Equations

User Guide

Note: Equations is still a work in progress, so this documentation may document features that have not been incorporated yet, or it may document 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/equationsapp/.

For Equations, Version 0.2.2
January 17, 2008
http://www.donkeyengineering.com/equationsapp/

Table of Contents

Equations Overview

Equations is being designed as a simple but powerful calculator for Mac OS X. It will improve upon the PEMDAS Dashboard widget by expanding existing features and by introducing completely new ones.

Keep in mind that this application is still a work in progress. If there is a feature you would like to to see, or you find a bug, please provide feedback at http://www.donkeyengineering.com/equationsapp/.

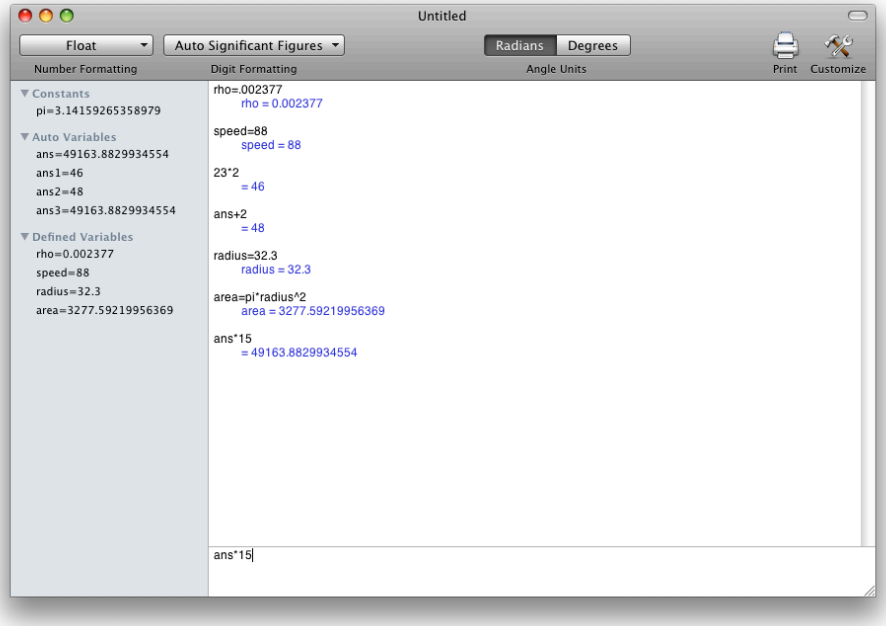
If you would like to get started, see the Getting Started section.

Features Overview

Equations already has many of the same features as the PEMDAS Dashboard widget, and more features will gradually be added as the application matures.

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.



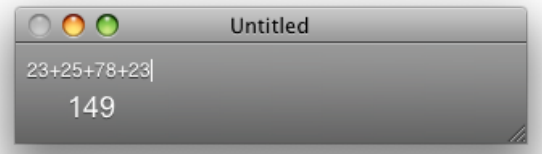
Number Formatting

Equations has several different kinds of number formatting, including float, scientific, engineering, percents, binary, octal, and hexadecimal . It also lets you limit the number of decimals or significant figures to display, and lets you add thousands separators.

See the Number Formatting section for more information.

Mini Window

Equations can resize into a much smaller window for quick calculations, or for typing in data from other applications. Click on green zoom button in the upper left hand corner of the window to use this feature.



Cocoa Based

Equations is written in Objective-C, and the interface is composed of standard Cocoa controls. This way, you get the benefits of a Cocoa application, such as customizable toolbars, and text areas that behave in a standard fashion. It is also a universal binary, and it has drag and drop installation.

For additional information on how the application is written, see the Technical Details section.

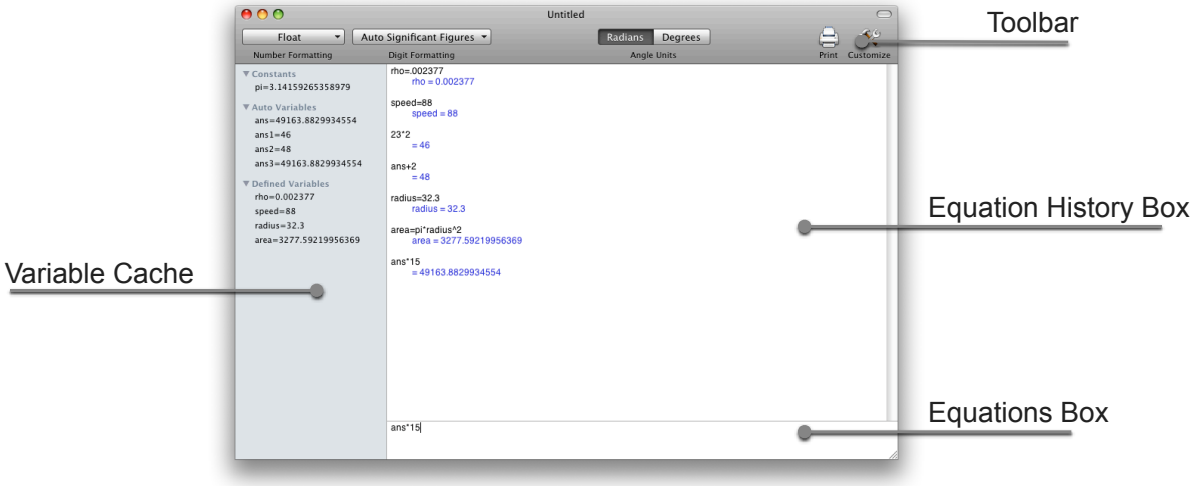
Requirements

To use Equations, you must have:

- a Mac computer with an Intel or PowerPC processor.
- Mac OS X v10.5.6 Leopard or later.
- 5 MB of available disk space.

The User Interface

The Equations interface is relatively simple. Note that the current interface is only temporary since it makes it easier to develop the calculation engine, and will eventually be replaced.



Variable Cache

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

Equations Box

This text box is where you type your equation. When you want it to be calculated, hit the "return" 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 calculated, and their results.

Toolbar

The toolbar contains several useful buttons. You can customize the look of the toolbar by right clicking on it and selecting "Customize Toolbar".

Typing Binary, Octal, and Hexadecimal Numbers

Equations inputs numbers in base 10. However, you can also enter binary, octal, and hexadecimal numbers by using the following prefixes, which are case insensitive:

- **0b**: Binary number prefix.
- **0o**: Octal number prefix.
- **0x**: Hexadecimal number prefix.

After you input a number, you can also format the outputted number. For more information on how to do this, see the Number Formatting section.

Examples

Binary numbers (note the answers are shown in base 10):

```
0b0101
    = 5

0b1111
    = 15
```

Octal numbers (note the answers are shown in base 10):

```
0o77
    = 63

0o100
    = 64
```

Hexadecimal numbers (note the answers are shown in base 10, and also note that the hexadecimal number itself is case insensitive):

```
0xFF
    = 255

0x1a
    = 26
```

Note that you can use different bases in the same equation:

```
0xFF-0b10100110
    = 89

0o27*0x1B+0b10
    = 623
```

Typing Equations

You can type equations in the Equations Box. You can type multiple equations if you separate them with a semicolon. * *(see below)* 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 * *(see below)*

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
```

* *Development Note: typing equations with a semicolon is currently disabled while the calculation engine undergoes changes. It will be re-enabled in an upcoming release.*

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

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

To understand why, an example is that if you typed `2^3^2`, it would translate visually to:

$$2^{3^2}$$

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 would calculate it right to left.

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

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

It should be noted that many other math programs and handheld calculators do not follow this rule. Instead, they will incorrectly 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 Equations 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 Equations 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 Equations, there are several different ways to format numbers and answers. Note that formatting does not change the actual number, it only changes what appears on the screen. In other words, if you have the significant figure precision set to 2, the number 5.4321 will be displayed as 5.4. However, all calculations will use the 5.4321 value.

General Formatting

There are several different ways that Equations 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.
- **Percent**: Puts the number into "percent" form, which is similar to Float, except the number has been multiplied by 100.
- **Binary**: Puts the number into binary (base 2) form. The number is prefixed with 0b. [1]
- **Octal**: Puts the number into octal (base 8) form. The number is prefixed with 0o. [1]
- **Hexadecimal**: Puts the number into hexadecimal (base 16) form. The number is prefixed with 0x. [1]

[1] Note that when Binary, Octal, or the Hexadecimal formatting is applied to a non-integer number, it will be floored prior to formatting. When this happens, a (Floored) notice will appear after the number. See below for examples.

Examples

Using the Float preference:

```
123456.78
    = 123456.78

0.012345
    = 0.012345
```

Using the Scientific preference:

```
123456.78
    = 1.2345678e5

0.012345
    = 1.2345e-2
```

Using the Scientific preference, with decimal places limited to 4 (see the Decimal Places below for more information on limiting decimal places):

```
123456.78
    = 1.235e5

0.012345
    = 1.235e-2
```

Using the Engineering preference.

```
123456.78
    = 123.45678e3

0.012345
    = 12.345e-3
```

Using the Engineering preference, with decimal places limited to 4, and the “Add Zeros To Show Exactly This Amount” not selected (see the Decimal Places section for more information on limiting decimal places):

```
123456.78
    = 123.4568e3

0.012345
    = 12.345e-2
```

Using the Percentage preference, and Thousands Separators enabled:

```
123456.78
    = 12,345,678%

0.012345
    = 1.2345%
```

Using the Binary preference:

```
123456.78
    = 0b0001 1110 0010 0100 0000 (Floored)

0.012345
    = 0b0 (Floored)

255
    = 0b1111 1111

64
    = 0b0100 0000
```

Using the Octal preference:

```
123456.78
    = 0o361100 (Floored)

0.012345
    = 0o0 (Floored)

255
    = 0o377

64
    = 0o100
```

Using the Hexadecimal preference:

```
123456.78
    = 0x1E240 (Floored)

0.012345
    = 0x0 (Floored)

255
    = 0oFF

64
    = 0x40
```

Significant Figures

You can limit the amount of significant figures that are displayed. If the number has fewer significant figures than the amount you specify, Equations will leave the number alone.

Example

With significant figures being limited to 4 significant figures:

```
123456.78
    = 123500

0.012345
    = 0.01235

0.012
    = 0.012
```

Decimal Places

You can limit the amount of decimals to be displayed. With the “Add Zeros To Show Exactly This Amount” preference selected, Equations will add additional zeros to the number to show exactly that amount of zeros. This preference can be useful when dealing with things like money.

Examples

With decimal places being limited to 4 decimal places, and the “Add Zeros To Show Exactly This Amount” option selected:

```
123456.78
    = 123456.7800

0.012345
    = 0.0123

0.012
    = 0.0120

0.000012
    = 0.0000
```

With decimal places being limited to 4 decimal places, and the “Add Zeros To Show Exactly This Amount” option not selected:

```
123456.78
    = 123456.78

0.012345
    = 0.0123

0.012
    = 0.012

0.000012
    = 0
```

Thousands Separators

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

Example

```
123456.78
    = 123,456.78
```

Variables

Equations has support for variables. You can assign the result of any equation to a variable.

Defining a Variable

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:

```
myVariable=3+5
    myVariable = 8
```

will set the variable `myVariable` to 8. You can then use `myVariable` in any equation as follows:

```
myVariable*2
    = 16
```

All variables you define will show up in the variable cache.

Auto-Cached Variables

If you calculate an equation, and you do not assign the result to a variable, Equations will automatically assign the result to a variable.

All the auto-cached variables are named ans1, ans2, ans3, etc., and will show up in the variable cache. Note that If assign a variable ‘ans3’ to a number before Equations is at ‘ans3’, Equations will skip over it and go to the next available variable name in the auto-cache format.

Note that the 'ans' is always the result of the previous equation, whether or not you assigned the result to the variable. So after doing something like a=3*4, both 'a' and 'ans' will be 12.

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

Overview

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

The User Interface

The user interface design is based on standard Mac OS X controls, and the design process tries to follow the recommendations of the Apple Human Interface Guidelines. Stylistically the goal is to be similar to the iWork applications

A lot of time is spent trying to simplify and increase the speed of calculating things, and communicating useful data and feedback on the calculations.

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, it was made Leopard-only. In 2008, the calculation engine was fine tuned to run on the iPhone/iPod Touch platform, and was heavily modified to take advantage of Objective C 2.0 features, which increases its speed and allows for for greater expandability in the future.

Design Goals

The goal of the calculation engine is not to be computationally fast (although that is a secondary goal). Instead it is intended to be fast and easy to work with from a user standpoint.

Precision

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

The User Guide

The Equations User Guide is written in Pages '08 in Word Processing mode. The body font used is Arial, 10.5 point, 90% black, with 1.1 line spacing. Arial is also used for the various headers, and Courier is used as a code font. An attempt was made to follow the Apple Publications Style Guide.

Release Notes

Following are the release notes for each release:

Version 0.2.2 (January 17, 2009)

- Renamed the app to "Equations", and changed the icon.
- Added support for binary, octal, and hexadecimal numbers:
  - Added ability to input numbers by using the the "0b" prefix for binary numbers, "0o" for octal, and "0x" for hexadecimal.
  - Added Binary, Octal, and Hexadecimal output formatting types.
- Revised the decimal place number formatting, to allow either the exact amount of decimal places to be shown by adding zeros, or just to limit the number of decimal places displayed.
- Added a Percentage output formatting type.
- Fixed a bug in the variable cache where it wouldn't scroll vertically.

Version 0.2.1 (December 22, 2008)

- Divided up the variable cache to separate different types of variables.
- Added the ability to scroll previous formulas by pressing option + up/down arrows.
- Added the ability to insert variables from variable cache by clicking on the variable while holding the option key.
- Added a mini mode, accessible by clicking on the green “zoom” button in the upper left corner of the window.
- Changed the formula entering box so it now automatically inserts the ans variable if you just type an operator after calculating a formula.

Version 0.2 (November 18, 2008)

- Finally updated the program.
- Renamed the application 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.