iteration of the PEMDAS Dashboard widget -- it will take some of the widget's features further, and it will introduce some new features that are not easy to do with the web technologies behind Dashboard.

Keep in mind that this app is still a work in progress, so it is less feature complete than usual. If there is a feature you would like to to see, or you find a bug, please provide feedback at http://www.donkeyengineering.com/formulasapp/.

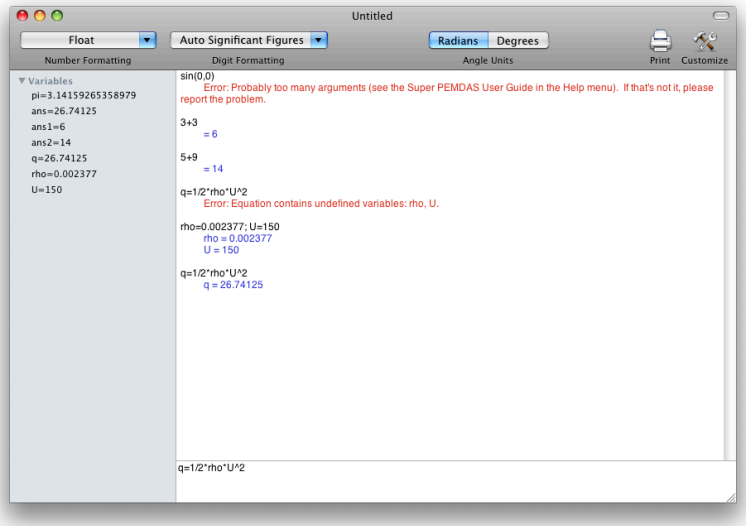
If you would like to get started, see the Getting Started section.

Features Overview

Formulas includes many of the same features as the Formulas widget. As the program matures, features will gradually be added.

Variable Cache

On the side of the application is the variable cache. This lets you easily see all of the variables active in the current document.



Number Formatting

Formulas has several different kinds of number formatting, including float, scientific, and engineering. It also lets you specify the number of decimals or significant figures to display, and lets you add thousands separators.

See the Number Formatting section for more information.

Cocoa Based

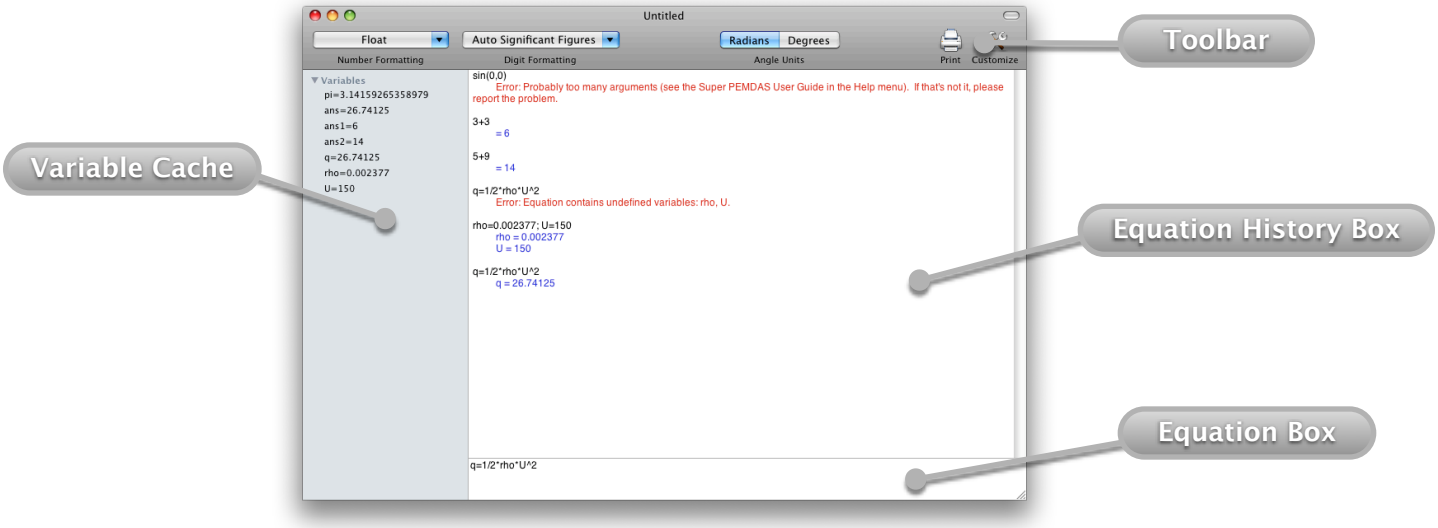
Everything in Formulas is written in Objective-C and C, and for the most part the interface is composed of standard Cocoa controls. This way, you get many of the Cocoa benefits, such as spell check, customizable toolbars, and text areas that behave in a standard fashion. It is also a universal binary, and you don't need to install or fiddle with extra runtime environments to get it to load. For more information, see the Technical Details section.

What's New

This is the initial release.

The User Interface

Note that the current user interface is only temporary. It is easier to manage from a development point of view, but it will eventually will be replaced.



Variable Cache

The variable cache shows all the defined and undefined variables in the document.

Equation Box

The equation box is where you type your equation. When you want it to be calculated, hit the "enter" key on your keyboard. To make the box bigger, drag the line on the top of the box.

Equation History Box

The equation history box shows all the equations you've entered, and the associated results.

Toolbar

The toolbar contains useful buttons. You can customize the look of the toolbar by control clicking (or right clicking) on it, then selecting "Customize Toolbar".

Typing Equations

You can type equations in the Equation Box. You can type multiple equations if you separate them with a semicolon. When you are finished and want to calculate the equation, hit the enter (or return) key.

Equation Syntax

Following are some syntax examples For information on how equations are calculated, see the Order of Operations section.

Multiple Equations

Multiple equations can be typed on the same line. Simply separate them with a semicolon.

Example:

3+2; 4+5;

Equals Sign

The equals sign equates one side of an equation to the other.

Example:

a=27+2

Order of Operations

Order of operations is calculated left to right, in the following order:

1) Parenthesis
2) Exponents [1]
3) Multiplication or Division
4) Addition or Subtraction

[1] Exponents are calculated right to left. See the Exponents Calculation section below for an explanation.

Exponents Calculation

Exponents are a special case of order of operations -- they are evaluated right to left, rather than left to right.

For example, if you typed 2^3^2, the visual equivalent would be:

$$2^{3^2}$$

If you then started to reduce the exponents, you would first evaluate 3^2, so it would look like:

$$2^{3^2} = 2^9$$

In other words, you start at the right side and work your way left.

On the other hand, if the calculation engine went from left to right, it would evaluate 2^3 first. However:

$$2^{3^2} \neq 8^2$$

It should be noted that a surprisingly large percentage of math programs and handheld calculators do not follow this rule. Instead, they will evaluate 2^3^2 as 8^2, or 64. Therefore, when using other calculators, it is always a good idea to check to see how it works.

Supported Functions

Trigonometric Functions

For trigonometric functions, you can set whether you want to work in Radians or Degrees. Setting the working mode to radians or degrees can be done in the toolbar, and it can also be set within the Formulas Preferences.

Be careful, radians is selected by default.

Available functions:

- **sin**(x) = Sine of **x**.
- **cos**(x) = Cosine of **x**.
- **tan**(x) = Tangent (sin(x)/cos(x)) of **x**.
- **csc**(x) = Cosecant (1/sin(x)) of **x**.
- **sec**(x) = Secant (1/cos(x)) of **x**.
- **cot**(x) = Cotangent (1/tan(x)) of **x**.
- **asin**(x) = Arcsin of **x**.
- **acos**(x) = Arccos of **x**.
- **atan**(x) = Arctan of **x**.
- **atan2**(y,x) = Arctan of y and x, where y and x are coordinates of the angle vector. This bypasses the quadrant limitations of **atan()**, and returns an angle from -pi to pi (-180º to 180º).

Hyperbolic Functions

Note: if you are working in degrees, x will be converted to radians before calculation. i.e., in degree mode, sinh(90) will be evaluated as sinh(pi/2).

- **sinh**(x) = Hyperbolic sine of **x**.
- **cosh**(x) = Hyperbolic cosine of **x**.
- **tanh**(x) = Hyperbolic tangent of **x**.
- **csch**(x) = Hyperbolic cosecant (1/sinh(x)) of **x**.
- **sech**(x) = Hyperbolic secant (1/cosh(x)) of **x**.
- **coth**(x) = Hyperbolic cotangent (1/tan(x)) of **x**.

Exponential & Logarithmic Functions

Depending on Formulas preferences, these functions may change.

- **`exp(x)`** = Notation for raising e to the `x`.
- **`ln(x)`** = Natural log of `x`.
- **`log(x)`** = Log base 10 of `x`.

Other Functions

- **`sqrt(x)`** = Takes the square root of `x`.

Number Formatting

In Formulas, there are several different ways to format numbers and answers. Note that formatting is just that; it only changes what is *displayed*. In other words, if you have the significant figure precision set to 2, and you assign variable 'a' the value 5.4321, it will be displayed as 5.4. However, all calculations will use the 5.4321 value.

General Formatting

There are several different ways that Formulas can format answers:

- **Float**: Leaves the number alone.
- **Scientific**: Puts the number in scientific notation.
- **Engineering**: Puts the number in engineering notation, which is similar to scientific except the exponents are shown in multiples of 3. This is very convenient when working in Metric.

Significant Figures

You can set the maximum amount of significant figures to be displayed. Note that if the number has fewer significant figures than the amount you specify, Formulas will not add on extra significant figures.

Example

With the preference set to 4 significant figures:

123456.78
=123500

0.012345
=0.01235

0.012
=0.012

Decimal Places

You can set the exact amount of decimals to be displayed. Note that if the number has fewer decimal places =than the amount you specify, Formulas will add extra 0's.

Example

With the preference set to 4 decimal places:

123456.78
= 123456.7800

0.012345
=0.0123

0.012
=0.0120

Thousands Separators

Thousand separators are only available when the number is formatted as a Float. Currently, only commas are available as thousands separators

Example

123456.78
= 123,456.78

Variables

Formulas has support for variables. You can set a variable using the equals operator; anything to the left of the equals is the variable, to the right of the variable is what the variable is assigned. So typing:

a=3+5

will set the variable a to 8. You can then use 'a' in any equation, so 2*a will give you 16.

Cached Answers

Variables will show up in the answer/answer cache. In fact, all the auto-cached answer variables... ans1, ans2, ans3, etc. are just variables. You can re-assign them like any other variable. If you assign ans3 to something before Formulas is at ans3, Formulas will skip over it and go to the next available variable in the auto-cache format.

If you assign a variable, the answer is not cached in an ans{number} type variable (like ans1), since that would be redundant. However, the 'ans' variable will still be the result of the calculation. So after doing something like a=3*4, both 'a' and 'ans' will be 12.

If you assign a variable ans9=4, and Formulas was not at 'ans9' in the answer caching yet, Formulas will skip over 'ans9' when it gets there, and go to the next available answer.

Variable Names

Variable names are case sensitive, must be alphanumeric (A-Z, a-z, 0-9), and cannot start with numbers.

Locked Variables

Some variables, like 'ans' and 'pi', are locked. ('ans' is locked because as long as there was not a calculation error, 'ans' will always be the last answer you calculated. It would be bad to have Formulas start changing variables on you, without you realizing.)

All function names (sin, e, exp, etc.) are locked.

Overview

From a development point of view, there are two primary components of Formulas: the user interface and the calculation engine. Both portions are written in Objective-C.

The User Interface

The user interface is based on standard OS X controls to speed development and to make it blend in with the rest of the system.

The Calculation Engine

The calculation engine is the portion of the program which handles all the equations, formatting, and variables. It is proprietary and closed-source.

History

Development was started on the engine in early 2006 using Javascript. It went through several releases as part of the PEMDAS Dashboard widget, and in early 2007 it was ported to Objective-C. In mid-2007, to speed up the development process, garbage collection was turned on and it was made Leopard-only. In 2008, the calculation engine was fine tuned to run on the iPhone/iPod Touch platform, and internally a lot of code was rewritten to take advantage of Objective C 2.0 features, so it is much faster and will allow for more expandability in the future.

Design Goals

The goal of the calculation engine is not to be computationally fast. Instead, the goal is to allow the user to think more about the calculations being worked on, and how to communicate those results, rather than thinking about how to trick the calculator to do what they want it to do.

Precision

The calculation engine uses double (64 bit) precision, which can hold approximately 16 decimals and has upper/lower bounds of approximately $\pm 10^{309}$.

Release Notes

Following are the release notes for each release.

Version 0.2 (November 18, 2008)

- Finally updated the program.
- Renamed the app from Super PEMDAS to Formulas, updated links to point to the new website, updated the documentation.
- Incorporated latest revision of the Formulas calculation engine (which is faster and has a lot of bug fixes).
- The toolbar now autosaves the configuration it is in.

Version 0.1 (December 20, 2007)

- Initial release.