BUILDIT QUICK HELP

BUILDIT MENU

Start Simulation

This command is used when model simulations require a loop for repetitious calculations. Consequently, this command requires cell names `_step`, _stepsize, and `_criteria` to be defined first in the current workbook.

You can stop a simulation run at any time by pressing the Esc key or the Control-C keys.

Do Operations Only

Executes all actions listed only in the operation section, without starting the loop, if any.

Reset Model

Executes all actions listed only in the prerun section, without starting the loop, if any. This command is often used when you have `INI` actions to initialize variables. Using this command will then give these variables their initial values.

Trace Workbook

Produces a cells network map for one or more worksheets in the current workbook. This map depicts visually all cells interrelationships. All levels of cells precedents and dependents are traced.

Trace Selected Cells

Produces a cells network map for only the currently selected cells. Only first-level cell precedents are traced.

Fix #NAME? Error

This command fixes the Excel's error message that one or more external links in a workbook cannot be updated. The exact error message, which depends on your Excel version, would resemble the following:

"This workbook contains one or more links that cannot be updated"

This error happens when a workbook uses the BuildIt's functions (`interpolate` or `solve`), but BuildIt was not installed in the same location in all your computers. Consequently, Excel is unable to locate BuildIt in the computer you are currently using to open your workbook.

Because of this update link error, cells that use the interpolate or solve functions will return the #NAME? error value.

To fix this error, just choose the “Fix #NAME? Error” command from BuildIt’s menu each time you open the workbook in different computers.

Note this command works to fix only BuildIt’s links and not the links to other sources, add-ins, or programs.

Help

Displays this help file.

About BuildIt

Displays information about the BuildIt add-in.

CELL NAMES

BuildIt requires some cell names to be defined. These cell names indicate to BuildIt where to find certain required information such as where the information about the loop or actions to be executed are located in the current workbook. Table 1 shows the list of cell names.

Table 1. BuildIt cell names (which always begin with an underscore).

Cell name	Brief description
_step	Iteration step (loop counter)
_stepsize	Interval or increment size
_criteria	Condition when to end the loop: TRUE to continue or FALSE to end the loop
_operation	Marks the start of the Operation section
_prerun	Marks the start of the Pre-run section
_option	Marks the start of the Option section
_scenario	Marks the start of the Scenario section
_read	Marks the start of the variables list to be included in the model output
_write	Marks the start of the output listing

ACTIONS

BuildIt supplies 12 actions to perform specific tasks (Table 2).

Table 2. List of BuildIt actions.

Code	Action name	Brief description
ITG	Integration	Numerical integration using the five-point Gaussian method.
DIF	Differentiation	Numerical differentiation to determine either the first or second derivative using the three-point central (midpoint) difference method.
ROT	Rotate	Rotates the order in an array of values to the left, right, up, or down direction. For example, given an array of (1, 2, 3, 4, 5), rotating it to the right or down would give (5, 1, 2, 3, 4), but rotating it to the left or up would give (2, 3, 4, 5, 1).
REV	Reverse	Reverses the order in an array of values. For example, given an array of (1, 2, 3, 4, 5), reversing it would give (5, 4, 3, 2, 1).
SHU	Shuffle	Randomly shuffles the order in an array of values. For example, given an array of (1, 2, 3, 4, 5), a shuffle might give (3, 4, 1, 2, 5); the sequence randomly determined.
SOR	Sort	Sorts the order in an array of values either ascendingly or descending.
ARR	Arrange	Arranges the order in an array of values according to their given sequence. For example, given an array of (10, 20, 30, 40, 50) and the corresponding sequence as (3, 1, 4, 5, 2), this action would give the new array of values as (20, 50, 10, 30, 40).
ACC	Cumulative	Determines the cumulative sum or product of a given variable.
REP	Copy value	Copies the values in source cells to destination cells.
UPD	Update value	Updates the values of one or more variables by their rates of change using the Euler method.
INI	Initialize value	Initializes one or more variables with their given initial values.
RUN	Runs a macro or script	Runs a user-defined VBA script or macro. BuildIt supplies three utility macros: 1) ClearOutput, 2) EnableScreenUpdate, and 3) DisableScreenUpdate.

All BuildIt actions have the following syntax:

ACTION $\{param_1:s|r\}\ \{param_2:s|r\}\ \ldots\ \{param_n:s|r\}\ \{op:s=TRUE\}$

where *ACTION* denotes a three-letter code for the type of action requested (Table 2), and $param_{1,2,\ldots,n}$ are the action's first, second, and so on until the *n*-th parameters, where the number of parameters depends on the type of action.

Each parameter's ":s|r" notation means a parameter can either be of type ":s" or ":r". The ":s" type means the parameter must be supplied either as a constant (such as numbers, logical values TRUE and FALSE, or text) or a single cell address (such as A1 and B5). The ":r" type is the same as ":s" except it can additionally take a contiguous cell range (such as A1:B3 and C4:C6).

Note that the last parameter *op* is unique because this parameter is always of type ":s" and it only accepts logical values TRUE or FALSE. The last parameter indicates whether or not the given action should be executed. If *op* is TRUE, the action is executed, but a FALSE value means the action will be not be carried out. This last parameter gives the flexibility of executing a given action only in certain conditions. If the last parameter is unspecified, it is TRUE by default, which explains the notation "op:s=TRUE".

ITG operation

ITG {x:s} {fn:s} {output:s} {lower:s} {upper:s} {single_integral:s=TRUE} {op:s=TRUE}

Integrates the function *fn* of *x* over the interval [*lower*, *upper*], and the result of the integration is to be stored in *output* cell. The second last parameter, *single_integral*, is either TRUE for single integrations or FALSE for multiple integrations. If omitted, *single_integral* is assumed TRUE by default.

DIF action

DIF {x:s} {fn:s} {output*:s} {n:s=1} {interval_size:s=0.001} {op:s=TRUE}*

Gives the first or second derivative of $d^n fn/dx^n$, where *n* is either 1 (first derivative) or 2 (second derivative). By default, *n* is 1. The *interval_size* for numerical differentiation is 0.001 by default, but this value may require some experimentation as too small or large a value may give large errors. The *output* parameter is the location where the result of differentiation should be stored.

ROT operation

ROT {array_of_*values:r} {direction:s=RIGHT} {op:s=TRUE}*

Rotates the order in *array_of_values* in the direction specified by *direction* (LEFT, RIGHT, UP or DOWN). For example, given an array of (1, 2, 3, 4, 5), rotating it RIGHT or DOWN would give (5, 1, 2, 3, 4). Instead, rotating it LEFT or UP would give (2, 3, 4, 5, 1). Directions LEFT and RIGHT are given for arrays specified in a single cell row, such as cell range A1:F1. Directions UP and DOWN are for arrays specified in a single cell column, such as in cell range A1:A6.

REV operation

REV {array_of_*values:r} {op:s=TRUE}*

Reverses the order in *array_of_values*. For example, given an array of (1, 2, 3, 4, 5), reversing it would give (5, 4, 3, 2, 1). Reversing it again produces the original sequence.

SHU operation

SHU {array_of_*values:r} {op:s=TRUE}*

Shuffles the order in *array_of_values* at random. For example, given an array of (1, 2, 3, 4, 5), a shuffle might give (3, 4, 1, 2, 5), the sequence randomly determined.

SOR operation

SOR {array_of_*values:r} {ascending_order:s=TRUE} {op:s=TRUE}*

Sorts the order in *array_of_values* either in an ascending or descending order. Specify *ascending_order* as TRUE to sort the values ascendingly, else set to FALSE for a descending sorting order.

ARR operation

ARR {array_of_values:r} {array_of_*sequence:r} {op:s=TRUE}*

Arranges *array_of_values* in the order specified by *array_of_sequence*. For example, given *array_of_values* as (10, 20, 30, 40, 50) and the corresponding order in *array_of_sequence* as (3, 1, 4, 5, 2), arranging the array would give the new *array_of_values* as (20, 50, 10, 30, 40), where 10 corresponds to sequence 3, 20 to 1, 30 to 4, 40 to 5, and 50 to 2.

ACC operation

ACC {x:s} {cumulative_x:s} {operator_*type:s} {op:s=TRUE}*

Obtains the cumulative sum of *x*, if operator_*type* is + or the cumulative product of *x* if *operator_type* is *. The cumulative sum or product is stored in *cumulative_x*.

REP operation

REP {destination:r} {source:r} {num_of_replacements:s=1} {op:s=TRUE}

Copies the cell values from *source* to destination as many as *num_of_replacements* times (if omitted, the default is once).

UPD action

*UPD {x:r} {rate_of_change:r} {n:s=1} {*op*:s=TRUE}*

Using Euler's method, this action estimates $x_{t+\Delta t}$ based on its current value x_t and rate of change dx_t/dt and time interval Δt as

$$x_{t+\Delta t} = x_t + \frac{dx_t}{dt}\Delta t$$

where action parameter *x* is x_t; parameter *rate_of_change* is dx_t/dt; and Δt is the interval size (`_stepsize`). Parameter *n* (defaults to 1 if *n* is unspecified) is the number of subintervals to have within each Δt period.

INI operation

INI {destination:r} {source:r} {op:s=TRUE}

Copies the cell values from *source* to *destination*. This action is used primarily for setting initial values to model parameters.

RUN operation

RUN {script_name:s} {op:s=TRUE}

Executes the script or macro named script_*name*. Users can create their own scripts (using VBA) and use RUN to execute their scripts. BuildIt, however, supplies its own three utility scripts: 1) ClearOutput, 2) DisableScreenUpdate, and 3) EnableScreenUpdate script. The ClearOutput script clears the model output from previous simulation runs. This script is useful to reduce output clutter. The DisableScreenUpdate script turns off the screen update feature in Excel. Disabling the screen update feature can increase execution speeds in particular for models with intensive calculations or with many iterative calculations. Run EnableScreenUpdate script to re-enable the screen update feature. Scripts ClearOutput and DisableScreenUpdate are often run from the Option section so that they are used before the simulation runs begin. Note that the screen update feature is always turned back on automatically by BuildIt at the end of every simulation run.

FUNCTIONS

BuildIt provides two functions: 1) `interpolate`, and 2) `solve` (Table 3).

Table 3. BuildIt functions

Function	Brief description
`interpolate`	Given an array of (*x*, *y*) values, the interpolate function performs a linear interpolation, if needed, to return the corresponding *y* to the given *x* value.
`solve`	An array function to solve linear and non-linear simultaneous equations. This function uses the Newton iteration method.

Interpolate function

interpolate (array_of_x:r, array_of_y:r, find_x:s)

Given an array of (*x*, *y*) values, the `interpolate` function performs a linear interpolation to return *y* corresponding to *find_x*. It is important to remember that *array_x* must be sorted either ascendingly or descendingly before this function can be used reliably.

Solve function

*solve (*variables*:r, constants:r, initial_values:r)*

The `solve` function is an array function which returns its results in two or more cells. The `solve` function solves linear and non-linear simultaneous equations. Before this function can it be used, the equations to be solved must be arranged so that the unknown variables are on the left hand side of the equations and constants on the right hand side. In particular for solving non-linear equations, good initial estimates for the unknown variables may be required for accurate answers. For example, the following are two non-linear equations:

$$3\sin(x) + y^2 = 3$$
$$x^2 + 2y = 1$$

where there are two unknown variables: *x* and *y*. Fig. 1. shows one way how the `solve` function could be used to determine *x* and *y*.

	A	B	C	D	E
1		**initial values**	**variables**	**constants**	**answer**
2	x	1	=3*SIN(B2)+B3^2	3	=solve(C2:C3,D2:D3,B2:B3)
3	y	1	=B2^2+2*B3	1	=solve(C2:C3,D2:D3,B2:B3)

Fig. 1. Using the `solve` function in Excel to solve $3\sin(x) + y^2 = 3$ and $x^2 + 2y = 1$ for *x* and *y*.

Both equations are implemented in Excel by arranging the unknowns to be in cell column C and the constants in column D. Cells B2 and B3 contain the initial estimates for *x* and *y*, respectively. Select range E2:E3, then type "=solve(C2:C3,D2:D3,B2:B3)", and finally press the keys Control+Shift+Enter (rather than just the Enter key because `solve` is an array function) to obtain the answer. The `solve` function would return 1.294 and -0.338 for *x* and *y*, respectively.