Mesa 2.0 User Guide

Index

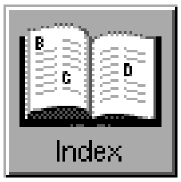
Getting Started


This is Mesa's icon.

For a quick introduction, click here 7

Press the F1 key and click objects on the screen for quick descriptions.

For details, click here



Click the Index button and click the topic you want. Or click a Help topic above. *For details, click here*

Mesa Introduction

Mesa is a traditional spreadsheet program that runs on NEXTSTEP computers. Mesa looks and works similarly to Lotus 1-2-3 and Microsoft Excel. If you have used either of these programs, then Mesa should be very easy to learn.

Please note that while Mesa works very much like 1-2-3 and Excel, Mesa does do things in slightly different ways or menu options will be located in different places. For example, Mesa uses a NEXTSTEP convention called *Inspectors* to achieve many results. Just because Mesa does not work the exact same way as 1-2-3 or Excel does not mean that Mesa cannot produce the same result. If you have questions, please contact us or look in the Help.

Mesa contains many features to make building worksheets and presenting data intuitive and efficient. Mesa has many other features to help you work more naturally. Please take some time to read throug the Help so that you can discover more about Mesa.

If you have questions, please contact P&L Systems technical support team either via e-mail (feedback@plsys.co.uk) or by phone at

(44) 1494 432422.

See also

More about Mesa *16*
Mesa Basics *30*

What’s new in Mesa 2.0?

Users of Mesa 1.5B should read through this to get an idea of the new features available in Mesa 2.

Summary

New User Interface
Mesa 2.0 has been enhanced to make the User Interface compliant with NeXTSTEP/Openstep standards and therefore more intuitive and easier to use.   All icons have been completely redesigned.

New features
Mesa 2.0 includes:

- New Toolbox
- Graphics in Reports
- QuickHelp on all icons
- SplitScreen formula bar
- Automatic Range updating in Reports, Queries, Extracts and Labels
- New Inspector panel
- Improved Formula Builder
- Oracle and Quickbase Database AddIns incorporated within Mesa
- HTML Table Export AddIn
- dBASE file import and export AddIn
- Informant! Data Feed AddIn

as well as a host of smaller yet still useful items requested by 1.5B users: eg

- Formula entry line now supports all standard NS selection keystrokes
- *Find* now available from the Edit menu
- *Font Larger/Smaller* now available from Font menu
- *Esc* can now Abandon an entry
- *Home* key now takes you to A1
- *PgUp* and *PgDn* keys work
- *Dates* will now copy out of Mesa as strings, not numbers (settable in Preferences)

Compatible with Mesa 1.5B
Mesa 2.0 is fully backwards compatible with Mesa 1.5B. All your Mesa 1.5B worksheets (including reports, graphs, queries etc) will load and work identically.

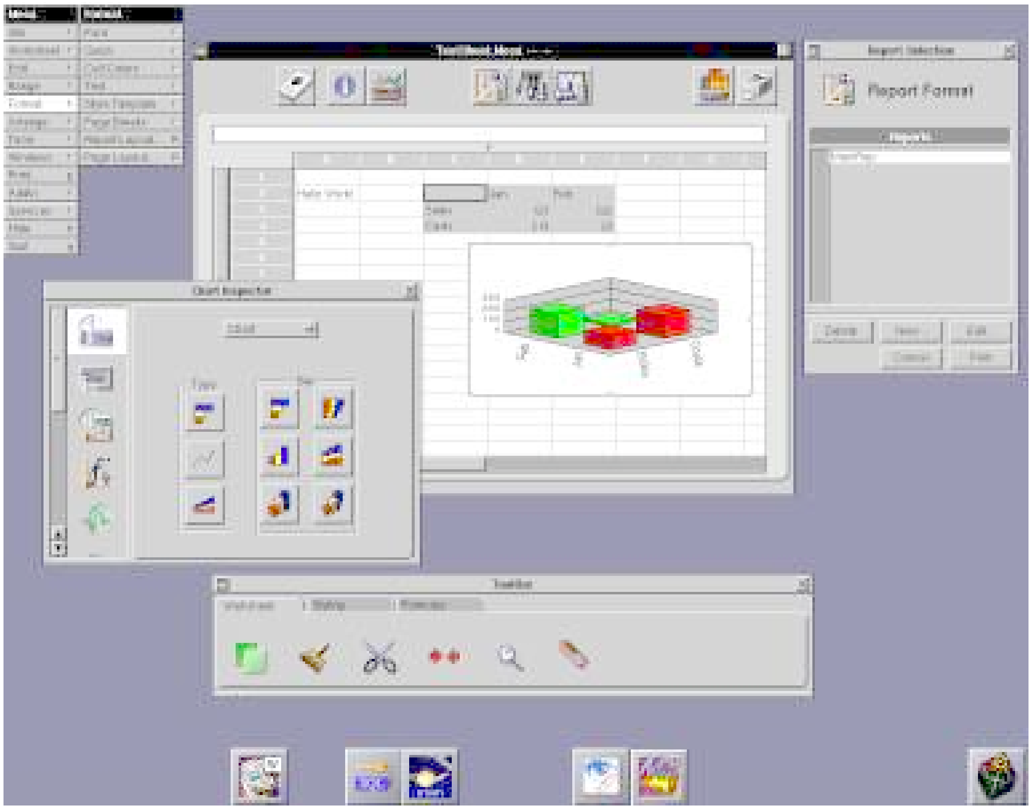
All the features of Mesa 1.5B are still there in Mesa 2.0. However, if you are used to 1.5B you should invest a little time in exploring Mesa 2.0 to locate the menu options that you need.

Revised User Interface

The worksheet, menus and Inspectors have been completely overhauled to make them NeXTSTEP/Openstep compliant.

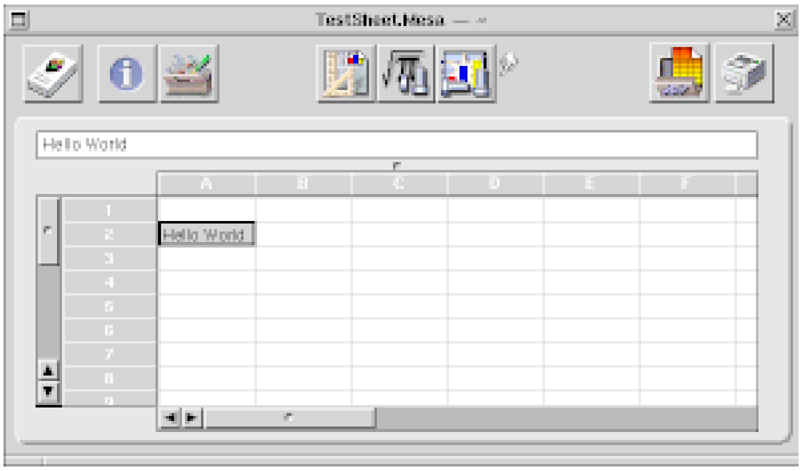
- New Inspector panels
- All icons have been completely redesigned to present a crisper and more pleasing effect
- Next-Style help has been added

As you read through the new features below, you will see screenshots of the new-look Mesa. Here's a full screen shot to give you an idea.



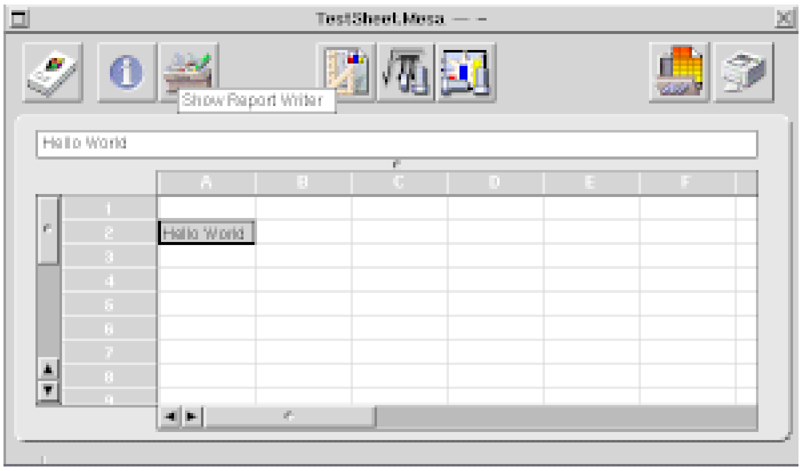
New-look Worksheet

The Worksheet window now includes an Icon Bar containing short-cut routes to the most frequently used panels. As in Mesa 1.5B, the Save and Print icons can be replaced with any other icon you want (from the Menus, Report Layouts, Queries or Extracts panels).



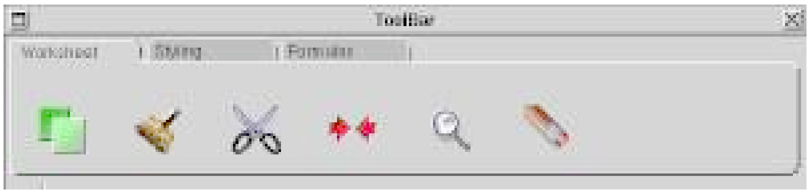
QuickHelp

Although we think that the completely new icon-set Mesa 2.0 uses is far crisper and cleaner than 1.5B, we recognise that no-one can work out the meaning of an icon just by looking at it. So we've provided a QuickHelp feature.   Position the mouse over any icon, and after 2 seconds a short description of that icon's function will pop up. Move the mouse and the QuickHelp description disappears.



The ToolBox

The new ToolBox lets you keep short-cut icons for anything you'd find useful. Bring up the Toolbox with Tools->Toolbox, or by selecting the ToolBox icon on the worksheet.



The Toolbox is fully customisable so that you can set up the shortcuts that *you* need. You can also resize it and position it anywhere you like on the screen.

QuickHelp works on all icons in your ToolBox.

Setting up the Toolbox

Adding an icon

Use Command-Drag to drag an icon off a menu. Drag it to the position on the toolbar you want and then release the mouse button.

Deleting a icon

Drag it outside the ToolBox onto the workspace and let go with the mouse.

Moving an icon

Drag it to the new location. Note that you can't drag icons between Tabs.

Adding a Tab

Click on a Tab name twice (so that the tab area turns dark grey), and press Return. A new tab with a name like 'New2' will be created.

Renaming a Tab

Doubleclick on the Tab Name, and type in the name you want

Deleting a Tab

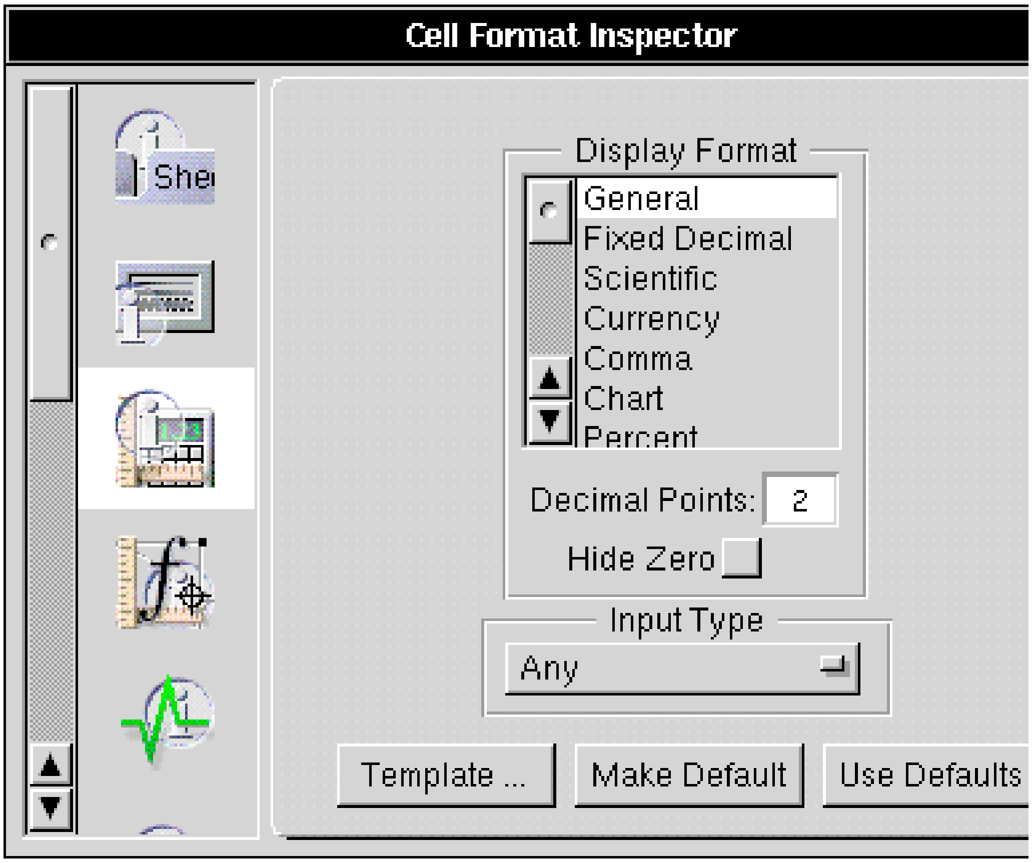
Click on a Tab name twice (so that the tab area turns dark grey), and press Backspace. The Tab and all the icons it contains will be removed from your toolbar.

The Inspectors

There are Mesa Inspectors for the following:

Inspector	*You create and change..*
Sheet	Dimensions, general behaviour
Borders	Lines around cells
Cell Formats	Display Formats
Cell Styles	Font, Colour etc
Events	Addins
SQL Queries	Queries into External databases
Extracts	Queries within the Mesa worksheet
Scenarios	Scenario tables
Layers	Positioning of Graphic objects on the worksheet
Labels	Names for areas on the worksheets
Charts	Change charts created via the Chart Selector Tool

All Mesa Inspectors are grouped together in the Inspector Panel. Bring this up with *Tools->Inspector* or by selecting the Inspector icon on the Worksheet.



Select the inspector you want by selecting the appropriate Icon from the scroll area at the left.

QuickHelp works on all icons in your Inspector panel.

The Inspector options for Charts will not be available unless you have either just created a new chart or have selected an existing chart on the worksheet.

Reports

Mesa 2.0 uses a ReportWriter Tool similar to the one in Mesa 1.5B to let you set up and save Reports. A Report is a a set of items to which you give a Name, holding

- the range(s) to print
- paper size/orientation
- Header and Footer options (page numbering, dates, graphics etc)
- general display options

You can create as many reports as you like, giving a different name to each one.

Putting a Graphic item on a report for Printing

New in Mesa 2.0 is the ability to place a tiff, ps or eps file on your printout.

To do this, drag the Graphic Tag onto the report layout. Then drag the icon for the graphic object directly over the Graphic tag. You can resize and move the graphic to fit your needs.

SplitScreen Formula Entry

You can make the Formula Entry area larger by dragging the 'dimple' at the top of the worksheet.

If you drag the Formula Entry area to zero size, then you can edit the cell contents directly in the cell.

Automatic Range Updating

Ranges in Reports, Extracts, and Labels will now fill automatically with the cell co-ordinates of the area currently selected on the worksheet.

To edit a defined range, select the range co-ordinates in the Report, Extract etc, and then click back in to the worksheet and reselect the area you require. This will automatically update the range.

Charts

Mesa 2.0 has a new Chart Selection Panel and a redesigned Chart Inspector.

One major improvement is that once you have created a new Chart, the chart selection panel disappears from the screen and is replaced by the Chart Inspector, from which you can select different chart types. This removes the problem in 1.5B whereby users created several charts on top of one another on the worksheet!

Key usage

So many users asked us to change the behavoius of ESC to be like Excel and 1-2-3 (ie, abandoning the entry) that we've done it! If you prefer ESC to bahave like it did in 1.5B, use *Preferences Keys* and uncheck the 'Escape Abandons' box.

If you use an Intel, Sparc or HP system, you can now use the Home key (to go to A1), and the PgUp and PgDn keys to move up/down the worksheet.

Dates as Strings

On the Preferences File menu you can now choose to have dates copy out of Mesa as strings rather than as numbers. This applies to Copy/cut and Paste operations, and to the Export Delinited Text option.

Mesa Features

More about Mesa

A spreadsheet is a program that allows the user to enter data into matrix of cells. Each cell can be individually addressed based on its row and column number. In Mesa, rows are numbered from 1 to 16384 and columns are labeled from A to XFD. Each cell has a value and formatting information. The cell's value is based on either its contents (e.g., a number like 5, or a string like *Hello*) or on a formula that is evaluated to result in the cell's value. A value can be a number (e.g., 1, 4.7, 5280), a string (e.g., *Hello*, *June Stock Prices*, *Eastern Region*), or an error (e.g., !ERR, !/0).

A formula can refer to the value in a cell during calculation. This is why spreadsheets are so powerful. Because the value in one cell can depend on the value in another cell, when the first cell is changed, the second cell updates itself properly. For example, if a cell contained the formula *=A1+1*, when the value in cell A1 is changed, the cell containing the formula automatically updates itself.

Cells also contain formatting information. This includes the text format of the display (e.g., General number, Scientific notation, Currency format), the color of the text, the background color of the cell, and more. Information can be presented in a more visually pleasing or informative manner by changing the cell formatting. Mesa allows some of the cell's formatting to be changed by the cell's formula so that a cell can automatically change its color if it contains certain values. This feature allows worksheets to alert the user to important cells that contain critical values.

Once the cells have been populated with values, and the spreadsheet has been formatted, it can be printed out, e-mailed to others, or saved. In addition, Mesa allows you to graph the values in cells to produce presentation-quality graphs and include those graphs in reports or copy the graphs or groups of cells and paste them into other NEXTSTEP applications including Create and DataPhile from Stone Design, Concurrence from Lighthouse Design, or any NEXTSTEP application that utilizes live links.

Mesa also allows data to be exchanged with SQL database systems. Mesa can retrieve data from an SQL system. Once the data is in Mesa, it can be formatted, you can enter formulas that calculate based on the data retrieved, and you can produce presentation quality reports. When you use Mesa this way, it becomes a report writing tool for SQL database systems.

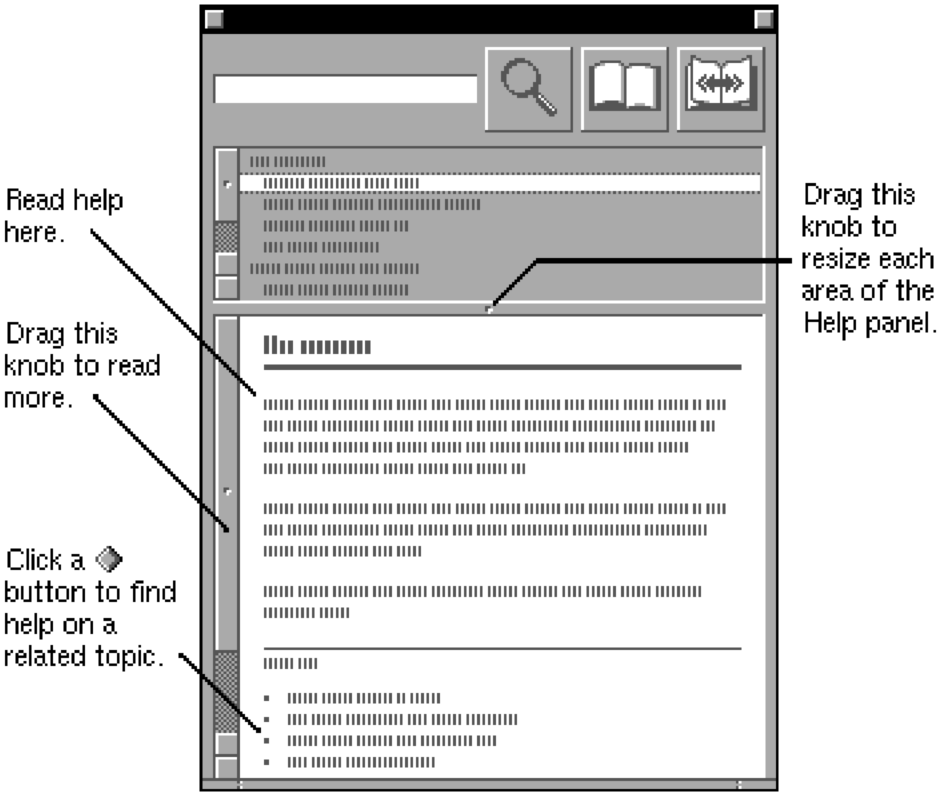
Mesa supports real time data feeds. You can purchase modules that add in to Mesa (AddIn's) that provide real time stock market data feeds, and other live information. This allows you to build a model (such as a stock portfolio), and then have the current real world information flow into the worksheet thus keeping the portfolio values up to date.

Mesa also includes MOLI (the Mesa Object Library Interface) which allows another application to link right into the data in a worksheet, execute database queries, graph information, and much more. Many NEXTSTEP sites link their mission critical custom applications into Mesa so that Mesa can graph data for them, create a unified look to all their applications, reduce the amount of programming time to build the custom applications, and to access many of Mesa's features.

Once the data has been entered into Mesa from various different sources, the formulas have been entered so that the raw data can be summarized, the formatting has been applied, and the charts and graphs have been created, you can print the worksheet in order to produce a presentation quality report. Mesa takes advantage of Display PostScript which is built into every NEXTSTEP system to produce crisp, presentation quality output on whatever printing device you have connected to your NEXTSTEP system.

Click for help

▸ Hold down the F1 key on the keyboard and click any command, window, or panel.



✓ When you hold down the F1 key, the cursor changes to a question mark ?.

✓ If your keyboard doesn't have an F1 key, you can try getting the question mark cursor by holding down the Alternate and Control keys at the same time.

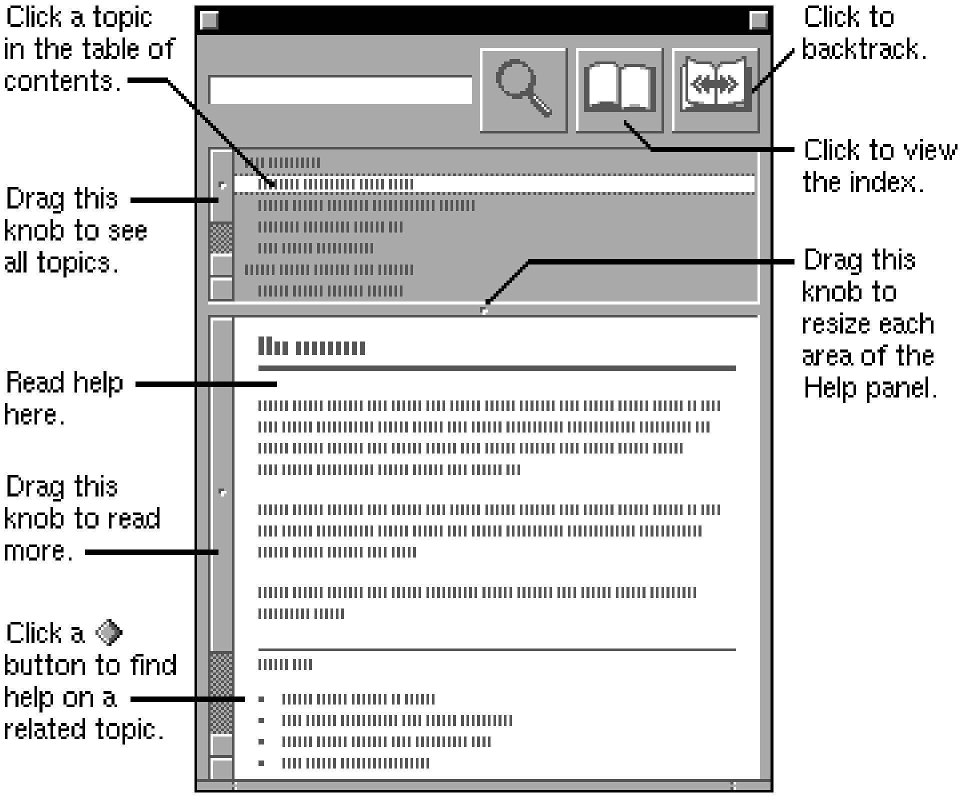
✓ When you click something with the question mark cursor, the Help panel opens. The help you requested appears in the panel (where you're reading right now).

See also

Get help by topic *19*
Find help by keyword *20*

Get help by topic

1 Choose Info from the main menu. **2** Choose Help from the Info menu. **3** In the Help panel, click the topic you want.



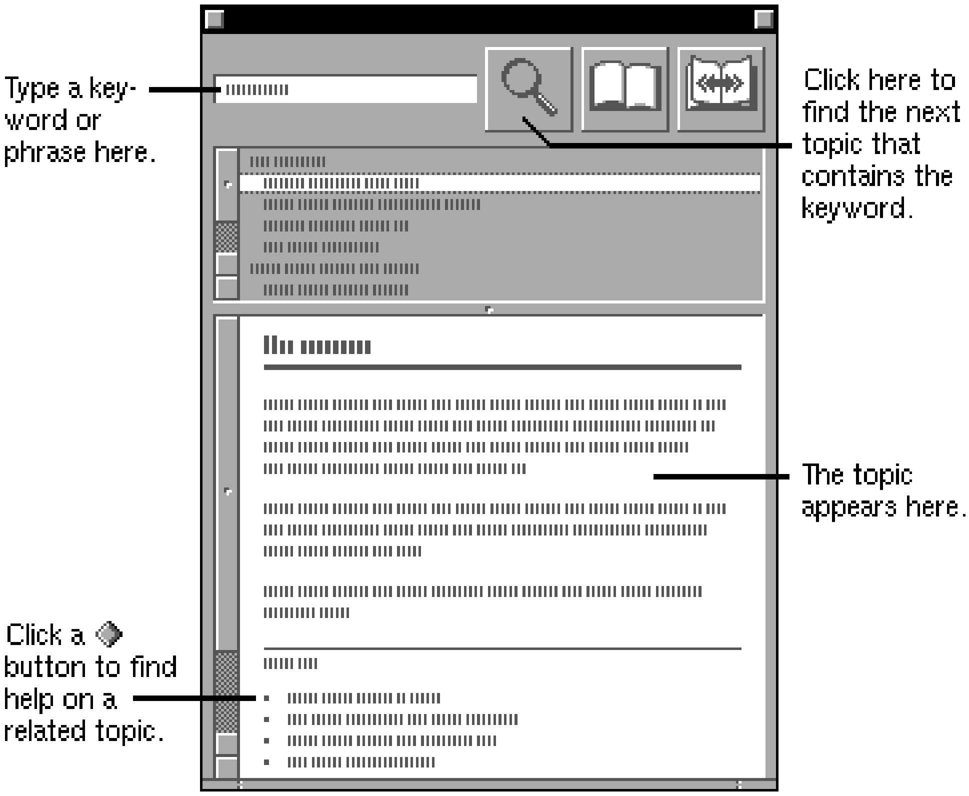
✔ To view a topic listed in the index, click the Index button.

✔ To go back to the topic you viewed last, click the Backtrack button.

See also

Find help by keyword

1 Choose Info from the main menu. **2** Choose Help from the Info menu. **3** In the field at the top of the Help panel, type a keyword. **4** Click Find.



✔ Keyword searches are particularly useful when you can't find the topic you want in the table of contents or the index.

✔ Help is organized like a stack of topics. The Find button searches through the stack from the current topic.

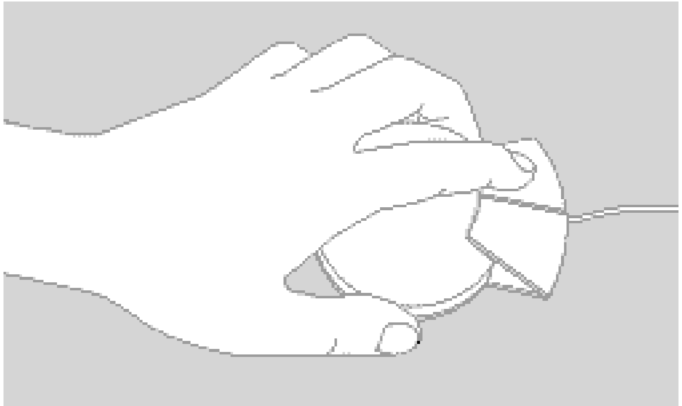
✔ You can click Find again and again to find the next topic that contains your keyword.

See also

Get help by topic *19*
Click for help *18*

Use the mouse

1 Move the cursor by sliding the mouse over a smooth, flat surface. **2** Position the cursor on an object such as a command, button, slider, or icon. **3** Use a mouse button to click, drag, or double-click.



You can normally use either the left or right mouse button—initially they each have the same effect.

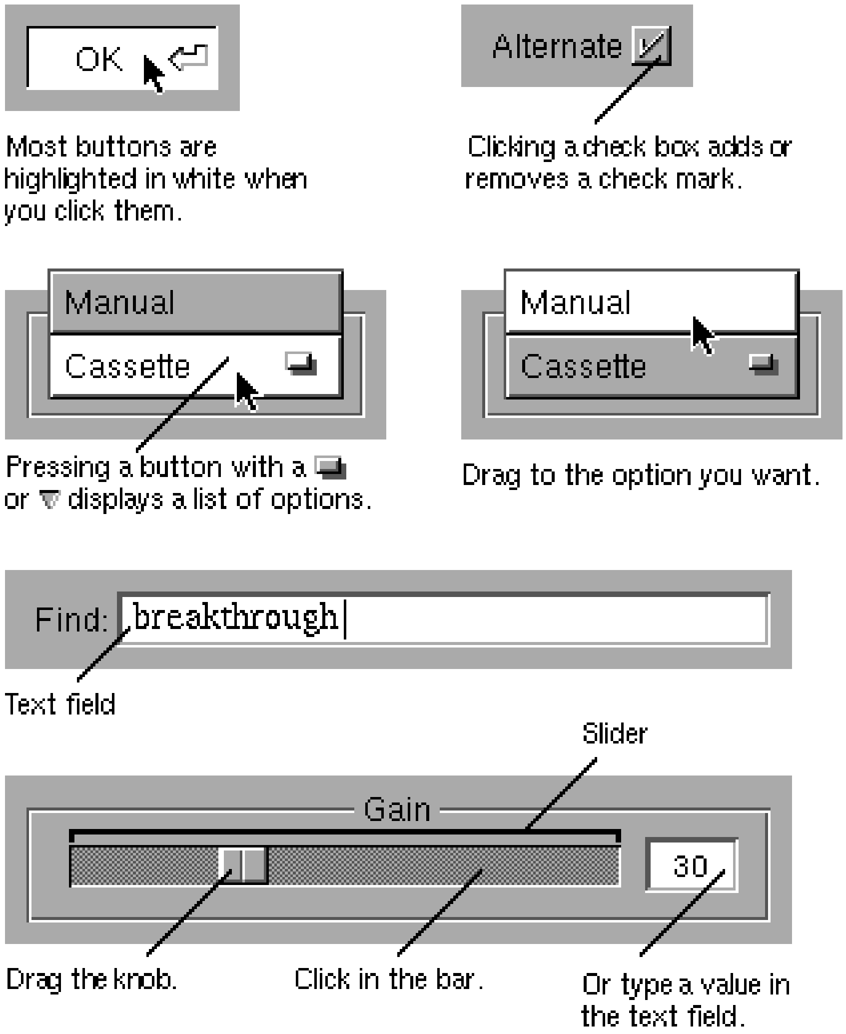
- ✔ Slide the mouse on a smooth surface to point the cursor at something on the screen. Then use a mouse button to perform an action on what you're pointing at.
- ✔ **To click** Without moving the mouse, press and release a mouse button.
- ✔ **To press** Without moving the mouse, press a mouse button and hold it down. Pressing usually has the same effect as repeated clicking.
- ✔ **To drag** Press and hold down a mouse button. Then move the cursor by sliding the mouse. Finally, release the mouse button.
- ✔ **To double-click** Click twice in quick succession. This often extends the action caused by a single click.
- ✔ If you run out of room while sliding the mouse, you can lift the mouse and put it back down somewhere else without moving the cursor. No matter how far you move the mouse, the cursor stays on the screen.

See also

The cursor
Use buttons, sliders, and text fields 22

Use buttons, sliders, and text fields

▶ To operate a button, click it or press it. ▶ To type in a text field, click in the field and start typing. ▶ To adjust a slider value, drag its knob or click in its bar.



✔ Buttons are labeled with text or graphics to show what they do. A dimmed button, shown in gray rather than black, is currently unavailable.

✔ A in a button means you can operate the button by pressing the Return key.

✔ You can move to the next text field in a window by pressing the Tab key. Hold down the Shift key and press Tab to move to the previous next field.

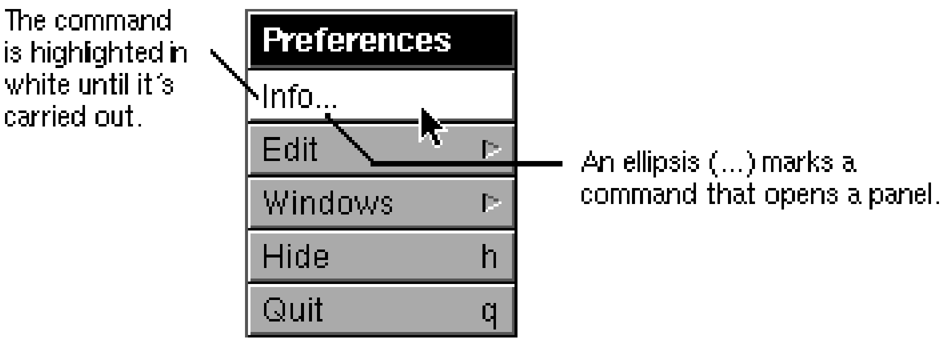
✔ If a text field next to a slider shows a value for the current setting, you can type in the field and press Return to change the setting.

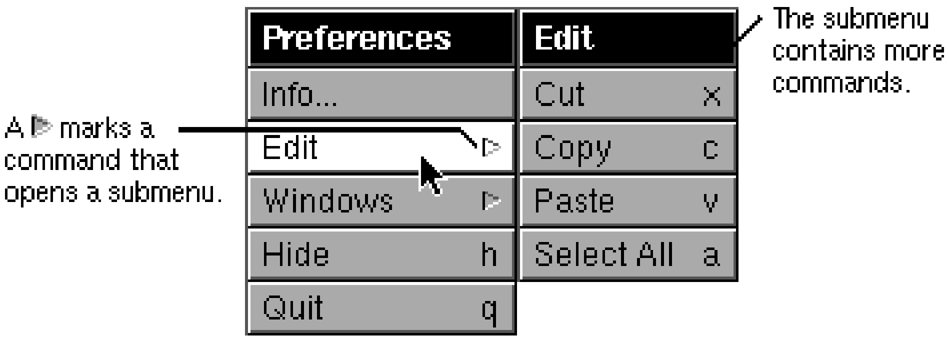
See also

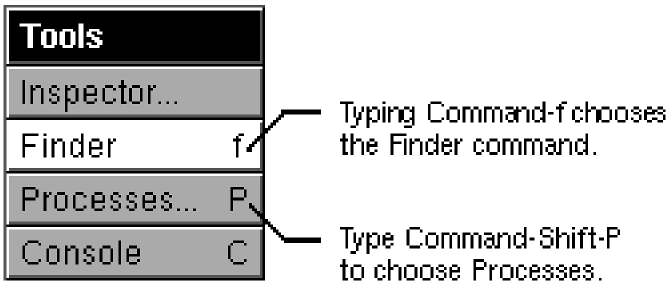
Choose commands	*24*
Use the mouse	*21*

Choose commands

▸ Click the command. *Or* ▸ Drag through a menu to the command. *Or* ▸ Hold down the Command key and type the character shown next to the command.







✓ Some commands open panels or submenus. You can use these to make additional choices.

✓ A submenu closes when you choose another command from the menu that opened it.

✓ When you drag through a command that opens a submenu, the submenu appears momentarily. You can drag through a menu to quickly scan the contents of its submenus.

✓ The Command key is the key to the left of the space bar that, on most keyboards, is labeled "Alt."

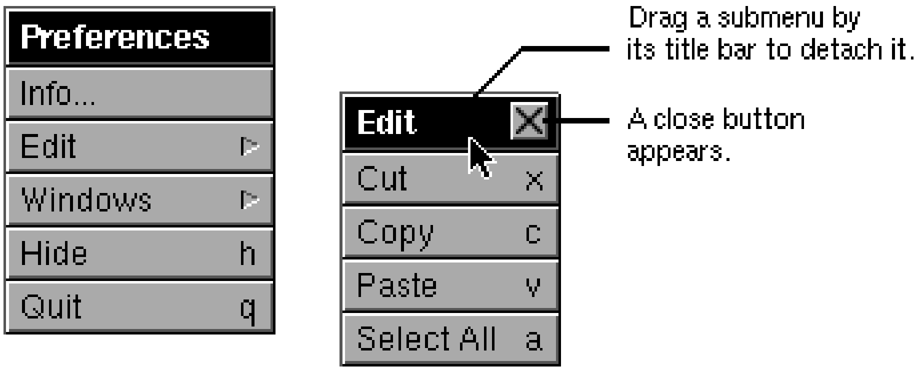
- ✓ A dimmed command, shown in gray rather than in black, is currently unavailable.
- ✓ For more details, see the *User's Guide*.

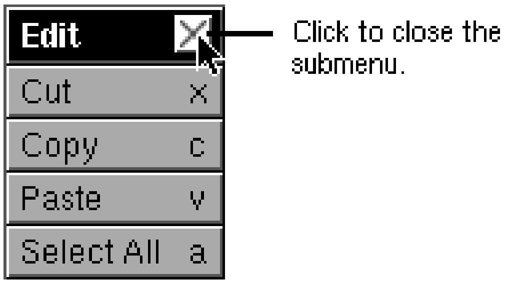
See also

Detach submenus *26*

Detach submenus

▸ To detach a submenu from the menu that opens it, drag the submenu by its title bar. ▸ To close a detached submenu, click its close button.





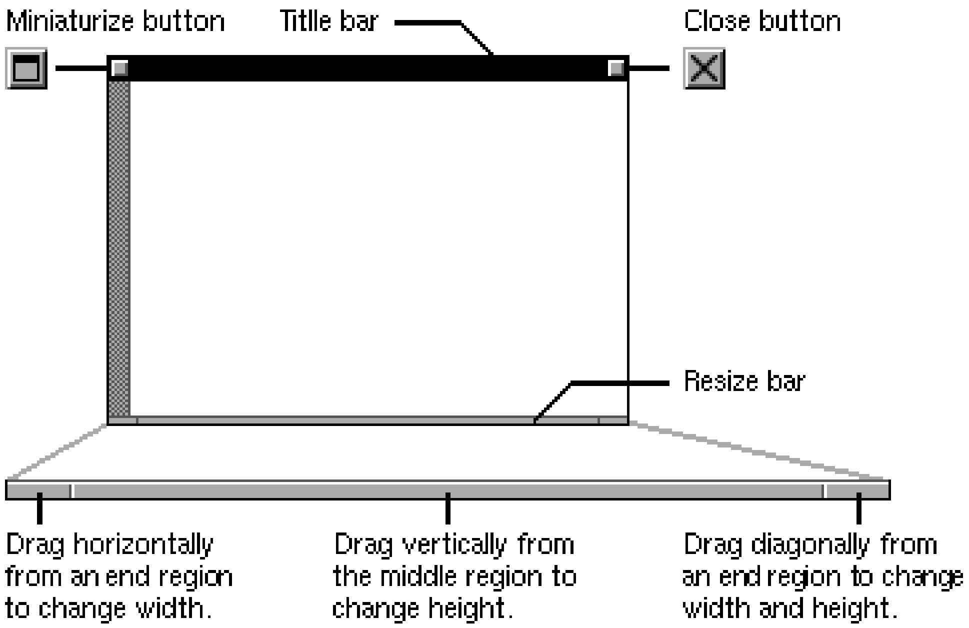
✔ A submenu normally closes when you choose another command from the menu that opens it. You can detach the submenu to keep it open as you work.

See also

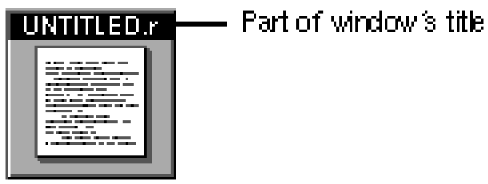
Choose commands *24*

Work with windows

▶ To select a window to work in, click in it. ▶ To move it, drag it by its title bar. ▶ To put it aside temporarily, click its miniaturize button. ▶ To close it completely, click its close button. ▶ To make it larger or smaller, drag its resize bar.



When you miniaturize a window, it appears as a miniwindow at the bottom of the screen.



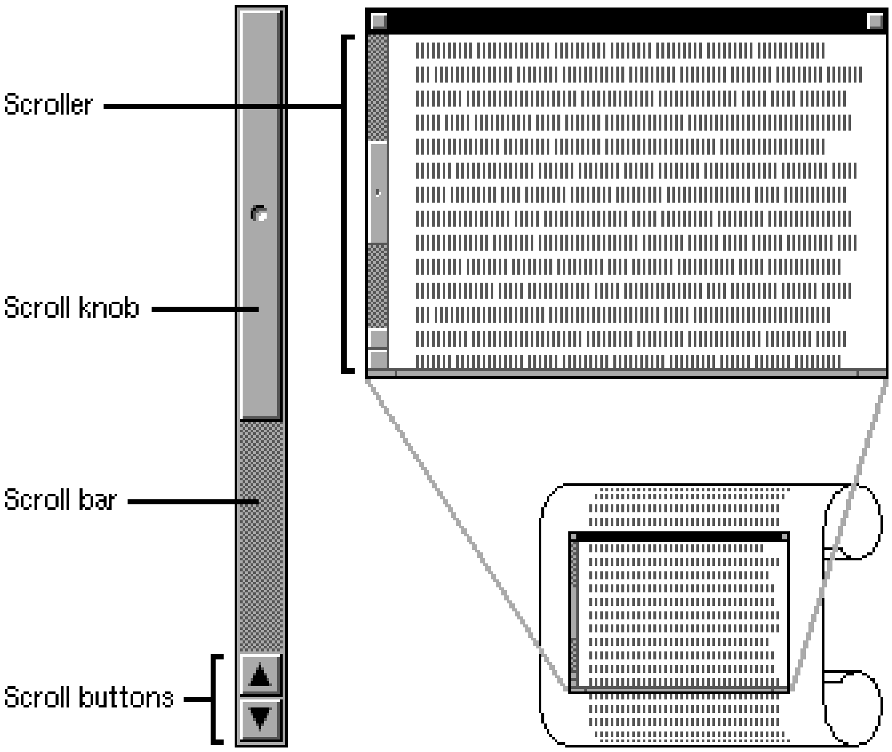
- ✔ The *key window* is the one where you're currently working. Its title bar is highlighted in black.
- ✔ Clicking in a window brings it forward, in front of other windows, and makes it the key window.
- ✔ Double-click a miniwindow to display the window full size.
- ✔ For more information on windows in general, see the *User's Guide*.

See also

Scroll to see more 28
Windows menu

Scroll to see more

▶ To scroll one line, click a scroll button. ▶ To scroll continuously, press a scroll button or drag the scroll knob. ▶ To scroll quickly to an approximate location, click in the scroll bar. ▶ To scroll a windowful, hold down the Alternate key and click a scroll button.



✔ When a window contains more than can be displayed at once, you scroll to see what isn't currently visible.

✔ The scroll knob's size and position indicate how much of the total contents you're viewing and where you are.

✔ The Alternate key is the key to the right of the space bar that, on most keyboards, is labeled "Alt."

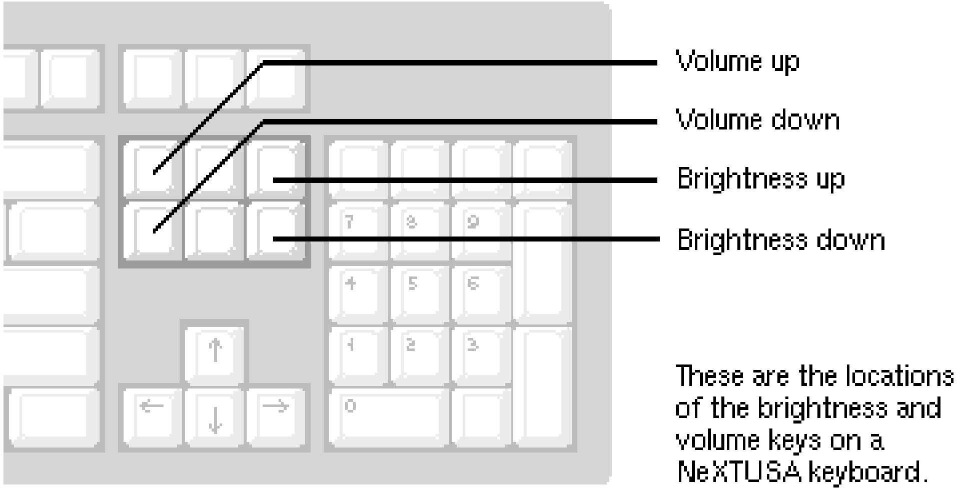
✔ For information on other ways to scroll, see the *User's Guide*.

See also

Work with windows *27*

Adjust brightness and volume

▸ Use the Preferences application. *Or* ▸ Use the standard brightness and volume keys for your computer's keyboard.



✔ You can adjust brightness and, if your computer can play sounds, its volume by using the Preferences application. See the *User's Guide* or start up Preferences and choose Help from the Info menu. You can also use the standard keys for your computer's keyboard as described in its owner's guide.

✔ If you're using a NeXTUSA keyboard arrangement, you can adjust brightness and volume using the keys next to the numeric keypad. The longer you hold down one of these keys, the more the brightness or volume changes.

✔ To turn speaker volume off on a NeXTUSA keyboard, you can hold down the Command key and press the volume-down key. To turn the volume back on, hold down the Command key and press the volume-down key again.

✔ On a color display, you can also adjust brightness with the controls on the display itself.

See also

Get help by topic 19

Mesa Basics

Click the links below to learn how to get started; what to expect when you first launch Mesa, and how to set the application preferences; how to open, close, and save Mesa documents, how to use the IconBar and ToolBox, and how to quit the application.

See also

Launching Mesa

To start Mesa, find the Mesa icon and double click it.

The icon may be on your Dock, or within the File Viewer.

Look for the file named *Mesa.app*.

If you are unfamiliar with the NEXTSTEP operating environment, check the NEXTSTEP reference manual.

After Mesa starts, you can display a blank spreadsheet by clicking the command **Worksheet** on the Main menu and then clicking **New** on the File submenu.

✓ This can also be done by typing *Command-n.*

✓ You can open an existing model by finding it in the Workspace, selecting its icon and double-clicking.

In order to open Mesa files, Mesa ***must*** be in your "Apps" directory or another directory checked by the Workspace Manager, such as /LocalApps.

See also

Startup Preferences	*307*
New Worksheet	*33*
Open an existing Worksheet	*32*
Quitting Mesa	*43*

Open an existing Mesa worksheet

Choose the *Open* option from the *Worksheet* menu to open an existing Mesa sheet.

The names of all files with an extension of Mesa, xls (Excel files), wks or wk1 (Lotus 123 files) will be displayed in the browser.

✔ You can also open an existing sheet by finding it in the Workspace, selecting its icon and double-clicking.

See also

Start a new Mesa sheet 33

Start a new worksheet

Choose the *New* option from the *Worksheet* menu to start a new Mesa sheet.

See also

Open an existing Mesa sheet *32*

Save a Worksheet

There are several ways to save your work.

- ✓ Use *Worksheet->Save* from the Mesa menu
- ✓ Click the Close button at the top right of the worksheet
- ✓ Use *Worksheet->Save As* from the Mesa menu

If it is the first time you have saved a sheet, or if you have selected *Worksheet-> Save As,* the Save panel appears. Use the browser to select a location and name for the file.

At this stage you can also enter a password for the sheet. If you do not enter a password, the sheet will open normally without asking for a password. The password does not encrypt your sheet; it merely provides a simple form of protection.

NOTE The close box —at the top right-hand corner of a view window—displays the save state of a whole worksheet (all tables and views). A window with unsaved changes displays a dirty close box .

See also

Multiple Windows *163*
New Worksheet *33*
Open an existing Worksheet *32*
Quitting Mesa *43*

Using Menus

Just like NEXTSTEP and Openstep, Mesa has a hierarchical menu structure consisting of a main menu with subordinate menus, some of which have submenus of their own.

A menu item that displays -> opens a submenu.

A menu item followed by an ellipsis (...) opens a dialog or inspector panel.

If a menu item is *grayed out* then it is not available at the current oment. However, if you perform an action which makes that menu item possible, then the item will ungray and be usable again.

If a letter or punctuation mark appears at the far right of the menu item, that command can be executed either by clicking with the mouse on the menu item, or by holding down the *Command* key while typing the indicated character.

EG *Command-s* is the ShortCut Key for *Worksheet-Save*

✔ Menus can be dragged to any point on the screen and released to ’stick’ at that point.

✔ Menu commands can be dragged as Icons into the ToolBox by using Command-Click-Drag

See also

The ToolBox *38*
ShortCuts

Using Inspectors

The Inspector panel (accessed with Tools->Inspector or by clicking the Inspector icon on the Worksheet iconbar) provides a series of panels allowing you to manipulate items on the Mesa worksheet.

Inspectors are provided for:

Sheet (formats and Templates)
Cell Format
Cell Style
Borders
Events
Database Queries
Database Extracts
Scenarios
Layers
Labels
Charts

Each Inspector provides an easy way to examine and change information about the item you're inspecting.

QuickHelp

The main worksheet window contains Icons (the IconBar) which take you immediately to other areas of Mesa.

There are also icons in the ToolBox and in the Inspector panels.

To find out what any icon does, move the mouse to it and leave the mouse over it for a short time. A QuickHelp description will come up.

✔ Use *Info->Preferences* to change the delay time for the QuickHelp 'balloon' or to switch QuickHelp off completely.

✔ You can Drag any menu item to the ToolBVox and create your own box of useful commands. *Command-Click* on a menu item to show it's icon, and just drag ot to the place on the ToolBox you want.

See also

Preferences	*307*
ToolBox	*38*

The ToolBox

The ToolBox is a panel in which you can stick icons representing menu actions, colours, fonts and more.

Use it to speed up your access to the commands you use most often.

The ToolBox is the key to your quick and easy use of Mesa.

The Toolbox is fully customisable so that you can set up the shortcuts that *you* need. You can also resize it and position it anywhere you like on the screen.

- ✓ Use *Tools->ToolBox* or select the ToolBox icon on the Worksheet to bring up the ToolBox window.
- ✓ When you use Mesa for the first time, the ToolBox will have a standard set of icons on it. You can remove or add icons just by using Command-Drag. To add an icon to the ToolBox, put the mouse over the Menu item you want to add, hold down the Command key and click the menu item. You'll see an icon appear. Then just drag the icon into the ToolBox.
- ✓ Leave the mouse for 2 seconds over any icon to see a short description of what that icon does.
- ✓ You can set up Named Tabs and group your icons into logical sets.

See also

ToolBoxTabs *40*
QuickHelp *37*

ToolBox Tabs

You can set up Tabs in the ToolBox to make it easier to find the icons you want.

✔ Double-Click on a Tab to go into Edit mode so that you can give it a name

✔ Press Return on a Tab to create a New Tab

✔ Press Delete on a Tab to delete a Tab. This will remove all the icons within that TabSet from the ToolBox

See also

ToolBox 38

Getting Help

Use *Info->Help* to raise the standard Help panel.

You can search for topics, browse the index, and backtrack through your help pages.

Use *Info->SlashHelp* to see the Mesa equivalent of many Lotus 1-2-3 commands

✓ Press F1 (or Alt-?) when the mouse is over any menu item to bring up the Help page for that item.

✓ Leave the mouse for 2 seconds over any icon to see a short description of what that icon does. This is called *QuickHelp.*

See also

QuickHelp *42*
The ToolBox *38*

QuickHelp

Icons can be confusing. Mesa lets you see a short description of any icon when you position the mouse on that icon and leave it there for 2 seconds.

✔ Use *Info->Preferences* to customise the QuickHelp delay time to suit your needs.

See also

Worksheet basics	*30*
The ToolBox	*38*

Quitting Mesa

To finish a Mesa session, take the *Quit* option from the main menu.

If you have any unsaved worksheets open, Mesa will prompt you asking whether you want to save them or not (individually) before quitting.

The Worksheet main window

The Worksheet main window consists of three main areas.

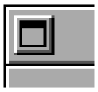
The Spreadsheet: a set of cells laid out in rows and columns

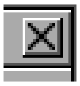
The Formula Bar: where you type or view the text, numbers or formulas in the cells

The IconBar: a set of icons which act as shortcuts to frequently-used areas of Mesa

---

✔ As in other NEXTSTEP/Openstep applications, there are Window tools which can be selected to Miniaturize or Close the window.


Miniaturize


Close

✔ The Worksheet window can be enlarged or shrunk by positioning the mouse in any corner of the window and dragging .

See also

Rows and Columns	*46*
Entering information	*47*
The ToolBox	*38*
Using Menus	*35*

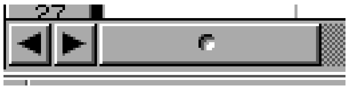
Moving around the Sheet

You can move around the worksheet in many ways:

The cursor keys will move you one cell in the appropriate direction

Clicking in a cell with the mouse will move you to that cell

The Page-Up/Page Down indicators in the Window Frame will move you one screen up or down



The Scroll bars will move you gently in the direction you choose with the mouse

---

- ✔ ESC will *either* Abandon what you're typing *or* entter an Move to A1, depending on your Preferences setting
- ✔ On an Intel or Sun system, the Home key will take you to A1. PgUp and PgDn will move you up and down the worksheet one screenful at a time.
- ✔ The Tab key will move you one screenful to the right: Shift-Tab will move you one screen to the left
- ✔ There are lots of other ShortCut key combinations to move you around (see below)
- ✔ *Cell->Goto* will take you straight to any cell you specify
- ✔ *Edit->Find* is a handy way to jump to part of your worksheet by Cell Content.

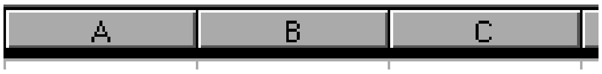
See also

ShortCut Keys
Cell Range Labels *211*
Find *62*

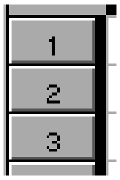
Rows and Columns

All cells in the worksheet are accessed by specifying the intersecting row and column of the cell. For example, the cell in the upper left corner of the worksheet is cell A1.

Columns



Rows



The number in the Row Frame designates the row number.

The letter in the Column Frame designates the column letter.

✓ Click on the Row Number in the Frame to select an entire row

✓ Click on the Column letter in the Frame to select an entire column

See also

Ranges *211*
Moving around the worksheet *45*

Entering Data

Cells contain and display one of four things:

✔ Text strings, such as	Profits
✔ Values, such as	12
✔ Dates, such as	21-02-1996
✔ Formulas, such as	=A4+A5

To enter data into an editable cell:

1 Move to the cell
2 In the Formula Bar, enter a number, date or text string.
3 Move away from the cell to enter the value

✔ Values and dates can be formatted in any number of differing formats. For example, the number 12 can be displayed as "$12.00," "£12," "12" or "1.2e+01."

✔ Once data is in a cell, it and its format can be copied and pasted into other cells.

See also

Entering Text *48*
Entering Numbers *49*
Entering Dates *50*
Formulas *66*
Deleting Data *61*
Editing Data *54*

Entering Text

Anything you type into the Formula Bar which contains a non-numerical character is considered by Mesa to be **text.**

The exceptions to the above rule are items starting with the Formula Indicators =, +, -, and @

✔ Text strings are automatically left-aligned in a cell. You can change the alignment by using the *Cell Format Inspector.*

✔ Text strings have a value of zero if they are included (usually by mistake) in the range of a calculation

See also

Entering Numbers	*49*
Entering Dates	*50*
Formulas	*66*
Deleting Data	*61*
Editing Data	*54*

Entering Numbers

Anything you type into the Formula Bar which contains nothing but numerical characters is considered by Mesa to be **a number.**

✓ Numbers are automatically right-aligned in a cell. You can change the alignment by using the *Cell Format Inspector.*

See also

Entering Text	*48*
Entering Dates	*50*
Formulas	*66*
Deleting Data	*61*
Editing Data	*54*

Entering Dates

Dates have to be entered using a specific format.

Use the **slash key** / as a separator if you are entering dates in the American format (MM/DD/YY)

Use the **period** as a separator if you are entering dates in the European format (DD.MM.YY)

In all cases, the date will display automatically in the format DD-Mon-YY

So the entries:

12/25/96

and

25.12.96

will both show correctly as

25-Dec-96

✔ You can format dates to look different (for example, showing just the Month and Year) by using the Cell Format Inspector.

✔ Mesa can do calculations with dates. For example, the formula:

= 1/15/97 - 1/10/97

will give the answer 5.

✔ Use the function =now() to enter the current date.

See also

Entering Text *48*
Entering Numbers *49*
Formulas *66*
Deleting Data *61*
Editing Data *54*

Copying data

Copying is a two-stage operation.

Select the range to be copied. Click *Edit->Copy.* Select the cell into which you want to put the data, and click *Edit->Paste* .

If you copy a formula, the cell references change relative to the new cell position of the formula.

If you copy text, numbers or dates, the copy is identical to the oiriginal entry.

✓ Use *Alt-Click* as a ShortCut key to copy a cell or range of cells. Select the range to copy, and then Alt-Click on the top left cell of the position you want to paste to..

✓ *Range->Fill or Range->SmartFill* is often a quicker way to copy data.

✓ Data selected by a copy operation can be pasted into other NeXTSTEP/Openstep applications.

See also

Paste *53*
Paste Special *55*
Fill *63*
Smart Fill *64*

Moving data

Moving Data is a two-stage operation.

Select the range to be copied. Click *Edit->Cut.* Select the cell into which you want to put the data, and click *Edit->Paste* .

If you copy a formula, the cell references change relative to the new cell position of the formula.

If you cut text, numbers or dates, the copy is identical to the oiriginal entry.

✓ You don't have to paste after a cut. It can be used as another way of deleting data from the worksheet.

✓ Data selected by a cut operation can be pasted into other NeXTSTEP/Openstep applications.

See also

Paste *53*
Paste Special *55*
Fill *63*
Smart Fill *64*

Pasting data

Use the Paste command to place the contents of the NEXTSTEP pasteboard wherever you want on any worksheet.

If the Pasteboard contains a TIFF or EPS image, Mesa will overlay that graphic image on your sheet.

- ✓ You can Paste the current contents of the pasteboard over and over again.
- ✓ Data from other NeXTSTEP/Openstep applications can be pasted into a Mesa worksheet.

See also

Editing Data

You edit data by selecting the cell you want to change, and clicking in the Formula Entry area at the top of the worksheet.

You can use all the standard NeXTSTEP text selection operations to cut, copy, delete etc your data.

When you finish editing, either press Enter or click away from the cell to save your changes

---

✔ Use ESC to abandon any changes you’ve made.

✔ Remeber that you can drag the Formula Area larger or smaller. If you remove the formula area completely, you’ll be able to edit the cell contentc ’in-cell’.

See also

Paste Special

Paste Special allows you to take data from the Pasteboard and paste it in - with a twist!

Paste Special is most commonly used to paste just data (ie no formatting information) or conversely just formatting information (ie no data) from another cell on the worksheet.

Paste Values lets you paste the values from the cell range that was cut or copied into the pasteboard to another range without affecting its format or content. Note that the values in the pasted range will be frozen and formulas will be converted to their calculated values.

Paste Format lets you paste the format of the cell range that was cut or copied into the pasteboard to a specified location on a worksheet without pasting any included values.

Move lets you paste the contents of the cell range that was cut or copied into the pasteboard to a specified location on a worksheet. If the original range was copied to the pasteboard, this command clears the original range as it moves the data to the new one. Any references within the range to cells outside of the range still point to the same place.

✔ You cannot Move ranges from one worksheet to another. To move a range from one worksheet to another, select the range in the first worksheet, **Edit** → **Cut** it, and **Edit** → **Paste** it into the second worksheet.

✔ The Move operation is different from a Cut and Paste operation. If you Cut and Paste a range of cells, formulas that referred to the old location of the range continue to refer that location. If you *Copy* and *Paste Special ->Move* the cells, formulas elsewhere in your worksheet are updated to reflect the new location of the range.

Transpose places a copy of the cells on the clipboard at the current location on the worksheet, but transposes the rows and columns of the range. The cell in the second row and first column in the original range is placed in the first row and second column, etc.

Add adds the value in each cell on the pasteboard to the values in each cell of the selected range.

Subtract subtracts the value in each cell on the pasteboard from the values in each cell of the selected range.

Multiply multiplies the values in the selected range by the value of each cell on the pasteboard.

Divide divides values in each cell of the selected range by the value on the pasteboard.

See also

Paste	*53*
Fill	*63*
Smart Fill	*64*

Undo

Use *Edit->Undo* to cancel any typing you have done in a cell and revert to the previous contents of that cell.

You can also use ESC to achieve the same thing.

NOTE: this is a much less powerful command than you may be used to in Excel or 1-2-3. Mesa does not yet support multi-level Undo and Redo.

Once you have pressed Enter (or taken any other action to confirm the contents of a cell). Undo will have no effect on that cell.

See also

Redo *58*

Redo

Use *Edit->Redo* to reverse the effect of *Edit->Undo*

If you have Undone a typing change, then *Redo* will set the contents of the cell back to the original.

NOTE This is a much less powerful command than you may be used to in Excel or 1-2-3. Mesa does not yet support multi-level Undo and Redo.

See also

Undo *57*

Clearing data

"Clearing" is used in Mesa to mean removal of the data and formatting information in a selection.

This is not the same as "Deleting" a selection. For example, if you Delete a row, the entire row is cut out of the worksheet. If you Clear a row, the data in the cells is removed but the row is still there.

Use *Edit->Clear* to clear a selection.

✓ You can use *Edit->Clear* to remove a graph from the worksheet.

✓ You can use the *BackSpace* key to clear the contents of a cell.

✓ Be careful! Because Mesa does not have a multi-level Undo, you can't change your mind!

See also

Clear Special

Clear Special allows you to clear only certain aspects of the selected cells.

You can clear:

- just the formats (eg font, color, borders)
- just the data (leaving the formatting intact)
- Numbers only (leaving all text intact)
- Strings only (leaving all numbers intact)

Deleting data

"Deleting" is used in Mesa to mean complete removal of the selected item from the worksheet.

✓ This is not the same as "Clearing" or "Blanking" a selection. For example, if you Delete a row, the entire row is cut out of the worksheet. If you Clear the same row, the data in the cells is removed but the row is still there.

Use *Edit->Delete* to delete a selection.

Cols To delete columns, highlight the columns you wish to delete, and click the Delete command. Columns to the right of the deleted columns will be shifted to the left appropriately.

Rows To delete rows, highlight the rows you wish to delete, and click the Delete command. The Rows below the deleted rows will be shifted up appropriately.

Cells If you highlight one or more cells on your worksheet and click Delete, the data in the affected row(s) will shift to the left by the number of cells you highlighted.

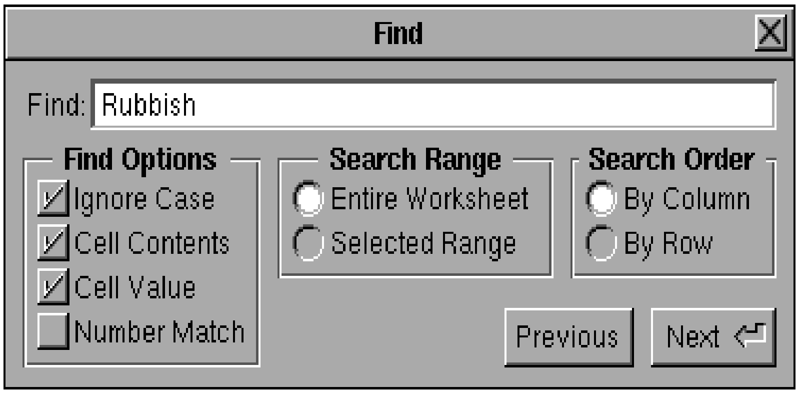
✓ Be careful! Because Mesa does not have a multi-level Undo, you can't change your mind!

See also

Clearing data *59*
Editing Data *54*

Find

Use *Edit->Find* to perform quick and easy searches of your worksheet in order to move to a particular cell.



Fill in the panel with the information you want to find.

Next finds the next occurrence of the data moving Down the sheet

Previous finds the next occurrence of the data moving Up the sheet

✓ There is no Replace option available within the Find panel

See also

Database Extract *195*
GoTo *316*

Fill with data

You can use *Range->Fill* instead of a Copy + Paste sequence if you want to copy the contents of a cell to **adjacent cells.**

Why? Fill is quicker because it uses only one command. The output from a Fill comand is identical to that from a Copy + Paste sequence.

Select the cell containing the data you want to copy, and Drag to also select all the cells you want to copy the data into. Then use the appropriate *Range->Fill* command to paste the data in.

✓ *Fill* and *SmartFill* do different things. Read about both of them before deciding which is most appropriate for you.

See also

Fill Smart *64*
Paste *53*
Copy *51*

SmartFill

Use *Fill->Smart* to fill ranges with sequences of data.

Select the cell containing the data you want to start the sequence with, and Drag to also select all the cells you want to copy the data into. Then use the appropriate *Range->SmartFill* command to paste the data in.

✔ If the starter cell contains **text**, Mesa checks to see whether it recognises the text. If so, the sequential cels are filled in using the NEXT() function.   If not, Mesa tags a 1 on the end of the text and increases the number by1 in subsequent cells.

✔ If the starter cell contains a **number**, Mesa increases the number by 1 in subsequent cells using the NEXT() function.

✔ If the starter cell contains a **formula**, Mesa mimics the normal Copy+Paste operation using the SAME() function.

In all cases, if the contents of the starter cell is changed, the adjacent cells will automatically take on new values based on the new starter cell. This is especially useful if a formula has to be changed.

✔ *Shift-Return* is a useful ShortCut key for SmartFill

✔ **Time Sequences.** Mesa knows about

Days of the week
Months of the year
Quarters
Quarters and Years
Numbers
Dates

For example, if you enter 'May' into A1 and SmartFill across to H1, the cells fill with 'May' through to 'November'.

See also

Paste *53*
Copy *51*

- NEXT()
- SAME()

Formulas

Calculations= or '*formulas*' are used to perform the main function of any spreadsheet - working with numbers.

Formulas can be short and simple or long and as complex as you require. They can contain:

- numbers
- cell addresses
- Functions

To show Mesa that you're putting a Formula into a cell, put the Equals sign = first.

Here are some example formulas:

=A1+A2

=B3+25

=SUM(J2.K50)

✓ Use the Formula Builder to help set up complex formulas or find out more about Mesa Functions.

Entering a Formula into a Cell

Mesa recognizes an expression entered into a cell as a formula by the first character of the expression. Formulas must begin with @, =, +, or -. The functions in a formula are assumed to be the non @ character version of the functions unless the @ character is included with each function, even if the formula began with an @ character. This means that in formulas that contain a mixture of @ and non-@ functions, only the functions preceded by an @ character are @ functions.
In Mesa, you can enter a formula into a cell by typing it directly into the formula bar the same way you would enter numbers or strings, or you can build a formula with the Formula Builder.

The Formula Builder is an inspector that contains a scrollable list of all of Mesa's functions, divided up into different categories. Clicking on a function will give you a brief description of the function. The Formula Builder can be used to paste the different functions together.

Use *Tools->Formula Builder* or click the Formula Builder icon on the Worksheet IconBar to start the Formula Builder

The highlighted function will appear in the formula bar. Edit any arguments the function has by deleting the variable names from the function as it appears in the formula bar, and replace them with appropriate values. By pasting in operators and other functions, you can create large, complex formulas easily.

Converting Data

The *Range->Convert* submenu has the following commands:

Formulas to Values freezes the current values in the cell range that you selected. If you recalculate the worksheet, the frozen values will not change.

Strings to Numbers converts the string values in a given range to their number value. If a string cannot be converted, the cell is set to 0.

Numbers to Strings converts numeric entries in a range to their string value.

The *Format->Text->Case* submenu has the following commands:

To Upper Case converts string data in the cell or cell range that you selected to all capital letters.

To Lower Case converts string data in the selected cell or cell range to all lower case letters.

To Proper Case capitalizes the first letter of each word of string data in the selected cell or cell range

Functions and @Functions

In order to provide compatibility with other spreadsheets, Mesa provides functions both with and without a leading @ character. The two sets of functions are very similar, but not always identical, when they are not the same, the @ version of the function will be given a separate entry.

Functions without the @ character are compatible with Microsoft Excel functions; function with the leading @ character are compatible with Lotus 1-2-3 functions.

Function Arguments

Many of Mesa's functions can take a variety of different arguments. In almost all cases, Mesa's functions can take the specified type of argument, an expression that returns the specified type of argument, or the address of a cell containing the specified type of argument. The following is a brief explanation of what can be used as arguments in Mesa's functions.

Numbers

Many of Mesa's functions take numbers as arguments. The numbers can be entered directly into the formula, or they can be given as the cell address of a cell containing to number. A blank cell has a value of zero.
Occasionally a function needs a specific type of number. Here is a list of some of the specific types of numbers that functions may take.

integer: a number with no fractional portion.

1, 42, -28, 589

positive number: any number with a value of zero or greater.

0, 28, 39.541, 0.028

Strings

When strings are needed as arguments to functions, they can either be typed directly into the formula, in which case they need to be place in quotation marks, or the address of the cell containing the string may be given, in which case the contents of the cell does not have to be in quotation marks. For example, the following expressions both return START.

```
=UPPER(``start'')
```

```
        A
1       start
=UPPER(A1)
```

Dates

Mesa has a number of date related functions that take dates as their arguments. In most cases these functions take a date serial number as their argument. (Specific exceptions are given in the function descriptions, see “Date Functions” 88.)

A date serial number is a number used by Mesa to represent dates so that they may be formatted in a number of different ways. The number corresponds to the number of days since January 1, 1970. Dates after the year 2000 can be specified by placing a 1 before the year. The easiest way to obtain a date serial number is to enter the desired date into a cell as you would normally, and give the address of the cell containing the date.

Cell Addresses

A cell address in Mesa consists of the letter corresponding to the column of the cell, and the number of the row of the cell. For instance, cell A1 is the cell in the upper left hand corner of the worksheet. It is the cell in row one and column A.

Some Mesa functions take a cell address as their argument. In most Mesa functions, when you enter the address of a cell as an argument, the function uses the contents of that cell as the argument. In a few functions, however, the function is looking for information about the cell itself. This is the case for functions that ask specifically for a cell address as an argument.
An example of a function that takes a cell address as its argument is the function ISEMPTY, which checks to see if the cell at a given address is empty.

```
=ISEMPTY(U2)
```

Ranges

Ranges can also be used as arguments in Mesa's functions. In some cases, the function can take a list of numbers as arguments, and a range that contains an appropriate list may be entered as an argument. In other cases, a function may need to know the size an location of a database, and therefore needs a range as an argument.

Ranges can be specified in one of two ways. The simplest way is to specify it by giving the address of the cells in the upper left corner and lower right corner of the range separated by a colon. For example A1:F10 is the range that occupies rows 1 through 10 of columns A through F. You can also use the name of a ranged that was named using the Label Inspector (See "Label Inspector"216). If you had used the Label Inspector to create a range named Rates, you could enter *Rates* as a range argument.

```
=ROWS(A1:F10) -> 10
=ROWS(Rates)
```

Constants

Mesa has a number of function constants that can be used as arguments in Mesa's functions. For a complete explanation of Mesa's constants, see "Constants" 73.

Expressions

The arguments to functions can contain calls to other functions as well as formulas containing constants and operators. As long as the formulas returns an appropriate value, it may be used as an argument for another function. In this way you can create complex formulas using Mesa's functions.

Constants

Color Constants

Color constants are used by NEXTSTEP to specify how the amount of red, blue, and green present in a given color. Color constants can be used in Mesa's functions to manipulate the color of cells and ranges. Here is a list of the available color constants.

BLACKCOLOR, @BLACKCOLOR
BLUECOLOR, @BLUECOLOR
CYANCOLOR, @CYANCOLOR
DKGRAYCOLOR, @DKGRAYCOLOR
GREENCOLOR, @GREENCOLOR
LTGRAYCOLOR, @LTGRAYCOLOR
MAGENTACOLOR, @MAGENTACOLOR
REDCOLOR, @REDCOLOR
WHITECOLOR, @WHITECOLOR
YELLOWCOLOR, @YELLOWCOLOR

Format Constants

Mesa uses the following format constants in its formatting functions. These formats correspond to the different cell style formats in the Cell Style Inspector. (See “Cell Style Inspector”*169*.)

GENERALFORMAT, @GENERALFORMAT: This constant is used to specify the general format.
`=FORMAT(238.093, GENERALFORMAT) -> 238.093`

FIXEDFORMAT, @FIXEDFORMAT: This constant is used to specify the fixed decimal format.
`=FORMAT(42.692, FIXEDFORMAT) -> 42.69`

SCIENTIFICFORMAT, @SCIENTIFICFORMAT: This constant is used to specify the scientific notation format.
`=FORMAT(1.69, SCIENTIFICFORMAT) -> 1.89e+01`

CURRENCYFORMAT, @CURRENCYFORMAT: This constant is used to specify the currency format.
`=FORMAT(555.9, CURRENCYFORMAT) -> $555.90`

COMMAFORMAT, @COMMAFORMAT: This constant is used to specify the comma format.
`=FORMAT(238.093, COMMAFORMAT) -> 238.09`

CHARTFORMAT, @CHARTFORMAT: This constant is used to specify the chart format.

```
=FORMAT(7, CHARTFORMAT) -> +++++++
```

PERCENTFORMAT, @PERCENTFORMAT: This constant is used to specify the percent format.

```
=FORMAT(0.09389, PERCENTFORMAT, 2) -> 9.39%
```

DMYFORMAT, @DMYFORMAT: This constant is used to specify the date day-month-year format.

```
=FORMAT(9876.54321, DMYFORMAT) -> 15-Jan-97
```

DAYMONFORMAT, @DAYMONFORMAT: This constant is used to specify date day-month format.

```
=FORMAT(9876.54321, DAYMONTHFORMAT)
-> 15-Jan
```

MONYEARFORMAT, @MONYEARFORMAT: This constant is used to specify date month-year format.

```
=FORMAT(9876.54321, MONYEARFORMAT)
-> Jan-97
```

HMSFORMAT, @HMSFORMAT: This constant is used to specify date hours-minutes-seconds format.

```
=FORMAT(9876.54321, HMSFORMAT)
-> 1:02:13 PM
```

HOURMINFORMAT, @HOURMINFORMAT: This constant is used to specify date hours-minutes format.

```
=FORMAT(9876.54321, HOURMINFORMAT)
-> 1:02 PM
```

DINTL1FORMAT, @DINTL1FORMAT: This constant is used to specify the numeric date month-day-year format.

```
=FORMAT(9876.54321, DINTL1FORMAT)
-> 01/15/97
```

DINTL2FORMAT, @DINTL2FORMAT: This constant is used to specify the numeric date month-day format.

```
=FORMAT(9876.54321, DINT2FORMAT) -> 01/15
```

TINTL1FORMAT, @TINTL1FORMAT: This constant is used to specify twenty four hour time format with seconds.

```
=FORMAT(9876.54321, TINTL1FORMAT)
 -> 13:02:13
```

TINTL2FORMAT, @TINTL2FORMAT: This constant is used to specify twenty four hour time format without seconds.

```
=FORMAT(9876.54321, TINTL2FORMAT) -> 13:02
```

Math Constants

Mesa uses the following mathematical constants in its functions.

PI, @PI: The area of a circle of radius one, approximately 3.14159.

RANDOM, RAND, @RAND: This function returns a random number between zero and one, including zero and excluding one.

Date Constants

Mesa uses the following date constants in its date functions to specify current date or time.

NOW, @NOW: Returns the current date serial number.

TODAY, @TODAY: Returns the current date serial number without the current time, so that the serial number only contains the day month and year information.

Special Constants

CURRENTNUMBER or @CURRENTNUMBER: The current numeric value of a cell. Can be used for time series functions. To reset a cell containing this constant, use **Edit → Clear Special → Reset.**

```
To remember the largest number entered into
cell A1, use
=if(a1 > currentnumber,a1,currentnumber).

To count the number of times the worksheet
was recalculated, use
currentnumber + 1
```

CURRENTVALUE or @CURRENTVALUE: The current value of the cell (before the recalc). This is useful for comparing old cell values with other values, such as taking the larger of the current value or the value of another cell.To reset a cell containing this constant, use **Edit → Clear Special → Reset.**

ERR or @ERR: Returns an error. All cells that refer to a cell with an error will evaluate to errors as well.

ERRORCONST: Returns an error. All cells that refer to a cell with an error will evaluate to errors as well.

FALSE(), FALSE or @FALSE: Returns a logical false value.

FILENAME or @FILENAME: Returns the filename of the current worksheet.

HOSTNAME, or @HOSTNAME: Returns the name of the machine that Mesa is running on.

NA or @NA: Returns a special error, NA. All cells that refer to a cell with an error will evaluate to errors as well. @NA is useful to designate cells that have a value that is not available or defined.

NACONST: Returns a special error, NA. All cells that refer to a cell with an error will evaluate to errors as well. NACONST is useful to designate cells that have a value that is not available or defined.

PATHNAME or @PATHNAME: Returns the path of the current worksheet.

THISCOLUMN: Returns the column number of the cell containing this constant.

THISROW: Returns the row number of the cell containing this constant.

TRUE(), TRUE or @TRUE: Returns a logical true value.

USERNAME, or @USERNAME: Returns the user's login name.

Database Functions

For more information about criteria ranges, see “Criteria Ranges” *197*.

DAVE, or DAVERAGE

Takes the average of numeric records that match criteria.

DAVE(*database, offset, crit*)

database the range address of the database

offset
the offset of the column to be averaged

crit
the criteria range

This function takes a range and averages all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
       A      B     C
1    NAME   GRADE  GPA    GRADE
2    Joe    B        3.2   B
3    Sue    A        4.7
4    Robert C        2.7
5    Steve  A        4.5
6    Ann    B        4.2

The database range is A1:C6. The criteria
range is D1:D2.

To calculate the average GPA of everyone who
earned a B use this formula:

=DAVE(A1:C6,2,D1:D2) -> 3.7
```

@DAVE or @DAVG

Takes the average of numeric records that match criteria. Strings have a value of zero.

@DAVE(*database, offset, crit*)

database the range address of the database

offset
the offset of the column to be averaged

crit
the criteria range

This function takes a range and averages all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. This function is basically the same as DAVE() or DAVERAGE() except that those function ignore string values.

```
This is a sample database:
       A      B     C
1    NAME   GRADE  GPA    GRADE
2    Joe    B       3.2   B
3    Sue    A       4.7
4    Robert C       2.7
5    Steve  A       4.5
6    Ann    B       4.2

The database range is A1:C6. The criteria
range is D1:D2.

To calculate the average GPA of everyone who
earned a B use this formula:

@DAVE(A1:C6,2,D1:D2) -> 3.7
```

DCOUNT or @DCOUNT

Counts all the numeric items matching criteria in the column offset from the upper left corner of the database range.

DCOUNT(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and counts all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
       A      B     C       D
1    NAME   GRADE  GPA    GRADE
2    Joe    B       3.2   B
3    Sue    A       4.7
4    Robert C       2.7
5    Steve  A       4.5
6    Ann    B       4.2

The database range is A1:C6. The criteria
range is D1:D2.
```

```
To count the number of students who earned
B's use this formula:

=DCOUNT(A1:C6,1,D1:D2) -> 2
```

DMAX

Returns the maximum numeric value of the records in database that match criteria.

DMAX(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the maximum number of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
       A      B      C        D
1    NAME   SALES  REGION  REGION
2    Joe     3000  East    East
3    Sue     4500  North
4    Robert  2800  East
5    Steve   3700  South
6    Ann     2400  West

The database range is A1:C6.  The criteria
range is D1:D2.

To find the largest sales figure in the
Eastern region use this formula:
=DMAX(A1:C6,1,D1:D2) = 3000
```

@DMAX

Returns the maximum value of the records in database that match criteria. Strings have a value of zero.

@DMAX(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the highest number of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. Strings are assumed to have a value of zero. This function is basically the same as DMAX except that those function ignore string values

```
This is a sample database:
       A      B      C        D
1    NAME   SALES  REGION  REGION
2    Joe     3000  East    East
3    Sue     4500  North
4    Robert  2800  East
5    Steve   3700  South
6    Ann     2400  West

The database range is A1:C6. The criteria
range is D1:D2.

To find the largest sales figure in the
Eastern region use this formula:
@DMAX(A1:C6,1,D1:D2) -> 3000
```

DMAX() and @DMAX differ in that DMAX ignores string values and @DMAX treats strings as having a value of zero.

DMIN

Returns the smallest numeric item in the records that match the criteria.

DMIN(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the minimum number of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
       A      B      C        D
1    NAME   SALES  REGION  REGION
2    Joe     3000  East    East
3    Sue     4500  North
4    Robert  2800  East
5    Steve   3700  South
```

```
6    Ann     2400  West

The database range is A1:C6. The criteria
range is D1:D2.

To find the smallest sales figure in the
Eastern region use this formula:
=DMIN(A1:C6,1,D1:D2) = 2800
```

@DMIN

Returns the smallest numeric item in the records that match the criteria. String values will count as zeros.

@DMIN(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the minimum number of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. Strings are assumed to have a value of zero. This function is basically the same as DMIN except that those function ignore string values

```
This is a sample database:
       A      B     C        D
1    NAME   SALES  REGION  REGION
2    Joe     3000  East    East
3    Sue     4500  North
4    Robert  2800  East
5    Steve   3700  South
6    Ann     2400  West

The database range is A1:C6.  The criteria
range is D1:D2.

To find the smallest sales figure in the
Eastern region use this formula:
@DMIN(A1:C6,1,D1:D2) -> 2800
```

DMIN() and @DMIN() differ in that DMIN() ignores strings, while @DMIN() treats strings as having a value of zero.

DPROD

Returns the product of all the items matching the criteria in the column offset from the upper left corner of the range.

Strings are ignored.

DPROD(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and multiplies all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
      A      B         C         D
1   GATE   PATH   % PASSING    PATH
2   1      North       0.96    East
3   2      South       0.95
4   3      West        0.97
5   4      East        0.95
6   5      East        0.93
7   6      North       0.88
8   7      West        0.92
9   8      East        0.99
10  9      South       0.94

The database range is A1:C10. The criteria
range is D1:D2.

To find the product of the percentages for
the East path use this formula:
=DPROD(A1:C10, 2, D1:D2) -> 0.874665
```

@DPROD

Returns the product of all the items matching the criteria in the column offset from the upper left corner of the range. Strings are treated as zero values.

@DPROD(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and multiplies all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. Strings are treated as having a value of zero. This function is basically the same as DPROD except that those function ignore string values

```
This is a sample database:
      A      B         C         D
1   GATE   PATH   % PASSING    PATH
2   1      North       0.96    East
3   2      South       0.95
4   3      West        0.97
5   4      East        0.95
6   5      East        0.93
7   6      North       0.88
8   7      West        0.92
9   8      East        0.99
10  9      South       0.94
```

```
The database range is A1:C10.  The criteria
range is D1:D2.
```

```
To find the product of the percentages for
the East path use this formula:
@DPROD(A1:C10, 2, D1:D2) -> 0.874665
```

DPROD() and @DPROD differ in that DPROD() ignores string values, but @DPROD() treats strings as having a value of zero.

DSTDDEV or DSTDEV

Returns the standard deviation of the numeric records that match the criteria.

DSTDEV(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the standard deviation of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
      A       B        C        D
1   STATE   REGION   AVG RF   REGION
2   AL      South       3.7   West
3   CA      West        2.1
4   NJ      East        5.2
5   FL      South       4.9
6   LA      South       6.8
```

```
7   TX      West        5.0
8   MA      North       7.6
9   AZ      West        2.2
10  NV      West        2.4
```

```
The database range is A1:C10. The criteria
range is D1:D2.
```

```
To find the standard deviation of rainfall in
the Western region use this formula:
=DSTDEV(A1:C10,2,D1:D2) -> 1.2029
```

@DSTD

Returns the standard deviation of the records that match the criteria. Strings have a value of zero.

@DSTD(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the standard deviation of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. Strings are assumed to have a value of zero. This function is basically the same as DSTDEV except that those function ignore string values

```
This is a sample database:
       A      B     C        D
1    NAME   SALES  REGION  REGION
2    Joe     3000  East    East
3    Sue     4500  North
4    Robert  2800  East
5    Steve   3700  South
6    Ann     2400  West
The database range is A1:C10. The criteria
range is D1:D2.
```

```
To find the standard deviation of rainfall in
the Western region use this formula:
@DSTD(A1:C10,2,D1:D2) -> 1.2029
```

DSUM or @DSUM

Sums all the items matching criteria in the column offset from the upper left corner of the database range.

DSUM(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and sums all the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
      A       B        C        D
1   STATE   REGION   AVG RF   REGION
2   AL      South       3.7   South
3   CA      West        2.1
4   NJ      East        5.2
5   FL      South       4.9
6   LA      South       6.8
7   TX      West        5.0
8   MA      North       7.6
9   AZ      West        2.2
10  NV      West        2.4

The database range is A1:C10. The criteria
range is D1:D2.

To find the total amount of rainfall for the
Southern region use this formula:
=DSUM(A1:C10,2,D1:D2) -> 15.4
```

DVAR

Returns the variance of the numeric records that match the criteria. Strings are ignored.

DVAR(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the variance of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column.

```
This is a sample database:
      A       B        C        D
1   STATE   REGION   AVG RF   REGION
```

2	AL	South	3.7	West
3	CA	West	2.1	South
4	NJ	East	5.2	
5	FL	South	4.9	
6	LA	South	6.8	
7	TX	West	5.0	
8	MA	North	7.6	
9	AZ	West	2.2	
10	NV	West	2.4	

The database range is A1:C10. The criteria range is D1:D2.

To find the variance of rainfall in the Southern and Western regions use this formula:
=DVAR(A1:C10,2,D1:D3) -> 2.7192

@DVAR

Returns the variance of the records that match the criteria. Strings have a value of zero.

@DVAR(*database, offset, crit*)

database the range address of the database
offset
the offset of the relevant column
crit
the criteria range

This function takes a range and finds the variance of the numbers in the column at the offset value from the first column of the range that meet the criteria defined in another column. Strings are treated as having a value of zero. This function is basically the same as DVAR except that those function ignore string values

This is a sample database:

	A	B	C	D
1	STATE	REGION	AVG RF	REGION
2	AL	South	3.7	West
3	CA	West	2.1	South
4	NJ	East	5.2	
5	FL	South	4.9	
6	LA	South	6.8	
7	TX	West	5.0	
8	MA	North	7.6	
9	AZ	West	2.2	
10	NV	West	2.4	

The database range is A1:C10. The criteria range is D1:D3.

To find the variance of rainfall in the

```
Southern and Western regions use this
formula:
@DVAR(A1:C10,2,D1:D3) -> 2.7192
```

DVAR() and @DVAR() are different in that DVAR() ignores string values, while @DVAR treats strings as having a value of zero.

Date Functions

CONVDATE or **@CONVDATE**

Converts a number in the format YYMMDD to a date serial number.

CONVDATE(*date*)

date a date given in YYMMDD format

Mesa stores dates in terms of date serial numbers that can be used in various date functions, and then translated back into the desired format. Dates after the year 2000 can be specified by placing a 1 before the year.

```
=CONVDATE(930210) -> 8441 : This is the
date-number of 10-Feb-93
=CONVDATE(1100410) -> 14709 : This returns
the date number of 10-Apr-2010.
```

DATE or **@DATE**

Returns a date serial number for the year, month and day.

DATE(*yr, mon, day*)

yr the year since 1900

mon the month

day the day of the month

A date serial number is a number used by Mesa to represent dates so that they may be formatted in a number of different ways. The number corresponds to the number of days since January 1, 1970. Dates after the year 2000 can be specified by placing a 1 before the year.

```
=DATE(92,8,24) -> 8271 : This is the date
number for 24-Aug-92.

=DATE(79,10,31) -> 3590 : This is the
date-number for 31-Oct-79.
```

DATEVALUE, DAYS or **@DATEVALUE**

Datevalue converts a date string to a date serial number. If the value is a number it just returns the number.

DATEVALUE(*dtstr*), DATEVALUE(*num*)

dtstr a string corresponding to a date

num any number

This function converts dates entered as strings into the date serial number used by Mesa's date functions. This function will not effect numeric input.

```
=DATEVALUE("02/04/92") = 8069
A3: 'Jul-94

=DATEVALUE(A3) = 8947 : This is the date
number for 1-Jul-94
```

DAY or @DAY

Returns the day of the month for a date serial number.

DAY(*date*)

date a date serial number

DAY returns the day of the month for a date serial number *date*. The date can also be a reference to a cell containing a date.

```
@day(9131) ->
        4
A3: 18-Jul-92
=DAY(A3) -> 18

=DAY(NOW)  :  This formula will return the
current day of the month.
```

HOUR or @HOUR

Returns the hour for a date serial number.

HOUR(*date*)

date a date serial number

HOUR returns the hour for a date serial number *date*. The date can also be a reference to a cell containing a date. If a time is specified in a cell, but no date is specified, the integer portion of the date serial number is assumed to be zero.

```
=HOUR(TIMEVALUE("23:00:00")) -> 23
=HOUR(NOW) : This formula will return the
```

```
current hour of day.
```

MINUTE or **@MINUTE**

Returns the minute for a date serial number.

MINUTE(*date*)

date a date serial number

MINUTE returns the minutes from a date serial number *date*. The date can also be a reference to a cell containing a date. If a time is specified in a cell, but no date is specified, the integer portion of the date serial number is assumed to be zero.

```
=MINUTE(TIMEVALUE("11:35:20")) -> 35
=MINUTE(NOW) : This formula will return the
current number of minutes past the hour.
```

MONTH or **@MONTH**

Returns the month for a date serial number.

MONTH(*date*)

date a date serial number

MONTH returns the month of the year for a date serial number *date*. The date can also be a reference to a cell containing a date.

```
=MONTH(DATEVALUE("12/24/92")) -> 12

=CHOOSE(MONTH(NOW)-1,"Jan","Feb","Mar","Apr",'
:  This formula will display the current
month.
```

NETWORKDAYS or **@NETWORKDAYS**

Returns the number of working days between the given dates.

NETWORKDAYS(*date1*, *date2* [,*holidays*])

date1 the starting date serial number

date2 the ending date serial number

holidays the serial numbers of any intermediate holidays

This function allows you to determine the number of working days between the two dates *date1* and *date2*. You can optionally list any holidays that fall between the two dates, which will not be counted as working days.

```
=NETWORKDAYS(NOW,NOW+10) : This returns the
number of working days between now and 10
days from now.
```

NEXTMONTH or **@NEXTMONTH**

Returns a date serial number one calendar month after date.

NEXTMONTH(*date*)

date a date serial number

This function returns the date serial number of the date one month from the given date. If the day of the month is greater than 28, then it is set to 28.

```
A3: 27-Jul-92
=NEXTMONTH(A3) -> 8274 : This is the
date-number for
27-Aug-92.

=NEXTMONTH(11381) -> 11409 : This is the
date-number for 28-Mar-2001.
```

NEXTYEAR or **@NEXTYEAR**

Returns the next year of the date.

NEXTYEAR(*date*)

date a date serial number

This function returns the date serial number of the date one year from the given date. If the date is 2/29 of a leap year, the date returned is 2/28 of the following year.

```
=NEXTYEAR(NOW) : This formula returns a
date-number one year from the current date.
=NEXTYEAR(DATE(93,6,20)) -> 8936 : This is
the date-number for 20-Jun-94.
```

SECOND or **@SECOND**

Returns the seconds of a date serial number.

SECOND(*date*)

date a date serial number

SECOND returns the seconds from a date serial number *date*. The date can also be a reference to a cell containing a date. If a time is specified in a cell, but no date is specified, the integer portion of the date serial number is assumed to be zero.

```
=SECOND(TIMEVALUE("10:31:19")) -> 19
=SECOND(0.26903) -> 24 : This formula returns
the seconds for the date-value for 06:27:24
AM.
```

TIME or **@TIME**

Returns a date serial number for a given time.

TIME(*hr, min, sec*)

hr the hour, from 0 to 23

min the minutes

sec the seconds

This function takes the time, for a given 24 hour time into the corresponding date serial number used by Mesa’s date functions.

```
=TIME(11,30,00) -> 0.479167 : This is the
date-number for 11:30 AM.
=NOW+TIME(8,0,0) : This formula returns a
date-number for the time 8 hours from now.
```

TIMEVALUE or **@TIMEVALUE**

Converts a string to a date serial number.

TIMEVALUE(*time-str*)

time-str a string corresponding to a time

This function converts time entered as string into the date serial number used by Mesa’s date functions.

```
A1: '10
A2: '58
A3: '29
=TIMEVALUE(A1&":"&A2&":"&A3) = 0.45728 : This
formula takes the string contents of cells
A1, A2 & A3 and concatenates them into the
string "10:58:29", then converts that string
```

```
to a time value.

A3: 12:34 am
=TIMEVALUE(A3) -> 0.023611 : This is the
date-number for 12:34 am.
```

WEEKDAY or **@WEEKDAY**

Returns the day of the week for a date serial number, 0 - Sunday through 6 - Saturday.

WEEKDAY(*date*)

date a date serial number

This function is useful in determining what day of the week a given date serial number is.

```
=WEEKDAY(NOW-5) : This returns the day of the
week 5 days prior to today.

A7: "6/20/82"
=WEEKDAY(DATEVALUE(A7)) -> 0 : Returns the
day of the week for 20-Jun-82.
```

WORKDAY or **@WORKDAY**

Returns the date serial number a given number of working days from the `date` serial number. Working days are all days other than Saturday, Sunday, and the optional `holidays` listed.

WORKDAY(*date, days [,holidays]*)

date a date serial number

days any number of days

holidays the serial number of any intermediate holidays

This function takes a date serial number and determines the date serial number of the day a given number of working days from then. If there are any holidays listed, they will not be counted as working days by this function. You may list any combination of dates, cell addresses containing dates, and ranges containing dates as arguments for *holidays*.

```
=WORKDAY(NOW,10) : This returns the date
serial number 10 working days from now.
```

YEAR or **@YEAR**

Returns the year for a date serial number.

YEAR(*date*)

date a date serial number

YEAR returns the year for a date serial number. The date can also be a reference to a cell containing a date.

```
=YEAR(NOW-1000) : This returns the year of
the date 1,000 days prior to the current
date.
```

```
A7: '6/20/82
=YEAR(DATEVALUE(A7)) -> 82 : This formula
extract the year from the date-string
"6/20/82".
```

Financial Functions

CALCRATE or **@CALCRATE**

Calculates the interest rate on an investment, given its present value, the amount of each payment, and the number of payment periods.

CALCRATE(*pv*, *pmt*, *per*)

pv the present value of the investment

pmt the amount of each payment

per the number of payment periods

Given an investment with a present value of *pv*, payments of *pmt*, and *per* payment periods, the CALCRATE function returns the interest rate such that *pv* = @PV(*pmt*, *rate*, *term*). See “@PV” *101*.

```
Find the monthly interest rate of a 36 month
car loan on $17,000 with a monthly payment of
$500:
=CALCRATE(17000,500,36) = 0.31% per month or
3.75% per year.
```

@CTERM

Computes the number of periods required for an investment to grow to the specified Future Value with a given interest rate.

@CTERM(*rate*, *fv*, *pv*)

rate the interest rate

fv the desired Future Value of the investment

pv the Present Value of the investment

Given an investment with a Present Value of *pv*, the @CTERM function returns the number of periods required for it to grow to a Future Value of *fv* at an interest rate of *rate*.

```
How long will it take for a $20,000
investment to grow to $40,000 at an annual
rate of 5%?
@CTERM(0.05,20000,40000) -> 14.2 years
```

```
How many months will it take for a $4,000
investment to grow to $8,000, being
compounded monthly at a rate of 8%?
@CTERM(8%/12,4000,8000) -> 104 months (about
8 1/2 years)
```

DDB or **@DDB**

Computes the Double Declining Balance depreciation for an asset.

DDB(*cost*, *salvage*, *life*, *per*)

cost the original cost of the asset

salvage the salvage value of the asset

life the expected life of the asset

per the number or periods for the depreciation calculation

Given an asset that has a cost of *cost*, a salvage value of *salvage*, and a life of *life* periods, the DDB function will return the depreciated value of the asset after *per* periods.

```
A company buys a delivery van for $12,000. It
has a useful life of 5 years and will be
worth $2,000 at the end of that time. How
much depreciation can the company write off
in years 1,2 and 3?

    year 1:  =DDB(12000,2000,5,1) -> $4,800

    year 2:  =DDB(12000,2000,5,2) -> $2,880

    year 3:  =DDB(12000,2000,5,3) -> $1,728
```

FV or **@FV**

Computes the Future Value of a series of payments, given a specified interest rate. An optional present value and starting time for the first payment may be specified.

FV(*rate*, *per*, *pmt*[, *pv*[, *type*]])

rate the interest rate

per the number of payments

pmt the size of each payment

pv the Present Value

type the starting time of the first payment

Returns the Future Value of a stream of payments over *per* periods at *rate* interest rate. If present value *pv* is not specified, it is assumed to be 0. *Type* designates when the payments start.

```
If $2,000 were deposited every year into an
account earning 9% annual interest, how much
would the account be worth in 20 years?
       =FV(9%,20,-2000) -> $102,320.24
If a $50,000 were deposited instead and
accrued 9% interest from the start of the
year, what would the account be worth in 20
years?
       =FV(9%,20,,-50000,1) -> $ 280,220.54
```

IRR

Calculates the Internal Rate of Return of an investment.

IRR(*guess*, *num*1, *num2, ...*), IRR(*guess, range1, ...*)

guess any number

num any number

range any worksheet range

IRR() calculates the internal rate of return for a series of cash flow values. Each value represents either a payout (a negative value) or a payment received (positive value). The internal rate of return is the interest rate used to discount future cash flows such that the net present value of all the cash flows is zero. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
A rug-cleaning business is being offered for
sale for $180,000.  The owner says the
business should net $25,000, $35,000,
$39,000, $40,000 and $55,000 in years 1-5.
What is this project's internal rate of
return?
         A
1     -180000
2       25000
3       35000
4       39000
5       40000
```

```
6       55000
```

```
The initial expense is entered as a negative
number in cell A1.  The range of cash inflows
are given in cells A2:A6. To calculate this
project's IRR value:

       =IRR(B1,A1:A6) -> 2.3%
```

@IRR

Computes the Internal Rate of Return of an investment. String entries are considered zero values.

@IRR(*guess*, *num*1, *num2, ...*), @IRR(*guess, range1, ...*)

guess any number

num any number

range any range

@IRR() calculates the internal rate of return for a series of cash flow values. Each value represents either a payout (a negative value) or a payment received (positive value). The internal rate of return is the interest rate used to discount future cash flows such that the net present value of all the cash flows is zero. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
A rug-cleaning business is being offered for
sale for $180,000. The owner says the
business should net $25,000, $35,000,
$39,000, $40,000 and $55,000 in years 1-5.
What is this project's internal rate of
return?
         A
1     -180000
2       25000
3       35000
4       39000
5       40000
6       55000

The initial expense is entered as a negative
number in cell A1. The range of cash inflows
are given in cells A2:A6. To calculate this
project's IRR value:

@IRR(B1,A1:A6) -> 2.3%
```

NPV

Computes the Net Present Value of an investment with a given stream of cash flows at a given interest rate. String values are ignored.

NPV(*rate*, *pmt1*[, *pmt2*[, ...]])

rate the interest rate

pmt a number or range - the payment amounts

Given an interest rate of *rate* and payments *pmt1*, *pmt2*, etc., the NPV function returns the Net Present Value of the stream of payments. Each payment can be a value or a range. If it is a range, each of the numbers in the range is included in the calculation. You may list any combination of numbers, cell addresses, and ranges as arguments for *pmt*.

```
An apartment building is being offered for
sale for $400,000. Expected rental incomes
for years 1-5 are $79,000, $85,000, $85,000,
$103,000 and $120,000 and the interest rate
is expected to stay close to 7%. Is this a
worthwhile investment?

          A
1      -400000
2        79000
3        85000
4        85000
5       103000
6       120000

To figure out the net present value of the
cash in-flows:
       =NPV(7%,A2:A6) -> $381,596

To figure out the value of the project:
     Project Value = Initial Cost + NPV
          +A1+NPV(7%,A2:A6) -> -$18,404
No, this project is not worthwhile.
```

@NPV

Computes the Net Present Value of an investment with a given stream of income at a given interest rate. String values are treated as zeros.

NPV(*rate*, *pmt1*[, *pmt2*[, ...]])

rate the interest rate

pmt a number or a range - the payment amounts

Given an interest rate of *rate* and payments *pmt1*, *pmt2*, etc., the NPV function returns the Net Present Value of the stream of payments. Each payment can be a value or a range. If it is a range, each of the numbers in the range is included in the calculation. You may list any combination of numbers, cell addresses, and ranges as arguments for *pmt*.

```
An apartment building is being offered for
sale for $400,000. Expected rental incomes
for years 1-5 are $79,000, $85,000, $85,000,
$103,000 and $120,000 and the interest rate
is expected to stay close to 7%. Is this a
worthwhile investment?

           A
1      -400000
2        79000
3        85000
4        85000
5       103000
6       120000

To figure out the net present value of the
cash in-flows:
       @NPV(7%,A2:A6) -> $381,595

To figure out the value of the project:
     Project Value = Initial Cost + NPV
          +A1+@NPV(7%,A2:A6) -> -18,404
No, this project is not worthwhile.
```

PMT

Calculates the size of payments required to pay off a loan.
PMT(*rate*, *per*, *loan*[, *fv*[, *type*]])

loan	the amount borrowed
rate	the interest rate
per	the number of payment periods
fv	the [optional] Future Value of the loan
type	the [optional starting time of the first payment

Given a loan of *loan*, an interest rate of *rate*, and *per* payment periods, the @PMT function returns the size of the payments. If the Future Value *fv* is not specified, it is assumed to be 0. If *type* is not specified, payments are assumed to start immediately.

```
What would be the monthly payment on an
$40,000 home improvement loan, to be repaid
in 5 years at a fixed rate of 9%?
        =PMT(9%/12,5*12,-40000) -> $830.33
per month.
```

@PMT

Calculates the size of payments required to pay off a loan.

@PMT(*prin*, *rate*, *per*)

loan the amount borrowed

rate the interest rate

per the number of payment periods

Given a loan of *loan*, an interest rate of *rate*, and *per* payment periods, the @PMT function returns the size of the payments required.

```
What would be the monthly payment on an
$40,000 home improvement loan, to be repaid
in 5 years at a fixed rate of 9%?

@PMT(40000,9%/12,5*12) -> $830.33 per month.
```

PV

Computes the Present Value of an investment.

PV(*rate*, *per*, *pmt*[, *fv*[, *type*]])

rate the interest rate

per the number of payment periods

pmt the amount of each payment

fv the [optional] Future Value of the investment

type the [optional] starting time of the payments

Given an interest rate of *rate*, *per* payment periods, and a periodic payment of *pmt*, the PV function returns the Present Value of the investment. If a future value *fv* is not given, it is assumed to be 0. If *type* is not zero, payments are assumed to start in the next period.

```
What is the present value of the retirement
annuity which promises to pay $30,000 every
year for the next 20 years, given an interest
rate of 12%?

        =PV(12%,20,-30000) -> $224,083.31

What is the present value of the investment
which promises to pay $225,000 at the end of
20 years, given an interest rate of 12%?

        =PV(12%,20,-225000) -> $23,325.02
```

@PV

Computes the Present Value of an investment.

@PV(*pmt*, *rate*, *per*)

pmt the amount of each payment

rate the interest rate

per the number of payment periods

Given *per* payments of size *pmt* and an interest rate of *rate*, the @PV function returns the Present Value of the investment.

```
What is the present value of the retirement
annuity which promises to pay $30,000 every
year for the next 20 years, given an interest
rate of 12%?

@PV(30000,12%,20) -> $224,083.31
```

@RATE

Returns the interest rate for an investment, given the Future Value, Present Value, and number of payment periods.

@RATE(*fv*, *pv*, *per*)

fv the desired Future Value of the investment

pv the Present Value of the investment

per the number of payment periods

Given a Present Value of *pv*, the @RATE function returns the required interest rate necessary to achieve the desired Future Value of *fv* in *per* payment periods.

```
How much interest would a bank account need
to earn for an investment of $67,000 to grow
to $100,000 in 5 years or less?

@RATE(100000,67000,5) -> 8.34% or more.
```

SLN or **@SLN**

Returns the Straight-Line depreciation of an asset.

SLN(*cost*, *slvg*, *life*)

cost the original cost of the asset

slvg the salvage value of the asset

life the expected lifetime of the asset

Given an asset with an original cost of *cost*, a salvage value of *slvg*, and a lifetime of *life*, the SLN function returns the Straight-Line depreciation for the asset.

```
A company buys a delivery van for $12,000. It
has a useful life of 5 years and will be
worth $2,000 at the end of that time. How
much depreciation can the company write off
each year?

        =SLN(12000,2000,5) -> $2,000
```

SYD or **@SYD**

Calculates the Sum of the Years Digits depreciation for an asset.

SYD(*cost*, *slvg*, *life*, *per*)

cost the original cost of the asset

slvg the salvage value of the asset

life the expected lifetime of the asset

per the number of periods for the depreciation calculation

Given an asset with an original cost of *cost*, a salvage value of slvg, and a lifetime of *life* periods, the SYD function returns the Sum of the Years Digits depreciation for *per* periods.

```
A company buys a delivery van for $12,000. It
has a useful life of 5 years and will be
worth $2,000 at the end of that time. How
much depreciation can the company write off
in years 1,2 and 4?

year 1:  =SYD(12000,2000,5,1) -> $3,333

year 2:  =SYD(12000,2000,5,2) -> $2,667

year 4:  =SYD(12000,2000,5,4) -> $1,333
```

@TERM

Computes the number of payments required to reach a desired Future Value with a given payment size and interest rate.

@TERM(*pmt*,*rate*,*fv*)

pmt	the size of each payment
rate	the interest rate
fv	the desired Future Value

Given payments of size *pmt* and an interest rate of *rate*, the @TERM function returns the number of payments required to reach the desired Future Value *fv.*

```
How many $700 monthly payments would be
needed to accumulate $250,000 in a bank
account earning 4% interest?

@TERM(700,4%/12,250000) -> 235.6 monthly
payments (taking almost 20 years).
```

Math Functions

ABS or **@ABS**
Computes the absolute value of the given number.

ABS(*num*)

num   any number

The absolute value of a number is the value of the number ignoring positive or negative indicators. The absolute value of a positive number is given as the number. The absolute value of a negative number would be that number multiplied by negative one (-1); in other words the same number, except positive.

```
=ABS(-45.3)           -> 45.3
@ABS(-45.3)           -> 45.3
=ABS(COS(PI))         -> 1
=ABS(34)              -> 34
=ABS(``Absolutely!'') -> 0
```

ACOS or **@ACOS**
Returns the arccosine of the given number.

ACOS(*num*)

num   any number between -1 and 1

The arccosine of the number *num* is the angle whose cosine is *num*. The ACOS function gives the value of the angle in radians; the value of the angle can be obtained in degrees by using this function in conjunction with the RADTODEG function. See "RADTODEG or @RADTODEG"116.

```
=ACOS(0.7071) -> 0.7854 (radians) : This is
equal to an angle of 45 degrees.

@RADTODEG(@ACOS(0.7071)) -> 45 : This gives
the same answer as above, but in degrees.
```

ASIN or **@ASIN**
Returns the arcsine of the given number.

ASIN(*num*)

num   any number between -1 and 1

The arcsine of the number *num* is the angle whose sine is *num*. The ASIN function gives the value of the angle in radians; the value of the angle can be obtained in degrees by using this function in conjunction with the RADTODEG function. See "RADTODEG or @RADTODEG"116.

```
=ASIN(0.3090) -> 0.3141 (radians) : This is
equal to an angle of 18 degrees.
```

```
@RADTODEG(@ASIN(0.3090)) -> 18 : This gives
the same answer as above, but in degrees.
```

ATAN or **@ATAN**

Returns the arctangent of the given number.

ATAN(*num*)

num any number

The arctangent of the number *num* is the angle whose tangent is *num*. The ATAN function gives the value of the angle in radians; the value of the angle can be obtained in degrees by using this function in conjunction with the RADTODEG function. See "RADTODEG or @RADTODEG"116.

```
=ATAN(5.6713) -> 1.3963 (radians) : This is
equal to an angle of 80 degrees.
```

ATAN2

Returns the arctangent of the ratio of the given X and Y coordinates.

ATAN2(*x, y*)

x any number

y any number

The ATAN2 function finds the angle between the X axis and the point with the given coordinates *x* and *y*. This gives the angle of a line from the origin (0,0) to the point (*x*,*y*). The ATAN2 function gives the value of the angle in radians; the value of the angle can be obtained in degrees by using this function in conjunction with the RADTODEG function. See "RADTODEG or @RADTODEG"116.

```
=ATAN2(1,3) -> 0.3218 (radians) : This is
equal to an angle of 18.4 degrees.
```

@ATAN2

Returns the arctangent of the ratio of the given Y and X coordinates.

@ATAN2(*y, x*)

y any number

x any number

@ATAN2 finds the angle between the X axis and the point with the given coordinates *x* and *y*. This gives the angle of a line from the origin (0,0) to the point (*x*,*y*). The @ATAN2 function gives the value of the angle in radians; the value of the angle can be obtained in degrees by using this function in conjunction with the RADTODEG function. See “RADTODEG or @RADTODEG”116.

```
@ATAN2(3,1) -> 0.3217 (radians): This is
equivalent to an angle of 18.4 degrees.
```

ATAN2 and @ATAN2 are different functions. ATAN2 requires the x parameter to be first, whereas @ATAN@ requires the y parameter to be first.

AVE or **AVERAGE**

Computes the average, or mean, of a list of numeric values. Strings are ignored.

AVE(*num*, ...), AVE(*range*), AVE(*range1, range2, range3,...*)

num any number

range any worksheet range

Divides the sum of the given numbers by the count of numeric entries. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. String entries are ignored and have no effect on the result.

```
=AVERAGE(4,5,25,3) -> 9.25

Given the following data:
A1: 184
```

```
A2: 592
A3: 97
=AVE(A1:A3) -> 291
```

@AVE or @AVG

Computes the average, or mean, of a list of numeric values. Strings are treated as zero values.

@AVE(*num*,...), @AVE(*range*), @AVE(*range1, range2, range3,...*)

num any number

range any worksheet range

Divides the sum of the given numbers by the count of numeric entries. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. String entries are treated as zero and are computed into the result.

```
@AVE(4,5,25,3) = 9.25

Given the following data:
A1: 184
A2: 592
A3: 97
@AVE(A1:A3) = 291
```

AVE and AVERAGE are different from @AVE and @AVG in that AVE and AVERAGE ignore string values, while @AVE and @AVG treat string values as having a zero value.

COS or @COS

Returns the cosine of the given number.

COS(*num*)

num any number

The cosine of the angle *num* is the X coordinate of the point where a line at that angle from the X axis intersects a circle of radius one (1) centered on the origin (0,0). The COS function takes the value of the angle in radians; the value of the angle can be converted from degrees to radians within this function by using it in conjunction with the DEGTORAD function. See “DEGTORAD or @DEGTORAD”110.

```
=COS(PI) -> -1
@COS(LN(3)) -> 0.4548
```

COUNT

Counts the number of numeric entries. Cells or items containing blanks or strings are not counted.

COUNT(*num , ...*), COUNT(*range*),
COUNT(*range1, range2, ...*)

num any number

range any worksheet range

The COUNT function finds the number of numeric entries in the list of arguments, and ignores string values. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
         A       B
1      Duck      10
2      Soup      20
3         1.86   20
4         $300   10
5      '======   30
6                50
7           40   70
```

```
=COUNT(A1:A7) -> 3 : There are only three
items in column A because cells A1, A2, and
A5 are strings and A6 is blank.

=COUNT(A1:A7,B1:B7) -> 10 : This counts the 3
items in column A and the 7 items in column
B.

=COUNT("Good","Bad","Indifferent",0) -> 1
```

@COUNT

Counts the number of numbers and strings given.

@COUNT(*num, ...*), @COUNT(*range*),
@COUNT(*range1, range2, ...*)

num any number

range any worksheet range

The @COUNT function finds the number of numeric or string entries in the list of arguments. Cells or items containing strings are counted. Cells or items containing blanks are not counted. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
         A       B
1      Duck      10
2      Soup      20
3         1.86   20
4         $300   10
5      '======   30
6                50
7           40   70
```

```
@COUNT(A1:A7) = 6: There are only six items
in column A because cell A6 is blank.
```

```
@COUNT(A1:A7,B1:B7) = 13: This counts the 6
items in column A and the 7 items in column
B.
```

```
@COUNT(``Good'',"Bad","Indifferent") = 3
```

COUNT and @COUNT are different in that COUNT only counts numeric values. @COUNT also counts string values.

DEGTORAD or **@DEGTORAD**

Converts the given number from degrees to radians.

DEGTORAD(*num*)

num any number

Angles can be expressed in units of degrees or radians. There are 360 degrees, or 2π radians in a circle. All of Mesa's trigonometric functions express angles in radians. This function will allow you to convert angles given in degrees to radians for use in Mesa's trigonometric functions. Radians can be converted to degrees using the RADTODEG function. See "RADTODEG or @RADTODEG" *116*.

```
=DEGTORAD(180) -> 3.14159
```

DIV or **@DIV**

Divides the given number by the given divisor and returns an integer.

DIV(*num, div*)

num any number
div any non-zero number

This function divided the given number by the given divisor and returns the integer portion of the answer. The remainder can be obtained using the MOD function. See “MOD or @MOD” *115*.

```
=DIV(12,5) -> 2
```

EXP or **@EXP**

Raises e the power of the given number.

EXP(*num*)

num any real number

EXP raises e to the given power. The number e is defined to be the number such that the derivative of e^{num} is e^{num}. The number e is approximately 2.71828. The natural logarithm of e^{num} is *num.* See “LN or @LN”*112*.

```
=EXP(4) -> 54.5982

@EXP(12%*5) * 30000 -> $54,664  :  This
formula calculates the present value of
$30,000 invested at 12% continually
compounded interest for a term of 5 years.
```

FRAC or **@FRAC**

Returns the fractional component of the given number.
FRAC(*num*)

num any number

FRAC returns the non-integer portion of the given number, *num*. This function complements the INT function, see “INT or @INT”*111*.

```
=FRAC(NOW)  :  This formula returns only the
time component of the current date.

@FRAC(3.14159) -> 0.14159
```

INT or **@INT**

Returns the integer portion of the given number.

INT(*num*)

num   any number

INT returns the integer portion of the given number, *num*. This does not round the number off, it simply ignores all digits to the right of the decimal point. This function complements the FRAC function, see “FRAC or @FRAC”111.

```
=INT(RAND*10)    :  This formula returns a
random number between 0 and 9.

A3: 68293
@INT(LOG(A3))+1 -> 5  :  This formula tells
how many digits are the number in cell A3.
```

LN or @LN

Returns the natural logarithm of the given number.

LN(*num*)

num   any positive number

The natural logarithm of a number is the power to which e needs to be raised to result in the given number; in other words, the logarithm to base e. The number e is defined to be the number such that the derivative of e^{num} is e^{num}. The number e is approximately 2.71828. If the natural logarithm of *num* is raised to the power e, the result is *num.* See “EXP or @EXP”111.

```
=LN(27) -> 3.2958

B8: 1024
=LN(B8)/LN(2) -> 10  :  This formula returns
the base 2 log of cell B8.
```

LOG, LOG10 or @LOG

Returns the base ten (10) logarithm of given number.

LOG(*num*)

num   any positive number

The base ten (10) logarithm of a number is the power to which ten needs to be raised in order to result in the given number.

```
=LOG10(45) -> 1.6532
```

```
+10^(LOG(292)) -> 292  :  This formula takes
the base 10 log of 292, then raises 10 to
that power, resulting in the number it began
with.
```

MAX

Returns the largest number in the given range or list.

MAX(*num,...*), MAX(*range*), MAX(*range1, range2,...*)

num any number

range any worksheet range

MAX returns the largest number in a given list or range, ignoring strings and non-numeric input. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
=MAX(4.5,3.2,2.5,2.5,6.2) -> 6.2

=MAX(4,4,4) -> 4

         A     B     C
1       300         400
2         0         400
3       200         100
4       300        5000
5       700
6       600
7       300
8       500
=MAX(A1:A8,C1:C8) -> 5000
```

@MAX

Returns the largest number in the given range or list with strings considered to be zero.

@MAX(*num,...*), @MAX(*range*), @MAX(*range1, range2,...*)

num any number

range any worksheet range

@MAX returns the largest number in a given list or range. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Strings are treated as having a value of zero.

```
@MAX(4.5,3.2,2.5,2.5,6.2) = 6.2

@MAX(4,4,4) = 4

         A     B     C
1        300          400
2          0          400
3        200          100
4        300         5000
5        700
6        600
7        300
8        500
@MAX(A1:A8,C1:C8) -> 5000
```

MAX() and @MAX() differ in that MAX() ignores string values, while @MAX treats string values as zero.

MIN

Results in the smallest number in the given range or list.

MIN(*num,...*), MIN(*range*), MIN(*range1, range2,...*)

num any number

range any worksheet range

MIN returns the smallest number in a given list or range, ignoring strings and non-numeric input. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
A1: 300, A2: 400, A3: Hello!
=MIN(A1:A3) -> 300  :  This formula returns
300, and the string entry in cell A3 is
ignored.

A1: 0.0002, A2: 0, A3: -339492
+MIN(A1:A3) -> -339492
```

@MIN

Returns the smallest number in the given range or list, with string values considered to be zero.

@MIN(*num,...*), @MIN(*range*), @MIN(*range1, range2,...*)

num any number

range any worksheet range

@MIN returns the smallest number in a given list or range. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Strings are treated as having a value of zero.

```
A1: 300
A2: 400
A3: Hello!
@MIN(A1:A3)-> 0 :  This formula returns zero
because the string "Hello!" evaluates to
zero.
```

```
A1: 0.0002
A2: 0
A3: -339492
@MIN(A1:A3) -> -339492
MIN() and @MIN() differ in that MIN() ignores
string vales while @MIN() treats strings as
having a value of zero.
```

MOD or @MOD

Returns the remainder of the given number divided by the given divisor.

MOD(*num, div*)

num any number

div any non-zero number

MOD compliments the DIV function (see “DIV or @DIV”*110*) by returning the portion of the given number that does can not be divided evenly by the given divisor.

```
=MOD(12,5) -> 2
```

```
@NOW-@MOD(@NOW,7)+5  :  This function returns
the date-number for Monday of the current
week.
```

PROD

Multiplies all the numbers in the given list.

PROD(*num, ...*), PROD(*range*), PROD(*range1, range2, ...*)

num any number

range any worksheet range

PROD takes the product of all the numbers in the given list. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Strings within the list are ignored.

```
=PROD(3, 5, 10, 2) -> 300

A1:  3
A2:  5
B1: 10
=PROD(A1:A2, B1, 2) -> 300
```

@PROD

Multiplies all the numbers in the given list. Strings are treated as having a value of zero.

PROD(*num, ...*), PROD(*range*), PROD(*range1, range2, ...*)

num any number

range any worksheet range

PROD takes the product of all the numbers in the given list. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Strings within the list are treated as having a value of zero, so if a string appears in the list, PROD will return zero.

```
@PROD(3, 5, 10, 2) -> 300

A1:  3
A2:  5
B1: 10
@PROD(A1:A2, B1, 2) -> 300
```

PROD and @PROD differ in that PROD ignores strings, while @PROD treats strings as having a value of zero.

RADTODEG or **@RADTODEG**

Converts the given number from radians to degrees.

RADTODEG(*num*)

num any number

Angles can be expressed in units of degrees or radian. There are 360 degrees, or 2π radians in a circle. All of Mesa's trigonometric functions express angles in radians. This function will allow you to convert angles given in radians by Mesa's trigonometric functions to degrees. Degrees can be converted to radians using DEGTORAD. See "DEGTORAD or @DEGTORAD"*110*.

```
=RADTODEG(PI) -> 180
```

ROOT or **@ROOT**

Returns the given root of the given number.

ROOT(*num, n*)

num any positive number

n any number

This function can be used to find the *n*th root of a number, *num*. The *n*th root of a number *num* is the number that, multiplied by itself *n* times, returns the number *num*.

```
=ROOT(81,4) -> 3
```

ROUND or **@ROUND**

Rounds a number to a given number of decimal places.

ROUND(*num, prec*)

num any number

prec any integer

This function rounds the number *num* to *prec* decimal places. If *prec* is zero, the number is rounded to the nearest integer. If *prec* is greater that zero, it will round off *prec* places beyond the decimal point. If *prec* is less than zero, it rounds to the left of the decimal place.

```
=ROUND(350.2852,2) -> 350.29
@ROUND(25492,-3) -> 25000
```

SGN, SIGN, @SIGN, or **@SGN**

Determines whether a number is positive or negative.

SGN(*num*)

num any number

This function returns 0 if *num* is zero, -1 if *num* is less than 0, and 1 if *num* is greater than 0. This functions is useful in conjunction with other functions where it is necessary to determine whether a number is positive, negative, or zero.

```
B7: +RAND
=SIGN(B7-0.5) : This formula returns -1, 0,
or 1 randomly.

=SGN(-405) -> -1
```

SIN or **@SIN**

Computes the sine of the given number.

SIN(*num*)

num any number

The sine of an angle is the Y coordinate of the point where a line at that angle from the X axis intersects a circle of radius one (1) centered on the origin. The SIN function takes the value of the angle in radians; the value of the angle can be converted from degrees to radians within this function by using it in conjunction with the DEGTORAD function. See “DEGTORAD or @DEGTORAD”*110*.

```
=SIN(60*PI/180) = 0.8660 : This formula gives
the sine of 60 degrees.

A mountain road goes up at an incline of 25
degrees.  If the road is straight and is 20
miles long, what vertical distance will a car
travel to climb it? The formula is +20 *
SIN(25*(pi/180)) -> 8.45 miles
```

SQRT or **@SQRT**

Calculates the square root of a number.

SQRT(*num*)

num any positive number

This function finds the positive number which, when multiplied by itself, yields the number *num*. To find roots other than the square root, see "ROOT or @ROOT"117.

```
=SQRT(34) -> 5.8310
@SQRT(@LOG(200)) -> 1.5169
=SQRT(VAR(200,500,100)) -> 170 : This formula
computes the standard deviation of the values
given in the variance function.
```

STDDEV or **STDEV**

Returns the standard deviation of the ranges or list. Blank cells and strings are not counted.

STDDEV(*num,...*), STDDEV(*range*),
STDDEV(*range1, range2,...*)

num any number

range any worksheet range.

This function finds the standard deviation of the given range or list of numbers. The items in the list are separated by commas, and they may be numbers or cell or range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. The standard deviation is a measure of the statistical distribution of a set of numbers.

	A	B
1	NAME	SCORE
2	Anna	65.8
3	Bill	95.4
4	Donna	30.2
5	Mark	54.9
6	Maria	35.1
7	Susan	75.9
8	John	83.2
9	Rob	33.1
10	Ethan	81.8

```
=STDEV(B2:B10) -> 23.1
```

@STD

Returns the standard deviation of the ranges or list. Strings are counted as 0, and blank cells are not counted.

@STD(*num,...*), @STD(*range*), @STD(*range1, range2,...*)

num any number

range   any worksheet range

This function finds the standard deviation of the given range or list of numbers. The items in the list are separated by commas, and they may be numbers or cell or range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Strings within the list are treated as having a value of zero. The standard deviation is a measure of the statistical distribution of a set of numbers.

```
         A      B
1       NAME   SCORE
2       Anna    65.8
3       Bill    95.4
4       Donna   30.2
5       Mark    54.9
6       Maria   35.1
7       Susan   75.9
8       John    83.2
9       Rob     33.1
10      Ethan   81.8

@STD(B2:B10) -> 23.1
```

STDDEV() and STDEV() differ from @STD() in that STDDEV() and STDEV() ignore string values while @STD() treats string values as zero.

SUM or **@SUM**

Adds all the numbers in a list.

SUM(*num,...*), SUM(*range*), SUM(*range1, range2,...*)

num   any number

range   any worksheet range

This function finds the sum of all the numbers in the given list or range. The items in the list are separated by commas, and they may be numbers or cell or range references containing numbers. You can specify as many numbers or ranges as you want. You may list any combination of numbers, cell addresses, and ranges as arguments.

```
=SUM(300,400,2300,100) -> 3100

A1:  Blue
A2:  300
A4:  900
B1:  1200
```

```
=SUM(A1:A4,B1) -> 2400
```

TAN or **@TAN**

Returns the tangent of the given number.

TAN(*num*)

num any number

The tangent of an angle is ratio of the Y coordinate to the X coordinate of the point where a line at that angle from the X axis intersects a circle of radius one (1) centered on the origin. The TAN function takes the value of the angle in radians; the value of the angle can be converted from degrees to radians within this function by using it in conjunction with the DEGTORAD function. See "DEGTORAD or @DEGTORAD"*110*.

```
=TAN(1.24) -> 2.9119
```

VAR

Calculates the statistical variance of a list of numbers. Strings and blank cells are not counted.

VAR(*num,...*), VAR(*range*), VAR(*range1, range2,...*)

num any number

range any worksheet range

This function finds the statistical variance of a range of list of numbers. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Variance is a measure of how much a set of numbers vary from each other. Strings and blank cells in the list are ignored.

	A	B
1	NAME	SCORE
2	Anna	65.8
3	Bill	95.4
4	Donna	30.2
5	Mark	54.9
6	Marie	35.1
7	Susan	75.9
8	John	83.2
9	Rob	33.1
10	Ethan	81.8

```
=VAR(B2:B10) -> 531.6010
```

@VAR

Calculates the statistical variance of a list of numbers. Strings are counted as 0, and blank cells are not counted.

@VAR(*num,...*), @VAR(*range*), @VAR(*range1, range2,...*)

num any number

range any worksheet range

This function finds the statistical variance of a range of list of numbers. The items in the list are separated by commas, and they may be numbers or cell/range references containing numbers. You may list any combination of numbers, cell addresses, and ranges as arguments. Variance is a measure of how much a set of numbers vary from each other. This function is essentially the same as VAR, except that it treats strings as having a value of zero.

	A	B
1	NAME	SCORE
2	Anna	65.8
3	Bill	95.4
4	Donna	30.2
5	Mark	54.9
6	Marie	35.1
7	Susan	75.9
8	John	83.2
9	Rob	33.1
10	Ethan	81.8

```
@VAR(B2:B10) = 531.6010
```

VAR and @VAR are different in that VAR ignores strings while @VAR treats string values as zero.

Operators

! or::

Links worksheets

sheet!*adr*

sheet	The name of the worksheet to be linked
adr	The address of a cell or range in *sheet*

These operators create a link to another worksheet, specified by the filename of the worksheet, and the address of a cell or range in that worksheet. A worksheet link will return the value of the cell or range. Linked worksheets need to be in the same directory.

```
=’’Exports’’::B52 -> Returns the contents of
cell B52 in the worksheet named Exports.

=U2!I9 -> Returns the contents of cell I9 in
the worksheet whose name is in cell U2 of
this worksheet

=sum(‘‘Exports’’!A1:B20) -> Returns the sum
of the numbers in the range A1:B20 in the
worksheet named Exports

=sum(U2::prices) -> Returns the sum of the
numbers in the range labeled prices in the
worksheet whose name is in cell U2 of this
worksheet.
```

&

string concatenation

string1&*string2*

This operator concatenates the two strings on either side of it, forming one string.

```
=‘‘this’’&’’that’’ -> thisthat

                           A    B
1                        this that
=A1&B1 -> thisthat
```

= or ==

equal

exp1=exp2

Expressions containing this operator are true if the value of the first expression is equal to the value of the second expression.

!= or <>
not equal to

exp1!=*exp2*

This operator tests to see if one expression is not equal to another.

```
if(1!=0, ‘‘OK’’) -> OK
```

<
less than

num1<num2

Expressions containing this operator are true if the value of the first number is smaller than the value of the second number.

>
greater than

num1>num2

Expressions containing this operator are true if the value of the first number is larger than the value of the second number.

<= or =<
less than or equal to

num1<=num2

Expressions containing this operator are true if the value of the first number is either less than or equal to the value of the second number.

>= or =>
greater than or equal to

num1>=num2

Expressions containing this operator are true if the value of the first number is either greater than or equal to the value of the second number.

+
addition operator

num1+num2

This operator returns the sum of the given numbers.

```
=4+5 -> 9
```

-
subtraction operator

num1-num2

This operator subtracts the second number from the first number.

```
=10-2 -> 8
```

*****
multiplication operator

*num1*num2*

This operators returns the product of the given numbers.

```
=3*5 -> 15
```

/
division operator

num1/num2

This operator divides the first number by the second number.

```
=20/5 -> 4
```

^
exponentiation operator

num1^num2

This operator raises the first number to the power given by the second number. In other words it multiplies the first number by itself a number of times specified by the second number.

#AND#, &&, <AND>, or **AND**

logical AND

exp1#AND#*exp2*

This operator serves as a logical AND. When used to link two expressions, the entire expression is true only if both separate expressions are true.

#OR#, <OR>, OR, or ||

logical OR

exp1#OR#*exp2*

This operator serves as a logical OR. When used to link two expressions, the entire expression is true if either of the two expressions is true.

<XOR>, XOR, or **^^**

exclusive OR

exp1<XOR>*exp2*

Expressions containing this operator are true if either, but not both, or the given expressions are true.

Special Functions

#NOT#

Logical NOT.

#NOT# *arg*

arg a number or true/false value

This is the logical NOT function, but it is not used the same way as the logical AND and OR. For example, *#NOT# 1* will equal 0, while *#NOT# 0* will equal 1.

```
A3 = +("Blue"="Red") ,  which returns 0
#NOT#A3   =  1  :  The formula in  cell A3
asks "is the string 'Blue' = the string 'Red'?
The answer is 0 (false).  The #NOT# function
inverts this returning the answer 1 (true)

#NOT#1 = 0 (false)
```

@@

Evaluates a string as a cell address.

@@(*string*)

string any string that can be interpreted as a cell address

This function evaluates the string *string* as a cell address and returns that address. This can be useful for creating a cell address with string concatenation.

```
A3: Cheers
B3: A3
@@(B3) -> Cheers

A1: 10
A2: 20
A3: 30
B1: A1:A3
@SUM(@@(B1)) -> 60
```

ADDRESS or @ADDRESS

Returns a cell address for a given row and column number.

ADDRESS(*row*,*col*)

row the number of the row

col the number of the column

The ADDRESS() function returns the address for the *row* and *col* specified. Row and column numbering starts at zero, i.e. ADDRESS(0,0) is cell A1. This function can be useful in MScript scripts.

```
B1: 15
=ADDRESS(0,1) -> 15
```

BEEPIF or **@BEEPIF**

Beeps if the specified condition is true.

BEEPIF(*cond*)

cond a conditional expression

The BEEPIF function beeps and returns 1 if the condition *cond* is true, and returns 0 if the condition if false. This can be useful for alerting the user of certain conditions within the worksheet.

```
C7:  100
=BEEPIF(C7<200) :  This formula will beep one
time.

=BEEPIF(rand>0.5) : This formula will generate
a random number between 0 and 1.  If the
number is > 0.5, the it beeps and returns 1.
Otherwise it returns 0.
```

CELL or **@CELL**

Returns information about the attributes of a given cell.

CELL(*atrb*,*cell*)

atrb the cell attribute

cell the cell address

The CELL() function returns attribute information about a the cell at the address *cell*. The type of information requested is specified by the argument *atrb*, chosen from the list below.

```
U7: PI

=CELL("address",U7) -> $U$7
=CELL("row",U7) -> 7
```

```
=CELL("col",U7) -> 21
=CELL("contents",U7) -> 3.141593
```

ADDRESS - returns the address of the cell
ROW - returns the row of the cell
COL - returns the column of the cell
CONTENTS - returns the value of the cell
FORMAT - returns the format of the cell. Formats are:
- G - General
- Fn - Fixed Decimal n places
- Sn - Scientific Format n decimal places
- Cn - Currency format n decimal places
- ,n - Comma format n decimal places
- + - Chart Format
- Pn - Percent Format n decimal places
- T - Text Format
- H - Hidden Format
- D1 - Day Month Year Format
- D2 - Day Month Format
- D3 - Month Year Format
- D6 - Hours Minutes Seconds Format
- D7 - Hours Minutes Format
- D4 - International Date Format 1
- D5 - International Date Format 2
- D8 - International Time Format 1
- D9 - International Time Format 2

PREFIX - The string prefix in Lotus Format
PROTECT - Returns 0 if the cell is Unprotected, 1 otherwise
TYPE - b - blank cell, v - number value, l - string value
WIDTH - Returns the width of the column in characters
RANGE - Converts a single address to a range

CELLPOINTER or **@CELLPOINTER**

Returns information about the attributes of the currently active cell.

CELLPOINTER(*atrb*)

atrb the cell attribute

The CELLPOINTER function returns attribute information about a the currently active cell. The type of information requested is specified by the argument *atrb*. See "CELL or @CELL" 128 for information about attributes. The currently active cell is the cell on the worksheet whose contents is currently being displayed in the formula bar. Mesa worksheets always have one currently active cell. If a range has been selected, the currently active cell is the cell in the upper left corner of the range.

CHOOSE or **@CHOOSE**

Chooses a numbered item from a list.

CHOOSE(*num*, *item1*[, *item2*[, ...])

num the number of the chosen item

item the members of the list

Chooses the *num* item from the list of items *item1*, *item2*, etc. This function counts the listed items starting at zero. You may list any combination of numbers, cell addresses, and ranges as arguments for *item*.

```
=CHOOSE(2,"first","second","third","fourth")
-> third
```

```
=CHOOSE(MOD(INT(NOW),7),"Thu","Fri","Sat",Sun".
: This function returns the current day using
NOW to provide the current date.
```

COLS or **@COLS**

Returns the number of columns in a range.

COLS(*range*)

range any range

The COLS function returns the number of columns contained in the range *range*.

```
=COLS(A1:H20) -> 8
```

```
If RANGE1 is the name assigned to the range
B1:CC200, =COLS(RANGE1) -> 80
```

DIDCHANGE or **@DIDCHANGE**

Determines if the sheet has changed since it was last saved.

DIDCHANGE

DIDCHANGE returns a 1 if the sheet has changed since the last time it was saved, and a 0 if nothing has changed. Since the act of adding this function changes the sheet, it will only show as true when the sheet has been recalculated right after it has been saved. This can be useful in MScript scripts.

FORMAT or **@FORMAT**

Formats the number in the predefined format

FORMAT(*num*, *fmt-const* [,*prec*])

num any number

fmt-const any format constant

prec any integer [optional]

This function returns the given number *num* in the format specified by the format constant *fmt-const*. A list of the format constants can be found in “Format Constants” 73. You may optionally specify the precision *prec*, the number of decimal places in the number.

```
=FORMAT(18.9, SCIENTIFICFORMAT) -> 1.89e+01
```

GETINPUT or **@GETINPUT**

Returns the user-typed input string.

GETINPUT(*cell*)

cell any cell

The GETINPUT function returns the string typed into the cell *cell* by the user, rather than its value. This can be useful for MScript scripts.

```
A1 : =INT(3.2)
=A1 is 3
=GETINPUT(A1) is the string =INT(3.2)
```

HISTOGRAM or **@HISTOGRAM**

Creates a histogram for a given range.

HISTOGRAM(*range*, *divs*, *anchor* [, *min*, *max*])

range any range

divs the number of divisions in the histogram

anchor the starting cell of the histogram

min the [optional] minimum

max the [optional] maximum

The HISTOGRAM function groups the values in *range* into *divs* divisions and counts the number of items in each division. The formula should appear in *divs* cells, starting at *anchor*. It returns the number of items in the division that is offset from the anchor point. *Min* and *max* are optional parameters that set the minimum and maximum values of the histogram range, so that the distribution will only be calculated for values between *min* and *max*.

```
     A      B
1  Name   GPA
2  Joe    2.6
3  Sue    3.7
4  Robert 3.2
5  Steve  3.0
6  Ann    3.8
7  John   3.5
8  Mary   2.9
9  Chris  3.5

To divide the students into 5 divisions by GPA
and determine how many fall into each
division:
A10: =HISTOGRAM($B$2:$B$9,5,$A$10) -> 1
B10: =SAME($A$10) -> 2
C10: =SAME($A$10) -> 1
D10: =SAME($A$10) -> 2
E10: =SAME($A$10) -> 2
```

HLOOKUP or **@HLOOKUP**

Looks up a code in a horizontal table.

HLOOKUP(*code*, *table*, *offset*)

code the code to be looked up

table a range containing the lookup data

offset the number of rows offset from the top of the table.

Returns the cell *offset* cells from the top of the first match. If the first row contains numbers and *code* is less than the first value in the lookup row, !ERR is returned. If *code* is greater than the first value but no exact match is found, HLOOKUP returns the value from the last column. If the first row contains only strings and no exact match is found, it returns !ERR.

```
      A    B    C
1     1    2    3
2  $200  $40 $500

=HLOOKUP(2,A1:C2,1) -> $40
=HLOOKUP(0.5,A1:C2,1) -> !ERR
=HLOOKUP(5,A1:C2,1) -> $500
```

IF or **@IF**

Returns a conditional result.

IF(*exp*,*value1*[,*value2*])

exp the conditional expression for evaluation

value1 the first possible result

value2 the [optional] second possible result

The IF function evaluates *exp*. If it is a true or non-zero value, the result is *value1*. If it is false, zero, or a string, optional *value2* or an empty value is the result.

```
=IF(5>10,"Something is wrong!","Everything is
normal.")
-> Everything is normal.

A1: 3400
A2: 4000
A3: 5000
=IF(SUM(A1:A3)>=12400,"Full Capacity") -> Full
Capacity
```

IFELSE

Returns a conditional result.

IFELSE(*exp*, *value1*, *value2*)

exp the conditional expression for evaluation

value1 the first possible result

value2 the second possible result

The IFELSE() function evaluates *exp*. If it is a true or non-zero value, the result is *value1*. If it is false, zero, or a string, *value2* is the result.

```
=IFELSE(5>10,"Something is wrong!","Everything
```

```
is normal.") -> Everything is normal.

A1: 3400
A2: 4000
A3: 5000
=IFELSE(SUM(A1:A3)>40000,"Full
Capacity",''Okay'') -> Okay
```

INDEX

Returns a specified cell from a range of cells.

INDEX(*range*, *row*, *col*)

range any range

row the row offset from the upper left of the range

col the column offset from the upper left of the range

The INDEX function returns the cell that is *col* columns and *row* rows offset from the upper left corner of the range *range*.

```
     A     B     C
1     10    20    30
2   3942  4932  5929
3   4920  5929  9294
=INDEX(A1:C3,2,2) -> 9294

A10: 1
A11: 0
=INDEX(A1:C3,A10,A11) -> 3942
```

@INDEX

Returns a given cell from a range of cells.

INDEX(*range*,*col*,*row*)

range any range

col the column offset from the upper left of the range

row the row offset from the upper left of the range

The INDEX() function returns the cell that is *row* rows and *col* columns offset from the upper left corner of the range *range*.

```
     A     B     C
1     10    20    30
2   3942  4932  5929
3   4920  5929  9294
@INDEX(A1:C3,2,2) = 9294

A10: 0
A11: 1
@INDEX(A1:C3,A10,A11) = 3942
```

INDEX and @INDEX differ slightly. INDEX() specifies the offset in row then column order. @INDEX() specifies the offset in column then row order.

ISEMPTY or **@ISEMPTY**

Determines if the cell at the given address is empty.

ISEMPTY(*adr*)

adr a cell address

This functions tests whether or not the specified cell is empty. Mesa functions that return a True or False result return 1 for True, and 0 for False.

```
A1 : 5.2
B1 :
=ISEMPTY(A1) is 0
=ISEMPTY(B1) is 1 because cell B1 has no
contents.
```

ISERROR or **@ISERR**

Determines if the given expression returns an error other than @NA.

ISERROR(*exp*)

exp an expression

This function tests whether or not the given expression results in an error. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False. See “ISNA or @ISNA”*136*.

```
=ISERR(ERRORCONST) -> 1 (true)

=ISERR(@VALUE("ABCD") -> 1 (true)   :  This
formula tried to convert "ABCD" to a value
using @VALUE().  This returns !ERR, which
causes ISERR() to returns 1.
```

ISFORMULA or **@ISFORMULA**

Determines if the cell at the given address contains a formula.

ISFORMULA(*adr*)

adr a cell address

This function tests to see if the cell at the given address contains a formula. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False.

```
A1 : 3.2
B1 : =A1+1
=ISFORMULA(A1) is 0
@ISFORMULA(B1) is 1 because cell B1 contains a
formula that adds 1 to the contents of cell
A1.
```

ISNA or **@ISNA**

Determines if the given expression returns an NA error.

ISNA(*exp*)

exp an expression

This function tests to see if the given expression returns the NA error. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False.

```
=ISNA(@NA) -> 1 (true)

A1: 34
A2: 49
A3: =NA
=ISNA(SUM(A1:A3)) -> 1  :  This formula
returns 1 (true) because cell A3 contains NA,
which causes SUM(A1:A3) to evaluate to NA
also.
```

ISNUMBER or **@ISNUMBER**

Determines if the given expression returns a numeric result.

ISNUMBER(*exp*)

exp an expression

This function tests to see whether or not the given expression results in a number. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False.

```
=ISNUMBER(FIXED(300,0)) -> 0  :  This formula
converts the number "300" to a string using
FIXED(). It then uses ISNUMBER() to test if
that result is a value and returns 0 (false).

=ISNUMBER(34) -> 1

A3: +394 - 100
=ISNUMBER(A3) -> 1
```

ISREF or **@ISREF**

Determines whether or not the given expression is a cell or range reference.

ISREF(*exp*)

exp an expression

This function tests to see whether or not the given expression is a cell or range reference. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False.

```
=ISREF(A1) -> 1 (true)

If RANGE1 has been assigned A1:C10,
=ISREF(RANGE1) -> 1
```

ISSTRING or **@ISSTRING**

Determines whether or not the given expression results in a string.

ISSTRING(*exp*)

exp an expression

This function tests whether or not the given expression will result in a string when evaluated. In Mesa functions that return a True/False result, 1 corresponds to True, and 0 corresponds to False.

```
=ISSTRING(VALUE("340")) -> 0
=ISSTRING(LEFT("Goodbye!",4)) -> 1
=ISSTRING("Some text") -> 1
```

@N

Returns the given expression if it is a number, 0 otherwise.

@N(*exp*)

exp an expression

If the given expression is a number, @N returns the expression, otherwise @N returns 0. @N is guaranteed to return a number and is used in functions that require numbers as parameters.

```
@N(@STRING(300,0)) -> 0
@N(34) -> 34
```

NEXT or **@NEXT**

Returns the next logical value for a given value

NEXT(*value*[, *inc*])

value a number, string, or cell reference

inc the [optional] number by which to increment the value

The NEXT function returns the next logical value after *value*. The value is incremented by 1 unless an optional increment *inc* is specified. For example, if *value* is “Mon”, then NEXT(*value*) returns “Tue”. Values for months, days, and quarters are built in. If *value* is a number, NEXT adds 1 or *inc* to the number. For quarterly information, it will increment the quarter. When the quarter reaches 4, the next quarter is Q1 of the following year. For example, NEXT(“Q4 ’92”) is “Q1 ’93”.

```
=NEXT("Q4 ’92") -> Q1 ’93
=NEXT("January") -> February
=NEXT("Region 1") -> Region 2
```

NOT or **@NOT**

Logical NOT.

NOT(*num*)

num any number

If *num* is zero or not a number, the NUM() function returns 1; otherwise, it returns 0.

```
=NOT(34) -> 0
=NOT("Text") -> 1
```

RANGE or **@RANGE**

Gives the address for a numerically specified range.

RANGE(*ur, lc, lr, rc*)

ur number corresponding the upper row of the range

lc number corresponding the left column of the range

lr number corresponding the lower row of the range

rc number corresponding the right column of the range

RANGE returns the address for the range specified by the upper row, left column, lower row, right column. RANGE(0,0,1,1) is the range A1:B2. This can be useful in MScript scripts.

```
A1 : 0
B1 : 1
A2 : 5
B2 : 7
=SUM(RANGE(0,0,1,1)) is 13. This is the sum of
the values in the range A1:B2
=SUM(RANGE(A1,A1,B1,B1)) is also 13. Because
cell A1 contains 0 and cell B1 contains 1,
this is the same as the above example.
```

ROWS or **@ROWS**

Returns the number of rows in a range.

ROWS(*range*)

range the address of a worksheet range

This function counts the number of rows in the range *range* and returns the number.

```
=ROWS(A1:H20) -> 20
If RANGE1 is the name assigned to the range
B1:CC200, =ROWS(RANGE1) -> 200
```

@S

Returns the given expression if it is a string, or a zero length string otherwise.

@S(*exp*)

exp an expression

If the given expression is a string, this function returns the expression; otherwise it returns a zero length string. This function is used to guarantee that an expression is a string.

```
@S(343) -> "" (blank string)

A1: 2000
A2: Orders
@S(A1)&@S(A2) = Orders
```

SAME or @SAME

Executes the same function as in the given cell

SAME(*adr*)

adr the address of the cell containing the desired formula

Takes the formula in cell *adr* and executes it as if it had been copied and then pasted into the current cell. This is a powerful and valuable function. You can have a master function and a series of functions that are the same. Where you change the master function, the method of calculation used by the other functions is changed. You can use SAME() in conjunction with INDEX(), HLOOKUP(), VLOOKUP(), and to define a formula used in a calculation rather than having a complex IF() statement. An application for this is an employee pay calculation that depends on an employee type.

```
A1: 1
A2: 2

If B1: =A1 * 5 and B2: =SAME(B1), then B2 = 10
(A2*5).
If B1: =A1/5, then B2 changes to 0.4 (A2/5)
If B1: =SIN(A1), B2 changes to 0.9093
(SIN(A2))
```

SETALTCOLOR or @SETALTCOLOR

Returns the given value and sets the alternate color to the specified color.

SETALTCOLOR(*exp, colorconstant [, pattern]*), or
SETALTCOLOR(*exp, r, g, b [, pattern]*)

exp any expression

colorconstant
any color constant

r a number from 0 to 1, indicating the amount of red

g a number from 0 to 1, indicating the amount of green

b a number from 0 to 1, indicating the amount of blue

Sets the alternate color of the current cell and returns `the` given expression. *Colorconstant* is one of the colors listed in the "Constants" section (see "Color Constants" 73), or *r, g*, and *b* are numbers between 0 and 1 that set the percentage of red, green, and blue that will be mixed together to create the color. White is all ones and black is all zeros. *Pattern* is a number greater than or equal to 0, referring to one of the patterns available in the Patterns Panel. If any of the color components or the pattern number are out of bounds or result in errors, the color is not changed.

```
=SETALTCOLOR(A1,IF(A1 < -100,1,0),0,0) : This
sets the alternate color to red if the number
is less than -100, otherwise it sets the
alternate color to black.
=SETALTCOLOR(A1,MAGENTACOLOR) : This sets the
alternate color of cell A1 to Magenta.
```

SETBKGCOLOR or **@SETBKGCOLOR**

Returns the given value and sets the background color to the specified color.

SETBKGCOLOR(*exp, colorconstant [,pattern]*), or
SETBKGCOLOR(*exp, r, g, b [,pattern]*)

exp any expression

colorconstant
any color constant

r a number from 0 to 1, indicating the amount of red

g a number from 0 to 1, indicating the amount of green *b* a number from 0 to 1, indicating the amount of blue

Sets the background color of the current cell and returns `the given` expression. *Colorconstant* is one of the colors listed in the “Constants” section (see “Color Constants” *73*), or *r, g*, and *b* are numbers between 0 and 1 that set the percentage of red, green, and blue that will be mixed together to create the color. White is all ones and black is all zeros. *Pattern* is a number greater than or equal to 0, referring to one of the patterns available in the Patterns Panel. If any of the color components or the pattern number are out of bounds or result in errors, the color is not changed.

```
=SETBKGCOLOR(A1,IF(A1 > 50 && A1 <
100,1,0),0,0) : This sets the background color
to red if the number is greater than 50 and
less than 100, otherwise it sets the
background color to black.
=SETBKGCOLOR(A1,CYANCOLOR,56) : This sets the
background color of cell A1 to Cyan and the
pattern to pattern number 56.
```

SETCOLOR or @SETCOLOR

Returns the given value and sets the text color to the specified color.

SETCOLOR(*exp, colorconstant [,pattern]*), or
SETCOLOR(*exp, r, g, b [,pattern]*)

exp any value

colorconstant
any color constant

r a number from 0 to 1, indicating the amount of red

g a number from 0 to 1, indicating the amount of green

b a number from 0 to 1, indicating the amount of blue

Sets the color of the current cell and returns expression. C*olorconstant* is one of the colors listed in the “Constants” section (see “Color Constants” *73*), or *r, g,* and *b* are numbers between 0 and 1 that set the percentage of red, green, and blue that will be mixed together to create the color. White is all ones and black is all zeros. `Pattern` is a number greater than or equal to 0, referring to one of the patterns available in the Patterns Panel. If any of the color components or the pattern number are out of bounds or result in errors, the color is not changed.

```
=SETCOLOR(A1,IF(A1 > 100,1,0),IF(A1 >
100,0,1),0) : This sets the color to red if
the number is greater than 100, otherwise it
sets the color to green.
=SETCOLOR(A1,BLUECOLOR) : This sets the color
of cell A1 to Blue.
```

SIGNAL or **@SIGNAL**

If the condition is true, sends range of cells to the named Mach port.

SIGNAL(*cond*,*port*,*range*)

cond This function only sends data if this parameter is TRUE.

port A string containing the name of the port to send the data to.

range The range of cells to send to the named port.

This function works in conjunction with the MesaListen class of objects to allow a spreadsheet to send a range of cells to a custom application in when a certain event takes place. The MesaListen object in the custom application can then take the appropriate action. An example is that a signal could be sent if an arbitrage opportunity presented itself. The parameters of the opportunity could be programmed into the worksheet. When the parameters became true, perhaps based of real time data being fed into the worksheet, the worksheet could send a signal to a custom application that would perform a securities trade.

```
=SIGNAL(GOLD_PRICE > 50 *
SILVER_PRICE,''BUY_GOLD'',A1) - send a message
to the ``BUY_GOLD'' object when the price of
gold is 50 times higher than the price of
silver.
```

VLOOKUP or **@VLOOKUP**

Looks up `code` in the vertical `table`.

VLOOKUP(*code*, *table*, *offset*)

code a string or number identifying the desired values

table the range of the lookup table

offset the offset of the relevant column

Returns the cell *offset* cells from the left side of the first match. If the first column contains only strings and code is a number, it returns the value from the last row. If *code* is also a string, but doesn’t match any of the table entries, !ERR is returned.

```
     A    B
1    A    5.0
2    B    4.0
3    C    3.0
4    D    2.0
5    F      0
```

```
=VLOOKUP("A",A1:B5,1) -> 5.0
=VLOOKUP("D",A1:B5,1) -> 2.0
=VLOOKUP("Incomplete",A1:B5,1) -> !ERR
=VLOOKUP(3.0,A1:B5,1) -> 0
```

String Functions

CHAR or **@CHAR**

Returns the ASCII character for the given number.

CHAR(*num*)

num any integer representing an ASCII character

The CHAR function will convert the integer *num* to its corresponding ASCII character. The ASCII code for a given character can be found with the CODE function. See "CODE or @CODE" 145.

```
=CHAR(68) -> D
@CHAR(162) -> ¢
```

CLEAN or **@CLEAN**

Removes all control characters from a string.

CLEAN(*string*)

string any string

The CLEAN function will remove all control characters from the string *string*. This can be useful for data that has been imported to Mesa from another source.

CODE or **@CODE**

Returns the ASCII code for the first character in a string.

CODE(*string*)

string any string (only the first character will be used)

The CODE function returns the ASCII code for the first character in the string *string*. The character for a given code can be found with the CHAR function. See "CHAR or @CHAR"145.

```
=CODE("A") -> 65
@CODE("{") -> 123
=CODE("Hello") -> 72  :  This formula is
returning the ASCII code for "H"
```

EXACT or **@EXACT**

Compares two strings to determine if they are exact matches.

EXACT(*str1*, *str2*)

str1 the first string to be compared

str2 the second string to be compared

The EXACT function compares *str1* and *str2*, and returns 1 if the strings match exactly. It returns 0 otherwise. The EXACT function is case sensitive. When you use = to match strings, the comparison ignores case. For example, "Upper" = "upper" is true, but EXACT("Upper", "upper") is false.

```
=EXACT("Daytime","Daytyme") -> 0 (false)
A1="hello" and B1="hello": =EXACT(A1,B1) -> 1
(true)
```

FIND or **@FIND**

Finds the location of a substring within a string.

FIND(*substr*, *string*, *offset*)

substr the substring to be found

string the string to be searched

offset the location in the string to begin the search

The FIND function returns the location of the sub-string *substr* in the string *string*. It starts searching for *substr offset* characters from the left of *string*. It returns ERR if the string is not found.

```
=FIND("hampton","Northampton",1) -> 4

=FIND("hampton","Northampton",6) -> !ERR
(because it starts searching at the 6th
character)

A1:  "The third word in this sentence."
=FIND("word",A1,1) -> 10
```

FIXED or **@STRING**

Rounds a number to a specified number of decimal places and converts it to a string.

FIXED(*num*, *prec*)

num any number

prec the number of places of precision

The FIXED function rounds the number *num* to a precision of *prec* decimal places, then converts it to a string. If *prec* is negative, *num* will be rounded off to the left of the decimal point.

```
B3:  5251.769585987
=FIXED(B3,6) = 5251.769586

=FIXED(3/43,2)&" is now  a string" -> 0.07 is
now a string  :  This formula evaluates 3/43,
then converts it to a string with 2 decimal
places of precision and concatenates it with
the text " is now a string".
```

LEFT or **@LEFT**

Returns a substring from the left of a string.

LEFT(*string*, *len*)

string any string

len the length of the substring

The LEFT function returns a string containing the first *len* characters from the left of the string *string*.

```
=LEFT("The first twelve characters",12) -> The
first tw
=LEFT("ABCDEFG",4) -> ABCD
```

LEN, **LENGTH**, **@LENGTH** or **@LEN**

Returns the length of a string.

LENGTH(*string*)

string any string

The LENGTH function returns the number of characters in the string *string*.

```
LENGTH("10% discount on COD") = 19
A1:  "A short string "
B1:  "sentence"
=LEN(A1) -> 15
=LENGTH(A1&B1) -> 23
```

LOWER or **@LOWER**
Converts a string to lower case.

LOWER(*string*)

string any string

The LOWER function converts all the characters in the string *string* to lower case, regardless of their original case.

```
=LOWER("Bill Johnson") -> bill johnson
```

MID or **@MID**
Returns a substring from the middle of a string.

MID(*string*, *start*, *len*)

string any string

start the starting character of the substring

len the length of the substring

The MID function returns a string of *len* characters from the string *string*, starting *start* characters from the first character of *string*.

```
= MID("Have a great day!",7,5) = great

A1:  "Robert Quimby Douglas",
=MID(A1,7,6) -> Quimby
```

NEXT or **@NEXT**
Returns the next logical value for a given value

NEXT(*value*[,*inc*])

value a number, string, or cell reference

inc the [optional] number by which to increment the value

The NEXT function returns the next logical value after *value*. The value is incremented by 1 unless an optional increment *inc* is specified. For example, if *value* is “Mon”, then NEXT(*value*) returns “Tue”. Values for months, days, and quarters are built in. If *value* is a number, NEXT adds 1 or *inc* to the number. For quarterly information, it will increment the quarter. When the quarter reaches 4, the next quarter is Q1 of the following year. For example, NEXT(“Q4 ’92”) is “Q1 ’93”.

```
=NEXT("Q4 ’92") -> Q1 ’93
=NEXT("January") -> February
=NEXT("Region 1") -> Region 2
```

PROPER or **@PROPER**

Converts a string to Proper Case.

PROPER(*string*)

string any string

The PROPER function capitalizes the first letter of every word in the string *string* and converts all remaining characters to lower case.

```
=PROPER("MS. JANET KRAFT") -> Ms. Janet Kraft
=PROPER("all entries posted before 7/2") ->
All Entries Posted Before 7/2
```

REPEAT, **REPT**, or **@REPEAT**

Repeats a string the specified number of times.

REPEAT(*string*, *num*)

string any string

num the number of times to repeat the string

This function returns the string *string* repeated *num* times. The string is placed back-to-back with itself, with no white space.

```
=REPEAT("X",10) -> XXXXXXXXXX
=REPT("over and ",4) -> over and over and over
and over and
```

REPLACE or **@REPLACE**

Replaces a specified number of characters in a string with another string.

REPLACE(*string*, *offset*, *num*, *newstr*)

string any string

offset the position to begin the insertion *num* the number of characters to replace

newstr the string with which to replace the specified characters

The REPLACE() function replaces *num* characters in the string *string* with the string *newstr*, starting *offset* characters from the first character of the original string. The string *newstr* does not need to be the same length as the number of characters being replaced.

```
=REPLACE("The cost is $450, COD",13,3,"2,500")
-> The cost is $2,500, COD
=REPLACE("Robert J. Stevenson",7,1,"L") ->
Robert L. Stevenson
```

RIGHT or **@RIGHT**

Returns a substring from the right of a string.

RIGHT(*string*, *len*)

string any string

len the length of the substring

The RIGHT function returns a string containing the first *len* characters from the right of the string *string*.

```
=RIGHT("The last ten characters",10) ->
characters
=RIGHT("Ida M. Brown",5) -> Brown
```

TRIM or **@TRIM**

Removes extra white space from a string

TRIM(*string*)

string any string

This function removes all leading and trailing spaces from the string *string* and converts multiple consecutive spaces within the string to single spaces.

```
A3:  Account #        3040302     40303
=TRIM(A3) -> Account # 3040302 40303

=TRIM("    Anything    at    all.") ->
Anything at all.
```

UPPER or **@UPPER**

Converts a string to UPPER case.

UPPER(*string*)

string any string

The UPPER function converts all the characters in the string *string* to upper case, regardless of their original case.

```
=UPPER("Uppers only") -> UPPERS ONLY

=UPPER(MID("robert c. smith",8,2)) -> C.  :
This formula uses MID() to return the 8th and
9th characters from the string "Robert C.
Smith".  The UPPER() function then capitalizes
this result.
```

VALUE

Returns the number value of a string.

VALUE(*string*)

string any string

This function converts the string *string* into a number. If the string begins with a number but contains non-numbers, VALUE() returns as much as it can. If *string* begins with a letter, it returns 0.

```
=VALUE("3.14159") = 3.14159
=VALUE("1200") + 2000 = 3200
=VALUE("320/104") = 320
```

@VALUE

Returns the number value of a string.

@VALUE(*string*)

string any string

This function converts the string *string* into a number. If the string contains characters that are non-numbers, it returns an error.

```
@VALUE("3.14159") -> 3.14159
@VALUE("1200") + 2000 -> 3200
@VALUE("320/104") -> !ERR
```

VALUE() and @VALUE() are different in that if the argument to VALUE() cannot be completely converted to a *number, it returns as much of the number as possible. @VALUE() returns an error.*

Smart Functions

Mesa provides menu options in *Range->Smart Formulas* to allow you to quickly and easily perform the most common Function operations:

SUM
MIN
MAX
STDDEV
VAR

To use any of these functions, just highlight the range you want to operate on, as well as one cell to the right or below that range (this is where the answer will be put).

Then choose the appropriate Smart Function from the *Range-> Smart Formulas* menu.

Mesa will immediately create a formula of the form:

=SUM(a1.h5)

in the appropriate cell.

You can edit this formula manualy if you want to do so.

Formula Builder

Use Tools->Formula Builder or click the Formula Builder icon on the Worksheet Iconbar to get an inspector panel containing a scrollable list of all of the formula functions available in Mesa.

A pop-up menu lets you choose among the following categories: math, financial, string, constants, special, date, database, and operators. (You can select functions either with or without the @ symbol.)

Formulas beginning with the @ character (1-2-3 derived) can be pasted directly into the worksheet, but to use formulas without the @ (Excel-derived) , type ‘‘=’’ in the data input window of your worksheet, and then define the formula by selecting and pasting non-@ functions from the Formula Builder lists. You must type in any variables required by a function before you can go on.

You can paste operators from the Formula Builder *Operators* list into either @ formulas or non-@ formulas.

✓ Use the ’Sizing dimple’ to display more, less or none of the explanatory text associated wiith Functions.

Manual / Automatic Recalculation

Mesa always uses “natural order” recalculation so that the one answer for a given cell is returned correctly even if cell data in other parts of the worksheet are accessed.

By default, Mesa recalculates the entire worksheet each time data in the worksheet is changed. For large worksheets, this could take a second or two.

To disable automatic recalculation, uncheck the “Auto Recalc” check-box in the Sheet Inspector.

If Auto Recalc is disabled, remember to use the **Range** → **Recalculate Sheet** (*Command*-=) command to update the worksheet after data has changed.

✔ **Circular References**

Mesa cannot resolve circular references in cell formulas. Circular Reference debugging is provided to track and allow you to fix circular references.

To enable calculation debugging, select “Stop on Error” or “Stop on Circ.” from the pop-up in the Sheet Inspector.

Sizing the Worksheet

Use the Sheet Inspector to add rows or Columns to the Current worksheet

Use Preferences to change the dimensions of all future worksheets.

Resizing Rows

Rows can be made higher or lower - usually in order to accomodate large or smaller font-sized data.

You can do this in three different ways:

Manually: by moving the mouse to the separator in the Row Frame and dragging the row to the required height

Via the menu: use *Arrange->Row/Column Smart Resize* to set the row to the maximum height needed to accomodate the largest fontsize in any cell along the row

Automatic by Mouse: double-click on the Row Frame separator to do a Smart Resize.

✔ Adjacent rows can be resized at the same time. Click and drag in the Row Frame to select the rows you want: then use any of the above methods to resize.

Use *Command-Click-Drag* to resize *all* the rows in your selection

Use *Command-DoubleClick* to Smart Resize *all* the rows in your selection

See also

Column Resize *158*
Deleting and Inserting Rows *159*

Resizing Columns

Columns can be made wider and narrower - usually in order to accomodate large or smaller font-sized data.

You can do this in three different ways:

Manually: by moving the mouse to the separator in the Column Frame and dragging the column to the required height

Via the menu: use *Arrange->Row/Column Smart Resize* to set the row to the maximum height needed to accomodate the largest fontsize in any cell along the row

Automatic by Mouse: double-click on the Column Frame separator to do a Smart Resize.

✓ Adjacent columns can be resized at the same time. Click and drag in the Column Frame to select the rows you want: then use any of the above methods to resize.

Use *Command-Click-Drag* to resize *all* the columns in your selection

Use *Command-DoubleClick* to Smart Resize *all* the columns in your selection

See also

Row Resize *157*
Deleting and Inserting Columns *160*

Inserting and Deleting Rows

To insert a new row, select the row in the Row Frame to the left of which you want to add a row, and use *Edit->Insert*

To delete a row, select the row in the Row Frame, and use *Edit->Delete*

✔ Adjacent rows can be inserted or deleted at the same time. Click and drag in the Row Frame to select the number of rows you want: then use the appropriate command to insert or delete the rows.

✔ To add rows at the Bottom edge of the worksheet, use the Sheet Inspector's Size option.

See also

Row Resize *157*
Deleting and Inserting Columns *160*

Inserting and Deleting Columns

To insert a new column, select the column in the Column Frame to the left of which you want to add a column, and use *Edit->Insert*

To delete a column, select the column in the Column Frame, and use *Edit->Delete*

✓ Adjacent columns can be inserted or deleted at the same time. Click and drag in the Column Frame to select the number of columns you want: then use the appropriate command to insert or delete the columns.

✓ To add columns at the Right edge of the worksheet, use the Sheet Inspector's Size option.

See also

Hiding/Unhiding Rows & Columns

To hide rows, select the row(s), and use *Arrange -> Row/Column Size -> Hide Rows.*

To hide columns, select the column(s), and use *Arrange -> Row/Column Size -> Hide Columns.*

To unhide rows and columns choose the appropriate unhide command from the *Arrange -> Row/Column Size* menu.

✓ To choose the rows or columns to unhide once they are hidden, select the rows or columns that surround them. All hidden parts in this range will be set to visible status on selection of the unhide command.

See also

Deleting and Inserting Columns *160*

Freezing worksheet Titles

Use Mesa’s Worksheet Title Bars to ’freeze’ rows and columns on the screen.

✔ This is very useful when a worksheet has more columns or rows than you can display simultaneously within the worksheet window. For example, month names in the top row of the sheet can be frozen so that you can still see them even when you move more than 20 rows down the sheet.

Row Freeze Drag the thick black Title Bar at the top of the worksheet frame down into the worksheet. The content of all rows above the Bar will ’freeze’ on the screen when you next scroll downwards.

Column Freeze Drag the thick black Title Bar at the left side of the worksheet frame rightwards into the worksheet. The content of all rows to the left of the Bar will ’freeze’ on the screen when you next scroll rightwards.

✔ You can have both a Row and Column Freeze at the same time

✔ To **UnFreeze**, just drag the Title Bar back to it’s original position.

✔ *Any* rows or columns can be Frozen. Move first to the area you want to freeze, and then drag the Title Bars across.

See also

Worksheet Windows

Use *Windows->New Window* to create a new view of your currrent worksheet.

✔ This is very useful when a worksheet has more columns or rows than you can display simultaneously within the worksheet window. For example, you could show the Totals area in one window, the Input area in a different window, and a graph in a third window.

All windows can be moved, resized etc just like the main worksheet window.

To close the view window either click on the close box in the top right-hand corner of the window, or choose *Windows -> Close Window* from the menus.

✔ If you're building a formula, you can't click into another windiw to put it's address into the formula. You have to *type* the cell address.

✔ There is no realistic limit to the number of windows you can create for a single worksheet.

✔ When you save the Worksheet, the windows and their relative positions are saved as well. When you reopen the WorkSheet, all the windows will open up too.

See also

Linking Worksheets with Formulas

You can link Mesa worksheets so that information from one worksheet can be used in another. The file link is entered into a cell, just like a formula.

Mesa’s file link syntax is:

```
filename::range
```

or

```
filename!range
```

where *filename* is the path to the file to be linked. If *filename* is a path, it must be an absolute path. If it is not a path, Mesa searches for the file in the directory where current worksheet is saved. *Filename* can either be a string or a cell reference. This allows you to link to different worksheets based on the contents of a cell *Range* can be a cell address, range, or label.

If both files are in the same directory, you can use just the filename. Otherwise, you must specify the path of the linked file. Examples are given in the table below.

Linked Worksheet Examples

Formula	**Result**
=“Sales”::A1	Returns contents of cell A1 in worksheet “Sales” located in the same directory as the current worksheet.
=“/sam/Sales.Mesa”::c5	Prices Returns the contents of cell C5 in the “/sam/Sales” worksheet.
=sum(“Sales”!a1:b7)	Returns the sum of cells A1 through B7 in the “Sales” worksheet located in the same directory as the current worksheet.
=sum(“Sales”::Western)	Returns the sum of the cells in the range “Western” in the

	"Sales" worksheet located in the same directory as the current worksheet.
=X1::A1	Returns the contents of cell A1 in the worksheet whose name is contained in cell X1.
=sum(X1::widgets)	Returns the sum of the "widgets" range in the worksheet whose name is contained in cell X1.

Note that the NEXTSTEP environment is sensitive to capital letters in file names, so be sure to spell file names exactly as you see them. File names may contain spaces, which must be included in link references as well. When you work with linked files, be aware of the following points:

When you open a worksheet that has links, recalculate using *Command-=* unless Auto Recalc is on.
Links can be used in formulas, just like any other valid expression. For example: *@PV(2500,4%, "US_SALES"::b5*).

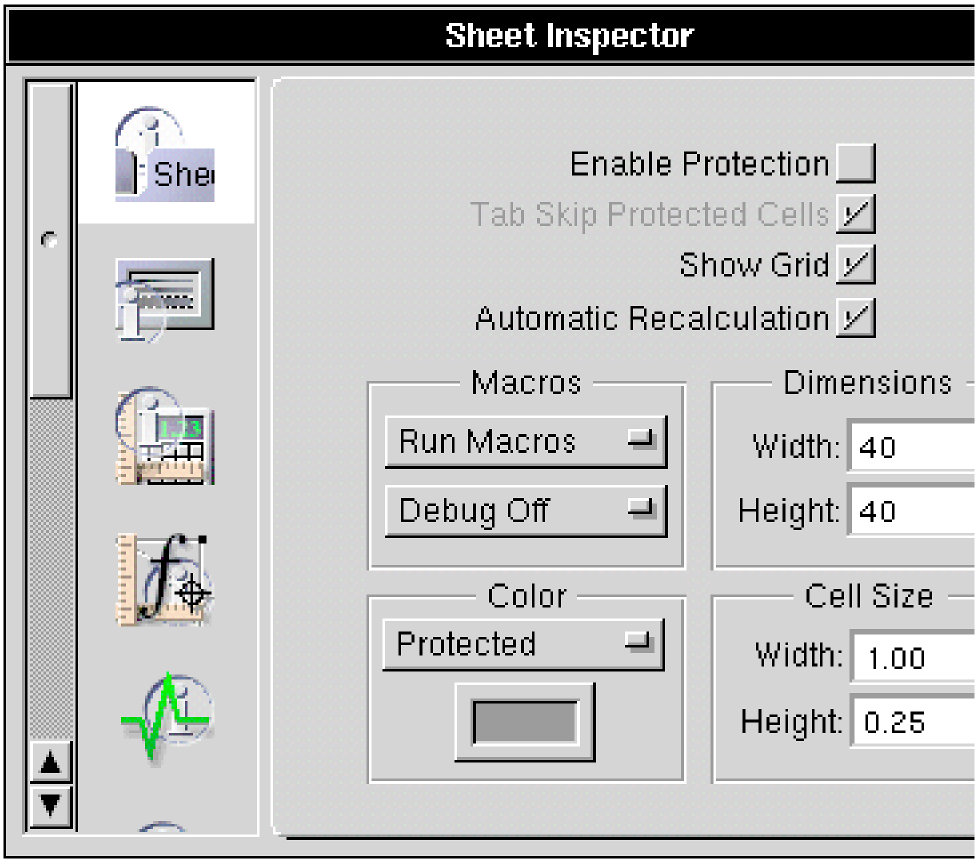
Referencing files via cells (i.e., *Q5::Prices*) is a powerful tool that allows changing where the data for a calculation is coming from without changing the formula. If you have sales data from four regions saved in worksheets named, *North*, *South*, *East*, and *West*, you could build a summary worksheet to generate totals for each region.

The formula *=sum(b5::total_sales)* would sum the total_sales region for the sheet named in cell B5. To see the summary for *North*, enter "North" into cell B5 and select **Sheet** → **Recalculate** (*Command-=*). To see the summary for *West*, enter "West" into cell B5 and select **Sheet** → **Recalculate**.

Sheet Inspector

The Sheet Inspector changes items that affect the whole worksheet or all cells in the worksheet. The **Tools** → **Inspector...** command opens the Inspector box, the sheet inspector is the top item in the list.

The Sheet Inspector displays and changes parameters of the currently active worksheet. When all worksheets close, the Sheet Inspector will automatically close because it has nothing to inspect.



Dimensions

Mesa worksheets can have up to 16,384 rows and columns. However, when a worksheet has this many cells, using the scrollers to move around the worksheet becomes very clumsy. The number of rows and columns displayed in the worksheet can be viewed and changed in the Dimensions section of the Sheet Inspector.

To change the dimensions, enter the desired number of rows or columns in the height or width boxes. Mesa worksheets also grow automatically when you move beyond the edge with the arrow keys.

NB. Demo versions of Mesa do not allow the dimensions to be changed.

The dimensions of the worksheet affect only how the information on the worksheet is displayed. A worksheet may display only 32 rows and columns, but may contain data in any cell in the worksheet. The data in cells that is not displayed is not lost and can be used in formulas and viewed by setting the worksheet dimensions to the appropriate size. There is no speed or storage size penalty associated with larger dimensions.

Recalc

Mesa always uses "natural order" recalculation so that the one answer for a given cell is returned correctly even if cell data in other parts of the worksheet are accessed. However, Mesa cannot resolve circular references in cell formulas. Circular Reference debugging is provided to track and allow you to fix circular references. To enable calculation debugging, select "Stop on Error" or "Stop on Circ." from the pop-up. To step or trace through macros, select the appropriate item from the pop-up.

By default, Mesa recalculates the entire worksheet each time data in the worksheet is changed. For large worksheets, this could take a second or two. To disable automatic recalculation, uncheck the "Automatic Recalculation" check-box. If Automatic Recalculation is disabled, remember to use the **Range** → **Recalculate** command to update the worksheet after data has changed.

Grid

The "Show Grid" check box turns the worksheet's grid on and off. The color of the grid can be changed by dropping a color chip in the color well in the "Color" section of the inpector when the drop-down list is set to "Grid". You can change the background color of the entire worksheet by dropping a color chip inside the same color well when "Background" is selected from the list. Similarly, the color of protected cells can be set by the same method.

All cells that do not have background shading will display this color as their background. If the "Clear" check box is checked, the background will appear white on the screen, but if the cells are copied and pasted to another application, the background from that application will show through. This is useful for creating presentations with Concurrence(tm) or Create(tm).

Protection

Cell Protection is used to block data entry into cells that are not specifically typed Unprotected. . To enable cell protection, check the ‘‘Enable Protection’’ check box.

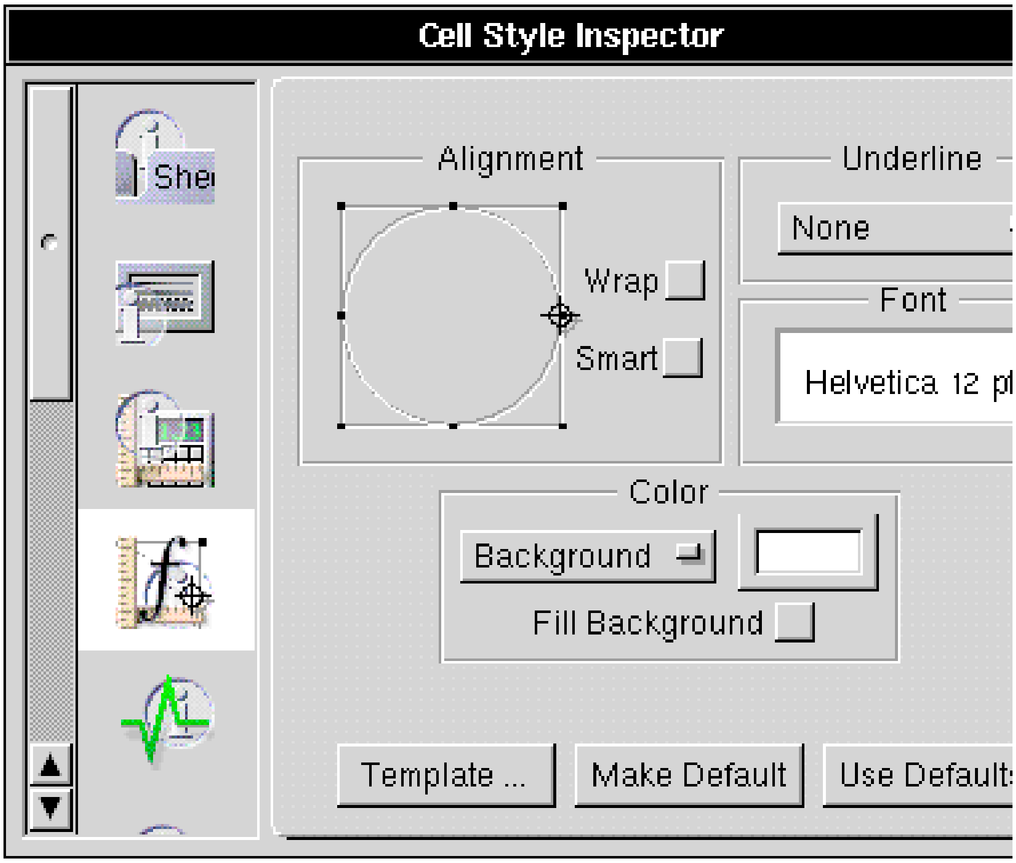
If protection is enabled, protected cells will be skipped when tabbing through a range if ‘‘Skip Protected Cells’’ is selected. Protected cells will be displayed in a special color if ‘‘Show in Special Color’’ is selected and the worksheet is protected. The color of the protected cells can be changed by dropping a new color into the color well.

Size

This allows the default cell height and width to be changed. The current default width and height is displayed in the current measurement units. To change the width or height,type the desired size directly into the height and width boxes.

Changing the default will not change the size of rows or columns that have been resized. For more information, see ‘‘Resizing Rows*157* and Columns*158*’’ .

The Cell Inspectors



The Cell Style and Cell Format Inspectors, used to inspect and change the contents of a cell, reflects the style and formatting information of the current cell. Changes made via the Cell Style Inspector will affect the style of all cells in the current range.

The Inspectors

Each view contains a button designated " Default." If this is selected, the default format from the current style template is used. If a change is made to the format, the template will be overridden and the new selection will be used for the cell.

To go back to using the default item for the cell, click on the *Default* button.

Changes made in the cell style inspector are applied to all cells in the currently selected range.

Format

The format view allows selection of the format used to display the value of the cell. Most items apply only to cells with a number value. Exceptions are the Text and Hidden formats.

The Text format displays the input of the cell. If a cell has a formula, the formula - not the calculated value - is displayed. The Hidden format displays nothing in the cell, as if the cell is blank. This format can be used to hide intermediate results that would otherwise clutter the worksheet.

Date Int'l 1 displays date serial numbers in the format MM/DD/YY.

Date Int'l 2 displays date serial number in the format MM/DD.

Time Int'l 1 displays the time portion of date serial number in the format HH:MM:SS.

Time Int'l 2 displays the time portion of date serial numbers in the format HH:MM.

The decimal selector changes the number of decimal places displayed when Fixed Decimal, Scientific, Currency, Comma, and Percent formats are selected.

Some commonly used formats are also available in the Range->Format->Quick menus.

Color

The information in cells is displayed in color. Normally, the contents will be displayed in the ‘‘Text Color.’’ However, if the value is a negative number, it will be displayed in the ‘‘Negative color.’’

The colors may be changed by dragging and dropping a color chip into the appropriate color well. The patterns may be changed by dragging and dropping a pattern chip from the pattern picker into the color well. The text color of a cell or a range of cells can be changed by dragging and dropping a color chip on the cell or range directly on the worksheet. The color picker can be opened by clicking on the border around the color well or with the **Tools** → **Colors...** command, and the pattern picker can be opened with **Tools->Graphics**→ **Patterns...**. It may be useful to place this command on the ToolBar so that the color picker can be quickly opened.

The background or shading of a cell can be changed by *Alt-Dragging* a color chip. To *Alt-Drag* a color chip, press the *Alternate* key before you click on the color well to pick up the color chip. When you drop the color chip on a cell or range, you will not see the color change until you un-highlight it.

The background pattern of a cell can be changed similarly.

By default, the cell is shaded with the background color of the sheet. To set a background fill, click on the *Fill* switch. If the background and the shading color are the same, then you will not see a change; however, if you change the background color of the sheet, the cell will retain its color. To change the cell shading, you can *Alt-Drag* color over the selected cell or range or drop the color in the *Shading* color well in the Cell Style Inspector.

Alignment

Use the Alignment tool to decide on the position of the data within the cell.

The Wrap option wraps string information in the cell down so that it does not spill out the sides of the cell, but is displayed on multiple lines within the cell.

Smart sets text back to left aligned, and Numbers to right-aligned.

Font

This view of the Cell Style Inspector allows you to change the font by dropping a font chip in the font well. Open the Font Panel by double-clicking in the font well, by clicking the button marked *Font*, or by using the **Format** → **Fonts...** (*Command-t*) command.

You can underline the contents of a cell by selecting a single, double, or wide line options.

Input Type

Mesa allows basic checking of data input. Cells can be designated as Numeric, String, Date, or Formula and that cell will accept only input of the designated type. If a cell is typed *Any*, then any type of data is allowed for input. *Unprotected* cells are like cells typed any, yet data can be entered into an unprotected cell even when sheet protection is enabled.

Style Templates

Style Templates can be used to create better looking and more maintainable worksheets. Each worksheet contains eight style templates numbered 1 through 8. Each cell has a format that is derived from the style template but can be changed.

Each cell has a style template, but each parameter about the cell can be changed. By default, cells derive from Style Template 1, but this can be changed either by clicking on another style template in the Cell Style Inspector, by selecting the desired style from the **Format** → **Style Templates** menu, or by typing *Command-1* (*Command-one*) through *Command-8*. After the style template for a cell has been selected, individual formatting items in the cell can be changed from the default. If the style template itself is changed via the sheet inspector, all cells that derive from the changed style template will change all attributes that have not been over-ridden in the particular cell style.

Different parts of the worksheet can be based on different style templates. Because cells default to style template 1, use that template for the basic style of the worksheet. The summary section may be in a different font (bold or larger, for example), so cells in the summary section could be style template 2. In the Sheet Inspector, change Style Template 2 so that its font is bold. Experimenting with style templates will help with these concepts.

Aligning cell contents

Use the Cell Inspector to align the contents of a cell.

Any kind of data - text, numbers or dates - can be aligned, to the Left, Right or Centre..

If you make a row higher, you can also use the Top, Centre and Bottom alignement options.

The *Wrap* option will ’word-wrap’ text within the row height.

✔ Mesa does not let you align data over a width greater than one column.

✔ The *Smart* option aligns Text to the left, and Numbers and Dates to the right.

See also

Row Resize *157*

Changing Color

Use the Sheet Inspector to change the color of data and background for the entire sheet.

Use the Cell Inspector to change the color of data and background for the selected range.

The colors may be changed by dragging and dropping a color chip into the appropriate color well. The text color of a cell or a range of cells can be changed by dragging and dropping a color chip on the cell or range directly on the worksheet.

Open the color picker by clicking on the border around the color well or with the **Format** → **Colors...** command.

By default, the cell is shaded with the background color of the sheet. To set a background fill, click on the *Fill* switch. If the background and the shading color are the same, then you will not see a change; however, if you change the background color of the sheet, the cell will retain its color. To change the cell shading, you can *Alt-Drag* color over the selected cell or range or drop the color in the *Shading* color well in the Cell Style Inspector.

✔ The **Negative** color is the one that Negative numbers will use.

✔ Use *Drag* from the Color Wheel to change the color of the **data** in a cell

Use *Alt-Drag* to change the **background** color of a cell.

When you drop the color chip on a cell or range, you will not see the color change until you un-highlight it.

✔ You can Command-Drag a color chip into the ToolBox.

See also

Patterns *232*

Changing Fonts

Use the Sheet Inspector to change the default font for the entire sheet.

Use the Cell Inspector to change the font for the selected range.

The fonts may be changed by dragging and dropping a font chip into the appropriate font well, or by dragging and dropping a color chip onto the cell or range directly on the worksheet.

Open the font panel by double-clicking on the Font well, or with the **Range**->**Format->Font** command.

✔ The NEXTSTEP Font Panel shows you a scrollable list of all the type families and typefaces available. The panel also has a table of type sizes in points, and a font viewer window (a “font well”).

Click the *Preview* button to see the result in the font viewer. If you like what you see, click the *Set* button and your font will be applied to the cell range you specified.

You can also drag the font from the font well by clicking in the well and dragging the tiny *font chip* icon to the cell or range you specified on your worksheet or drop a font chip on an element in a chart to change its font.

If you change your mind, click *Revert* to restore the original font.

See also

Patterns *232*

Changing Borders

Use the Sheet Inspector to change the default border type for the entire sheet.

Use the Border Inspector to change the border for the selected range.

Use the QuickBorder popup to fill in the commonly-user border types. Or click on each individual part of the 'Cell' in the Inspector

See also

QuickMenus 178

Number and Date Formats

Use the Sheet Inspector to change the number or date display format for the entire sheet.

Use the Cell Inspector to change the number or date display format for the selected range.

Most items apply only to cells with a number value. Exceptions are the Text and Hidden formats.

Options
The decimal selector changes the number of decimal places displayed when Fixed Decimal, Scientific, Currency, Comma, and Percent formats are selected.

Text Format. The Text format displays the input of the cell. If a cell has a formula, the formula - not the calculated value - is displayed.

Hidden Format. The Hidden format displays nothing in the cell, as if the cell is blank. This format can be used to hide intermediate results that would otherwise clutter the worksheet.

Date Formats
Date Int'l 1 displays date serial numbers in the format MM/DD/YY.

Date Int'l 2 displays date serial number in the format MM/DD.

Time Int'l 1 displays the time portion of date serial number in the format HH:MM:SS.

Time Int'l 2 displays the time portion of date serial numbers in the format HH:MM.

✔ Use the QuickMenus to apply the most commonly used formats.

See also

Fonts 175

QuickMenus

The most commonly used Formatting options are available as menu options from the Format menu.

The advantage of this is that you can easily Command-Drag them to the toolbox. We sugegst that you customize the toolbox to contain your most frequently used formatting tools.

Style Templates

Each worksheet contains eight style templates numbered 1 through 8. Each cell has a format that is derived from the style template but can be changed.

Style Templates can be used to create better looking and **more maintainable** worksheets. It's easier to apply 'Style Template 2' to all your headings than to remember every time the exact combination of formats you want to use.

✔ Use The Sheet Inspector to **define** the formatting characteristics for each Style Template.

✔ Use the Cell Inspectors to **apply** a Style to the selected cells

After the style template for a cell has been selected, individual formatting items in the cell can be changed from the default.

Different parts of the worksheet can be based on different style templates. Because cells default to style template 1, use that template for the basic style of the worksheet.

✔ If you change a Style Template via the sheet inspector, all cells that derive from the changed style template will change all attributes that have not been over-ridden in the particular cell style.

See also

Fonts *175*

Copy Formats

Use *Copy* plus *Paste Special->Formats* to copy all the formatting characteristics of a cell.

This lets you paste the format of the cell range that was cut or copied into the pasteboard, to a specified location on a worksheet without pasting any included values.

See also

Style Templates *179*
Paste Special *55*

Negative Colors

Use the Sheet Inspector to set the Negative color for the entire sheet.

All numbers less than zero will be displayed in this color.

Use the Cell Inspector to set the Negative Color for selected cells in the worksheet

Printing - Introduction

Printing is likely to be an important part of your Mesa usage.

Mesa has a flexible approach to printing which lets you do anything from quick-and-simple prints of the entire worksheet, through to complex report layouts designed with the ReportWriter.

✔ If you just want a quick print for draft purposes, look at the *Simple Printing* topic

✔ If you want to use headers and footers, shrink-to-fit, multiple print ranges etc, look at the *ReportWriter* topic

See also

Simple printing	*183*
The ReportWriter	*187*
The Print panel	*194*

Simple Printing

Select the range you want to print, and choose Print from the main menu (or click the Print button on the Worksheet)

Select a printer if necessary from the Print panel, and preview, print or fax your worksheet.

Use the Page Layout panel for simple control of page size etc.

Remember that the ReportWriter gives much more control over print layout.

- ✓ If you don't select a range, Mesa will print from A1 to the bottom leftmost data cell.
- ✓ If your range extends beyond the edges of the worksheet, then you can use the **Range** → **Goto Cell** panel to define the range to print.

  For example, type the range `A1:Z200` in the *Goto Cell* box, click *OK*, and Mesa will select the range from A1 to Z200 to print.
- ✓ Printouts will always show the cell grid. *If you don't want the cell grid to print, use ReportWriter to set gridlines off.*

See also

Using the Page Layout panel *186*
Using ReportWriter *187*

No Help!

Sorry - This helpfile is currently under development.

Setting Print Pagebreaks

Use *Format-> PageBreaks* to force a printout to start a new page at a particular column or row.

Set Vertical Breaks sets a vertical page break at the right of the selected range.
Set Horizontal Breaks sets a horizontal page break below the currently selected range.

Clear Vertical Breaks clears vertical page breaks in the selected range.
Clear Horizontal Breaks clears horizontal page breaks in the selected range.

✔ When page breaks are set, the row or column separation corresponding to the page break changes from white to black.

See also

Using ReportWriter	*187*
Using the Page Layout panel	*186*

Page Layout panel

The Page Layout panel allows you to:

- choose the orientation of the printed page
- choose the paper size
- set the margins
- set simple headers and footers
- do other print-related tasks

NB Settings in the Page Layout panel do not affect Report Writer layouts.

✓ You can include Date, Time, Filename etc in the header and Footer by using *Tags* (see below)

✓ Although you can use the Scaling option to make the print size smaller/larger, it usually involves a lot of manual experimentation to get the right setting. *Use ReportWriter if you need to shrink your printing to fit correctly on the page.*

See also

The ReportWriter

Use the ReportWriter to create sets of printing commands which can be used repeatedly to print your data and make it look good.

Start the ReportWriter with Tools->ReportWriter, or by clicking the ReportWriter button on the worksheet iconbar.

For each report you can:

- define data ranges
- define graphs to print
- set page orientation
- create Header and Footer contents, including graphics
- Smart-Size the print to fit the paper page
- choose whether or not to display gridlines
- set Print Titles
- name the Report
- and more...

To print a Report, either click the Print Report button on the ReportWriter, or select the report you want from the Print panel.

More detail on setting up a report is given in More on ReportWriter.

✔ A worksheet can have lots of reports associated with it.

✔ Reports can only print data from the worksheet they are in.

✔ Reports are saved automatically when you save a worksheet.

See also

More on ReportWriter *188*
Graphics in Reports *235*
Page Breaks

ReportWriter - the details

The following table summarizes the elements of the ReportWriter panel.

Item	Description
Report Name	Each Report has a name. That is how you tell Mesa which report you want to print or change. Report names should describe the report, e.g., Q1_Sales, Checking, Summary_Report.
Print Button	After a report has been added, you can click on the print button to print the currently selected report. The Print Button is a tool that you can Command-drag to the ToolBox or a Tool Holder in a window.
Ranges	A range is a group of cells or a graph that you want to print. Mesa starts printing each range on a new page. A report can contain up to 32 ranges. To print a graph, use the format: g=*graphname*
Range Scaling Options	The range scaling options Smart Width and Smart Scale change the size of the font used in printing the ranges. Smart Width will adjust the printed range so that it is only one page wide. Smart Scale will adjust the scale so that the entire range prints out on a single page.
Page View	A report starts as a blank page. Ranges are printed within the page margins. In addition, you can place tags on the page. You should place tags outside the margins so that they are not printed over. Tags contain information that you want to appear on every page such as print date or time, file name, or some text that you enter.

View Scale	A full-sized page is too big to be displayed in the Report Format panel. You can change the scale of the page (miniaturize it) so that you can see more of the page, or “zoom in” and work on a specific part of the page.
Margins	The margins are the borders on the page within which ranges are printed. You can adjust the margins by moving the mouse over a margin, clicking and dragging until the margin is sized to your satisfaction. If you hold the Command key while you are dragging the margin, Mesa will adjust the corresponding margin to the same size.
Page Layout	The Page Layout allows you to change page attributes. For example, you could print in landscape mode on legal paper by entering 14 for the paper width and 8.5 for the height and clicking “Set.”
Tags	Tags can be dragged and dropped onto your page and moved around. Tags contain information such as print time and date, file name, page number, etc. Tags are printed on every page. Be sure to place tags outside the margins so that they are not covered by the ranges.
Show Grid	This option will print the worksheet grid for all ranges
Print Across	This option will cause multiple-page ranges to print across before printing down.
Show Row/Col Headings	This option will show the worksheet row and column headings for each range.
Ignore Page Breaks	This option will cause any page breaks set within the worksheet to be ignored. Page breaks will still occur between ranges.

See also

Page Orientation 193
Page Breaks 185
Using Tags in Headers and Footers 192
Fonts in Headers and Footers

Setting Print Titles

If a print range is going to be wider or longer than one page then you may want to force 'title' rows or columns to print on the second and subsequent pages.

In the ReportWriter *Range* box, specify the columns or rows you want repeated using the format shown below. Do this *afte*r the print range specification:

You must use the format: T=*range* for Top titles, and S=*range* for Side titles.

Examples:

a2:j23, T=a1.j1

b2.j23, S=a2.a23

b2.j23, T=a1.j1, S=a2.a23

✔ Make sure you don't include the Title columns/rows as part of your print range. If you do, they will print twice on the first page.

✔ This option is only available in the ReportWriter. You can't achieve the same effect using the Page Layout panel.

See also

Printing	*182*
Using ReportWriter	*187*
Using the Page Layout panel	*186*

Page Layout Tags

You can set up simple headers and footers containing text and the following *tags*.

Mesa Tag Formatting Codes

Format	Mesa Code
Page number	&p
Total # of pages	&n
Current date	&d
Current time	&t
Document name	&f
“&”	&&

For example: Printed on &d

✓ Header and footer text/tags are always centered in the page width.

✓ These tags are also available (in a graphic form) in the ReportWriter.

See also

Using ReportWriter 187

Page Orientation

You can easily swap from Portrait to Landscape layout by clicking on the appropriate icon in the Report Layout o rthe Page Layout panels.

Note that in ther Report Layout, Tags are not automatically moved to reflect the new orientation. You’ll have to drag them to the corretc positions.

The Print Panel

Mesa uses the standard NeXTSTEP print panel accessed from the Print option on the main menu.

You choose which printer you wan to print to, or whether you prefer to fax the data or save it as a postscript file.

If you have set up Report Layouts, Mesa displays the list of available reports. To print a report, just select it from the list and carry on with the printing process as usual.

NB. If you don’t select a Report Layout, Mesa will print the currently selected range. If no range is selected, Mesa will print all data on the worksheet.

The Extract Inspector

This Inspector allows you to extract specific information from a database and put it elsewhere in the worksheet.

Extract Queries can be given names and re-used whenever necessary.

Some preparation needs to be done on the worksheet before you use the Extract Inspector. You must allocate an area for each of the following Extract ranges:

Data	(the data you want to enquire about)
Criteria	(where you specify the Conditions you want your data to meet)
Destination	(the area to which rows which meet your Criteria will be copied)

In the Extract Inspector, give the Extract Query a name, and fill in the Range details. Click the Extract Button to run the Extract and populate the Destination Range with the rows which satisfy your criteria.

✓ The button marked **Extract** can be *Command-dragged* ts an Icon on a Tool Holder on the Worksheet or to one of the spots in the Toolbox.

See also

Data Ranges *196*
Criteria Range *197*
Output Range *199*

Data Range

The rows and columns which hold the data you want to enquire about are the Data Range.

- ✔ Rows are sometimes called *Records*
- ✔ Columns are sometimes called *Fields*

The first row of the Data Range must contain Labels as the column identifiers (Field Names)

See also

Criteria Range *197*
Output Range *199*

Criteria Range

The Criteria Range holds the conditions which control which rows are extracted to the Destination Range.

The Criteria Range must contain the same column headings as the Data Range, and must be defined as at least the column names and one row below.

Criteria ranges must contain at least two rows. The top row contains the names of the fields in which to search, and the other rows list the conditions that must be met. Criteria may be numbers, strings, or formulas.

	A	B	C	D	E	F	G
1	ID	Name	Region	Sales		Region	Sales
2	101	John	West	high		West	high
3	102	Ann	South	high		South	
4	103	Barbara	East	low			
5	104	Fred	South	average			
6	105	Joseph	West	low			
7	106	Karen	North	average			

Criteria in the same column are treated as “or” statements, and criteria in different columns are treated as “and” statements. If the criteria range were `f1:f2`, all records whose *Region* field was “West” would be extracted from the database. If the criteria range were `f1:f3`, all records whose *Region* field matched either “West” **or** “South” would be extracted. If the criteria range were `f1:g2`, all records that matched the criteria of *Region* being “West” **and** the *Sales* field being “high” would be extracted.

For criteria containing formulas, the field must be explicitly referred to in the formula. For example, to extract the records for all employees whose ID numbers are greater than 103, the criteria would read `=A2>103`. The same rules apply for “or” and “and” in rows and columns.

✓ You can have as many rows as you like in a Criteria Range, If you add rows to the range, don’t forget to redefine the range in the Extract Inspector!

See also

Data Ranges *196*
Output Range *199*

Destination Range

The rows which satisfy the Criteria in a query are copied to the Destination Range.

✓ Column headings (Field Names) are *not* copied to the Destination range.

You should specify a Destination range which is wide enough and long enough to hold the maximum amount of rows you will ever retrieve.

If your destination range is too small, you run the risk of working with insufficient data.

✓

See also

Data Ranges 196
Output Range 199

The Query Inspector

The Query Inspector lets you set up Queries which access data in a non-Mesa SQL database, such as Sybase, Oracle or Quickbase.

Mesa 2 has built-in support for these three SQL databases. Other databases may be supported through Mesa Addins.

You need to be familiar with SQL (Structured Query Language) to set up Queries.

Queries can Select data from external databases into the worksheet, or can be used to update those external databases.

Once the data has been pulled into the Mesa sheet, you can manipulate it and report on it just as you can any 'native' data.   You can even perform further analysis using Extract queries if necessary.

Use the Query Setup options to define which database adaptor and table should be queried.

✔ The Run button for any Query can be *Command-dragged* to a ToolHolder or to one of the spots in the Toolbox.

See also

The Query Inspector - more

The following tables explain the items in each of the Query Inspector views.

Item	Description
Server	The server field designates the name of the database server to log onto and send queries to. For **Sybase** connection, enter the word: SYBASE. For **Quickbase** connection, enter the name of the Quickbase database you want to query.

See below for Example entries

Adaptor	The adaptor field designates the database adapter to be used. If this field is left blank, or the name "SYBASE" is used, Mesa will use its built-in Sybase Adapter. Enter "Quickbase" to use the BuiltIn Quickbase Adaptor.
Password	The password that will allow the user access to the server. The password will echo unless you change the default in the Preferences panel.
Prompt	The password is stored un-encoded with the worksheet. If the server password is to remain secure, it should not be kept with the worksheet. If Prompt is checked off, the user will be prompted for the server password the first time the server is accessed.
User	The name of the user. If this is left blank, the user's UNIX name will be used.
Database	If the database that the information is being extracted from is different from the user's default database, enter the new database name here. If a database is selected, Mesa issues a "use" command before it does the query.
Name	Each query has a unique name. An unlimited number of queries can be associated with each worksheet.

Query: Run

Clicking on this button executes the current query. This button may be Command-Dragged into the ToolBox, a Tool Holder, or onto the worksheet itself. Placing a query in a Tool Holder or on the worksheet makes it convenient to execute.

✔ The icon button for any Query can be *Command-dragged* to a Tool Holder or to one of the spots in the Toolbox.

✔ Examples of entries in the Database Setup panel@

For Sybase

Server	SYBASE
Adaptor	Sybase
User	sa
Password	
Database	pubs

For Quickbase

Server	PLAccounts
Adaptor	Quickbase
User	root
Password	
Database	

See also

Select Queries *203*
Insert and Update Queries *204*

Select Queries

This explains the function of the various input fields in a Select query.

Item	Description
Select	The SQL “Select” field. This designates the columns to be selected. If nothing is entered, Mesa selects “*” (all columns).
From	The table that the columns are to be selected from. If the From field is left blank, Mesa selects from the “sysobjects” table.
Order By	The SQL “Order By” field.
Compute	Enter any SQL compute fields here. If you want the compute fields to be placed in the destination range, make sure that Ignore Computed Rows is not checked.
Group By	The SQL “Group By” field.
Having	The SQL “Having” field.
Where	The SQL “Where” field.

See also

Select and Raw Query Options *207*
Insert and Update Queries *204*

Insert, Update and Delete Queries

All these queries affect the original external database.

They should therefore be used with care!

Insert Queries

Item	**Description**
Insert into	Designates the table that the values will be inserted into.
Values	The values to be inserted into the table.
Subquery	An optional subquery.

Update Queries

Item	**Description**
Update	The table that the update statement will act on.
Set	The columns and values to set.
From	An optional from field.
Where	The where clause that limits the update statement so it does not operate on all the records in the table.

Delete Queries

The delete statement has three fields.

Item	**Description**
Delete	The delete clause.
From	The table to remove rows from.
Where	The where clause that limits the delete statement so it does not remove all the records in the table.

See also

Select Queries	*203*
Select and Raw Query Options	*207*
Insert and Update Queries	*204*

Raw Queries

A Raw Query returns data into the worksheet just like a Select query.

The only unique field for Raw SQL is the SQL statement itself which can contain any valid string. This statement will be sent to the SQL server to be executed. If any values are returned, they are placed in the destination range.

See also

Select and Raw Query Options

Select and Raw queries have various options which can be set to control display within the worksheet..

Item	Description
Destination Range	The range of cells or named range in which to place the data resulting from the query.
Set Range Name	This named range label will be set to the range which has received the query information. This is useful for basing reports or calculations on the results of a query.
Include Column Titles	Mesa places the column titles for the selected columns in the first row of the destination range if this item is checked.
Display Errors in Dialogs	Places error messages in a dialog panel.
Place Errors in Range	Places error messages in the current row of the destination range.
Ignore Computed Rows	Only place non-computed rows in the destination range.
Run Before Each Recalc	If this option is selected, Mesa runs the query before each recalculation of the worksheet. This is handy if the data in your database changes frequently or if the query is based on the value of cells in the worksheet. However, running the query may be time consuming. Mesa will run this query before running each data point in a Scenario.
Set Style	This option will set the formatting of normal rows to Style Template 7 and the formatting of calculated rows to

Style Template 8. For more information, see “Style Templates” 179.

Page Break After Computed Rows	This option sets a horizontal page break after computed rows in your query. For more information about page breaks, see “Setting Page Breaks” 185.
Set Height of Rows	This option will set the rows of the destination range to the best row size after the query. This is useful for the formatting of reports — columns can be set to “Wrap” alignment and the text will flow well after a query. For more information on these formatting options, see “Row and Column Size submenu” 157 and “Alignment” 169.

See also

No Help!

Sorry - This helpfile is currently under development.

Sort

Range->Sort... displays a Sort dialog panel that allows you to sort the data on your worksheet.

First, select a cell range on your worksheet, then click on the **Sort...** command. You will see a dialog panel where you can choose to sort by rows or columns, and where you can specify up to three sort keys to be used to sort in either ascending or descending order.

If the first row or column of your range contains titles, click the *Has Titles* box to prevent the titles from being sorted.

Labelling Ranges

Labels are names that are assigned to a rectangular range of cells. Ranges can be designated as *AutoGrow*.

Use the Label Inspector to set up or delete Label names.

To add a range, type in an appropriate name, press *Tab*, enter the cell range for the label, and press ↵. The range will be added to the list.

To change a range, click on the range in the list. The name and cell range will be placed in the text fields. Make the appropriate changes and click on *Update*.

To remove a range from the list, select the range and click *Delete*.

To select a range on the worksheet, select the range name, then click *Goto*.

✓ Labels are also used to keep track of the location of macros. To execute a macro, select the label from the list and click on the *Run:* button.

✓ The *Run* button may be *Command-dragged* to either the ToolBox or to a tool holder in a window

See also

Quick Range Labelling *212*

Quick Labelling

You can use the *Range-> Labels* menu options to set up named ranges quickly.

Insert Label Index displays a list of all labels and their range addresses in a column on the worksheet, starting in the current cell. This is useful if you frequently refer to the ranges that you have labelled.

Label Range To: Right, Left, Top, Bottom. These four commands let you specify the direction of your label range, depending on the label you specify before invoking one of the commands.

If you enter a name in a cell, you can assign it as a range label to any adjacent cell by using the **Range** → **Labels** menu. For example, if you enter *Sales* into a cell and select **Range** → **Labels** → **Label Range to Right** the label Sales will now refer to the adjacent cell on the right. You can also choose *Left, Top,* or *Bottom*.

Delete All Labels removes all the labels from the worksheet.

See also

The Label Inspector *216*

Autogrow Ranges

This feature, applied to a named range, allows the range to grow beyond its defined size when new information is added.

Use the Label Inspector to define a range as Autogrow.

Use the ↵ or ↓ key to move the cell cursor past the bottom row of the range and see that a new row is added to the range each time you press the key. The formula and formatting information is copied from the line above.

AutoGrow ranges are useful for data entry templates, because you don't need to pre-allocate and format a range for data entry. Set the range to AutoGrow and as more data entry space is needed, the range will expand automatically.

See also

The Label Inspector *216*

Constrained Ranges

When you enter a formula containing a range label in your worksheet, Mesa lets you limit (constrain) its effect to specific rows and/or columns of a labeled range.

Let's assume you have a range called REGION (A1:D200) and you want to add up the data in column C.

You can enter a formula as `@SUM(REGION[][2])`, meaning "add up all rows in the third column of the named region." In this case, Mesa row and column numbering begins with 0.

The syntax is: *function*(*range label*[*rows*][*columns*])
Study the examples in the table below, keeping in mind that, if *all* columns in the range are to be included, the second set of brackets can be omitted. For example:

@SUM(REGION[0])
@SUM(REGION[][1,2])
@SUM(REGION[4,9])
@SUM(REGION[29,73][1,2])

The items in the brackets can be either numbers or the following keywords:

Keyword	Action
THIS	Constrains the range to the current row or column
NEXT	Constrains the range to the next row or column
PREV	Constrains the range to the previous row or column
BEGIN	Constrains the range to the first row or column
END	Constrains the range to the last row or column

The above keywords can be mixed with each other and absolute row or column numbers. For example, SUM(SALES[BEGIN,THIS]) computes the sum of the range SALES from the first row through the current row.

See also

Label Inspector

The Label Inspector lets you set up, rename, and delete range names on your worksheet.

To add a range, type in an appropriate name, press *Tab*, enter the cell range for the label, and press ↵. The range will be added to the list.

To change a range, click on the range in the list. The name and cell range will be placed in the text fields. Make the appropriate changes and click on *Update*.

To remove a range from the list, select the range and click *Delete*.

To select a range on the worksheet, select the range name, then click *Goto*.

Scenarios

The Scenario Inspector creates tables which take values for one or two variables, and the result of each value or set of values in a given formula.

Mesa scenarios allow you to use your spreadsheet to make calculations based on more than a single formula and set of variables. For example, you might want to calculate the monthly payments and total payments on a loan for payment plans with different interest rates. You could also use scenarios to find the orbital angular momentum of a body given different theoretical values of its semi major axis and orbital period. Scenarios are especially useful when you are concerned with a set of predicted or theoretical values and varying conditions.

Mesa has two types of scenarios.

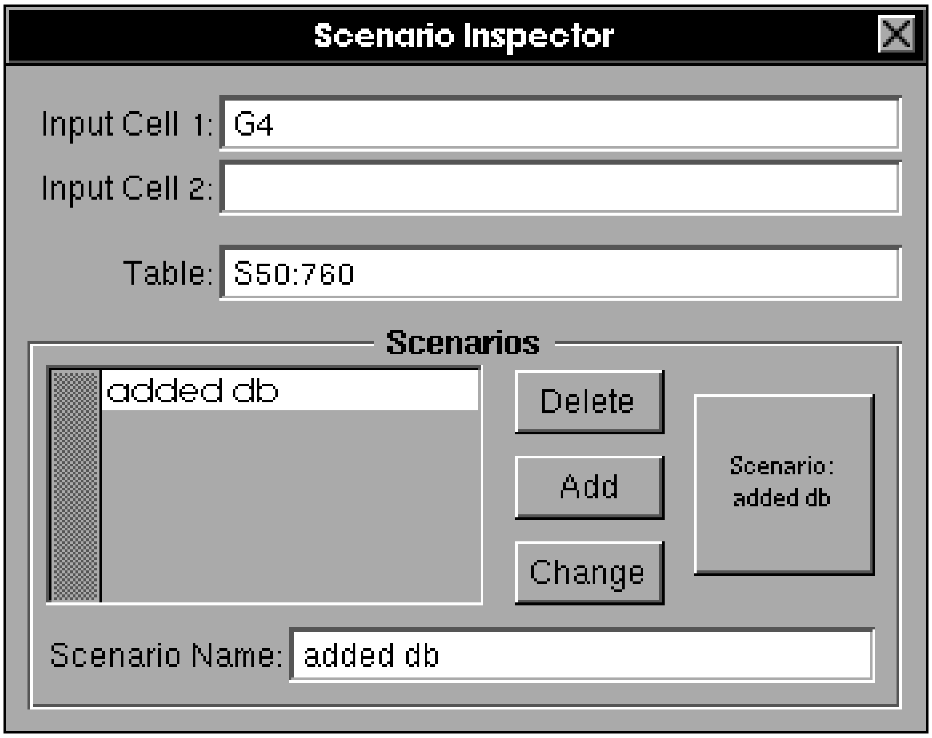
Single Input scenarios show the result of changing the value of a single variable in different formulas.

Double Input scenarios compute the result of a single two-variable formula for different values of each variable.

Both types of scenarios are created using the Scenario Inspector.

The Scenario Inspector creates a scenario table. A single input scenario will create a table which shows the result of using each different value of the variable in each formula. Double input scenario tables show the result of each different set of variables in the equation.

Using the Scenario Inspector



Mesa scenarios are calculated using the Scenario Inspector. To open the Scenario Inspector select **Sheet** → **Database** → **Scenario Inspector...** in the menu. Give the Scenario a name by typing it into the space at the bottom. Enter the cell name for the input cell or cells, and the range of the scenario table. Once the information is entered, click on the *Add* button. The name of the Scenario will appear in the list in the middle of the Scenario Inspector. You can remove, change, or run a Scenario by clicking on its name in the list, which will highlight it, and then clicking the appropriate button.

Mesa runs all ''Run before each recalc'' Sybase queries before running each data point in a scenario. Scenarios recalculate the worksheet after they execute, even if auto-recalc is off.

Creating Single Input Scenarios

To create a single input scenario, choose a column to enter the different values of the variable into. Leave the first cell of the column blank and enter the values starting with the second cell of the column. For example, if you wished to make a scenario using ten different values of a variable starting in the upper left corner of the spreadsheet you would leave cell A1 blank and enter the values in cells A2 through A11.

Next, enter the your formulas in the row corresponding to the blank cell at the top of the data column, and starting in the next column over. In our example, if you had five formulas you would enter them in cells B1, C1, D1, E1, and F1. When entering the formulas, refer to the variable using a cell name. This cell is the input cell, and can be any cell on the spreadsheet that is not part of the scenario table and does not contain a formula. You can also use the blank cell at the top of the input column. For example, you might enter the formula `=A1^2` in cell B1 to find the square of the values in column A.

Note that as a result of Mesa's normal formula computation, the cell you enter the formula in will display the value of the formula with the variable equal to the contents of the input cell.

Once you have entered the variable values and the formulas, use the Scenario Inspector to run the scenario. The Scenario Inspector will create a table with the values of the formulas computed for each of the variable's values in the appropriate cell. For example, cell D9 would contain the value of the formula in column D computed with the variable value in row 9.

The table is a range on the worksheet that includes the row and column with the input values and formulas. In our example above the range of the scenario would be A1:F11.

Creating Double Input Scenarios

To create a double input scenario, choose a column to enter one set of variables into, and a row to enter the other set of variable. The formula is entered in the cell where the two intersect. For example, if you wanted to place your scenario in the upper left corner of the spreadsheet, you would enter one set of variables in column A starting with cell A2, and one set of variables in row 1 starting with cell B1. The formula would be entered into cell A1. If there were ten values of each variable, the range of this scenario table would be A1:K11. When the scenario is run, a table will be created in the range with the value of the formula using each pair or input values in the appropriate cell. For example, cell C6 would contain the value of the formula using the input values in column C and row 6.

When you enter the formula into the formula cell, refer to the variables using two input cells. The two input cells may be any two cells on the worksheet that are not part of the scenario range and do not contain formulas. As long as these two conditions are met, the input cells can be anywhere on the sheet, and do not have to be blank cells. Once you have entered both the input values and the formula use the Scenario Inspector to run the scenario.

See also

Matrix manipulation

The Matrix submenu contains Mesa's matrix manipulation commands. These commands can be used to perform various operations on matrices.

Regress performs linear regression on the input range. Place your matrix numbers in a range on the worksheet with the x values in the leftmost columns, and the y value in the second to last column, with empty column next to the y values for the output. Highlight the range, and select the command. The results of the linear regression will be calculated and placed in the right-hand column.

Invert will calculate the inverse of a matrix. Highlight the matrix that you wish to find the inverse of, and a range of equal size immediately to the right of it. When you execute the invert command, the inverse of the matrix will be placed in the empty cells to the right of the original matrix. Only square matrices may be inverted.

Creating a Chart

First, select the range oin the worksheet that you want to chart. If the range includes labels at the left and the top, then these labels will be used as the X and Y axis labels on the chart.

Then click the Chart Icon on the Worksheet iconbar (or use select *Tools->Chart* from the menu).

The Chart Selection Panel will appear. There are three main Chart Types (Bar, Line and Area). Within each Type there are several variations: for example a Bar Chart can be Stacked, Horizontal, 3D and more.

Click on the Type you want. The Chart will draw itself on the worksheet at the current cursor position. The Chart Inspector will also appear.

Use the Chart Inspector to change and customize the chart.

✔ You can have as many charts as you like on the worksheet

✔ You can move and resize charts.

Chart Types

Once you have created a chart using the Chart Selection Tool, you can change it from one graph type to another by using the 'Chart' aspect of the Chart Inspector. Double clicking on a graph will open the Chart Inspector.

Each type of chart in Mesa can be modified in a number of ways using the Inspector. You can add titles labels, and legends, and change the orientation and color of the graphs.

In addition, some graphs have a *Current* view on the graph inspector that allows you to change the graph in a way specific to that graph type. For more information about graph titles, orientation, color, etc. see “Chart Inspector Commands” *226*.

The graph data is set to “Left Column has Titles” off, and Horizontal data orientation. To set these parameters, use the Chart Inspector. To open the Chart Inspector, you can double-click on a graph .

You must have the chart that you want to inspect selected in order to see the Chart Inspector.

To remove a chart or graphic from the worksheet, select it and press the ← key or select **Edit** → **Cut**.

Bar and Column Charts

This view allows you to determine the width of the bar or column by moving the slider to select the desired width. The line option will plot a line (the first series of data) on the same graph. If the orientation of the graph is horizontal, the line is plotted from the data in the left column. If the orientation of the graph is vertical, the line is plotted from the data in the top row.

Scatter Charts

Scatter charts can be drawn with data points, lines, or both. Select the drawing method using the radio buttons. Use the slider to adjust the line width.

XY Charts

In addition to the chart options for lines and data points that Scatter Charts allow, XY charts allow you to select whether the Y values of the plot will be matched with a single set of X values or if they will be matched with the X values of the adjoining row or column. The default is a single X axis.

High/Low/Close Charts
High/Low/Close charts have an option for displaying candlestick graphs. This option is chosen by default.

3D Charts
3D charts have sliders for adjusting the X and Y rotation of the graph and a wire frame option.

Sample Data: The following examples of chart use this data, unless otherwise noted:

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug
200	400	600	800	1000	1200	1400	1600

See also

Chart Inspector

Once you have created a chart using the Chart Selection Tool, you can use the Chart Inspector to tailor the chart to your precise requirements.

Switch between the various aspects of the Chart by using the PopUp List on the Inspector.   You can change Chart Type, add Titles, change the ranges in the Chart, change Colors, add and move the Legend,and more.

In addition, some charts have a *Current* view on the inspector that allows you to change the chart in a way specific to that chart type.

The Inspector allows you to invoke special features in Mesa's graphs, including graphing discontiguous ranges, adding labels, and creating mixed bar and line charts. The different views of the Inspector allow you to adapt different parts of your chart to suit your needs.

✔ Double clicking on a chart is another way to open the Chart Inspector.

✔ The Chart Inspector panel changes to say 'No Chart Selected' when a chart is no longer selected and reappears when the graph is selected again.

Chart Inspector commands

From the three Popup menus on the Chart Inspector, you can inspect and change the following aspects of the currently selected chart.

Name

Each chart has a unique name. You select a chart by name using **Range** → **Goto** (*Command-g*). You can also print a chart by name in a report. To change the name of a chart, select the name, be sure that the Graph Inspector is selected (i.e., the title bar is black not light gray), type the new name, and press ↵. You cannot give a chart the same name as another chart in the worksheet. If you try to, Mesa will beep and the name will not change.

Color

The *Color* view allows the background color, border color, legend background color, and 3D background color to be changed. If the “Clear” check box is checked, the background below the graph will show through. This is especially useful for developing presentations using Concurrence(tm) or Create(tm). The width of the border can be changed using the slider below the border color well.

To change the color or pattern of the plotted part of the graph you do not need to use the Graph Inspector. Simply drag and drop color chips onto the graph itself. Mesa will keep color shading and patterns consistent for data that is grouped together.

Labels

The different areas of the on the *Label* view if the inspector allow you to change the appearance of the graph’s labels. Labels are placed near the horizontal and vertical axis of a graph.

You can adjust the rotation of the labels using the *Label Rotation* area. Rotating labels enhances the visual appearance of the chart and can be used to display labels that would otherwise overlap. Labels may be rotated using the sliders or by typing the desired rotation in the text fields and pressing ↵.

The check boxes in the *Label Location* area allow you to designate whether the top row and/or the left column contains additional labels. You can also designate ranges containing additional labels using the *Range* view of the Graph Inspector. Flat graphs can display one set of additional labels, and 3D graphs can display two sets of additional labels.

The font wells in the *Fonts* area of the view display the current font used to draw the labels on the X and Y axis. To change the font, open the Font Panel by selection **Format** → **Fonts...** (*Command-t*) or by double-clicking in the font well. Build the desired font and drop the font in the appropriate font well. The fonts of the X and Y axis can also be changed by dropping a font chip on the X or Y axis or any label of the chart.

The *Display Format* area of the view allows you to change the format and precision of the axis labels. You can select any of the Mesa's standard format, including date formats. This will not affect the format of additional labels.

The *Label Position* area allows you to select the vertical centering of labels drawn on the Y axis. Select the setting that you prefer.

Titles

Charts can display titles on the top, bottom and left side. The *First* and *Second* titles are drawn at the top of the chart. The *X Axis* title is drawn under the X axis. The *Y Axis* title is drawn to the left of the Y axis and is rotated 90 degrees. Enter the text to appear in the titles. The font of each title can be changed by dropping a font chip in the font well or by dropping a font chip on the title of the chart. If *Cell Reference* is checked, the text is interpreted as an address and the contents of the cell at that address is displayed.

Orientation

The *Orientation* defines the order that the data will be accessed when drawing the chart. Horizontal orientation reads data ''across'' and additional data ranges are added under the base range. Data in a column will show up with the same color and pattern. Vertical orientation reads data ''down'' and additional ranges are added on the right side. Data in the same row is given the same color and pattern. Mesa tries to choose the correct orientation for your data when you create a graph.

Ticks

This view allows you to set how the format in which ticks on the X and Y axis will be drawn.

Scale

By default, Mesa automatically chooses the largest and smallest X and Y values for data plots. You can change these values manually to enhance the appearance of a chart. To turn off automatic scaling, click the *Auto Scale* check box. Enter the desired scale values in the text fields and press ↵. The minimum and maximum values must be different.

Graphs are displayed in linear scale by default, but can be displayed in log scale by clicking the “Log Scale” check box.

Grid

This view allows you to control the display of a light gray grid in the background of the chart. You can also select an optional black border around the graph area on the chart.

Range

You can add up to six discontiguous ranges to the base range of a data plot. Additional ranges will be added at the bottom or right side of the prior data range depending on the orientation of the chart. (See “Orientation” .) To add ranges, type the range into each of the six provided text fields.

In addition you can also add ranges containing titles to your graph using the *Top Titles* and *Side Titles* fields. The contents of these ranges will be added as additional labels to the graph the same way that selecting *Top Row* or *Left Column* in the *Label Location* area of the Label view would, except that the ranges entered as titles in this range do not need to be in a specific location in relation to the rest of the data.

Legend

This view allows you to place a data legend on the chart. The legend can be located on top, below, or on either side of the chart. The legend’s font can be changed by dropping a font chip in the font well or by dropping a font chip directly on the legend on the chart.

The color of the chart can be changed by dropping color chips on the legend or by dropping the color chips on the item that you want to change directly on the graph.

Current

Some chart types have special inspector views to change unique parameters. These views appear in the *Current* view depending on the chart type.

✔ You can have as many charts as you like on the worksheet

✔ You can move and resize charts.

Arrange

Arrange->Layer->**Send to Back**

Use this command if you have two or more charts on your worksheet at the same time to control which one is in full view. To move one chart behind another, click its frame to select it and then click the **Send to Back** command. The selected chart will be placed behind any other charts on the worksheet. This can be invoked with *Command--* (*Command-minus*).

Arrange->Layer->**Bring to Front**

This is the counterpart of the Send to Back command described above. To bring a chart to the front, click its frame to select it and then click the **Bring to Front** command or use *Command-+*.

See Also

222 Creating a new Chart

Shapes

You can pull Shapes onto the worksheet with the Graphics Shapes tool.

Use Tools->Graphics to bring up the Graphics panel, and select Shapes from the Popup list.

Drag any shape onto the area you want to affect.

If you double-click a Shape on the worksheet, you can change it’s color and border with the Shape Inspector.

✔ Rich Text is the only Shape that you can type text into. It’s the most useful shape foir this reason. Use it to add comments to your Charts or worksheets.

✔ You can Command-Drag any Shape into the ToolBox..

Patterns

You can pull Patterns onto cells or Graphs with the Patterns tool.

Use Tools->Graphics to bring up the Graphics panel, and select Patterns form the Popup list.

Drag any pattern onto the area you want to affect.

✔ Patterns only show up on cells or graphic objects which have a colored background.

✔ You can Command-Drag any pattern into the ToolBox..

Graphic Images

You can put Graphic objects onto the Mesa sheet and into Report Layouts.

Drag any TIFF or EPS file icon from the Workspace onto your Mesa sheet: it wil immediately ’stick’ on to the sheet.

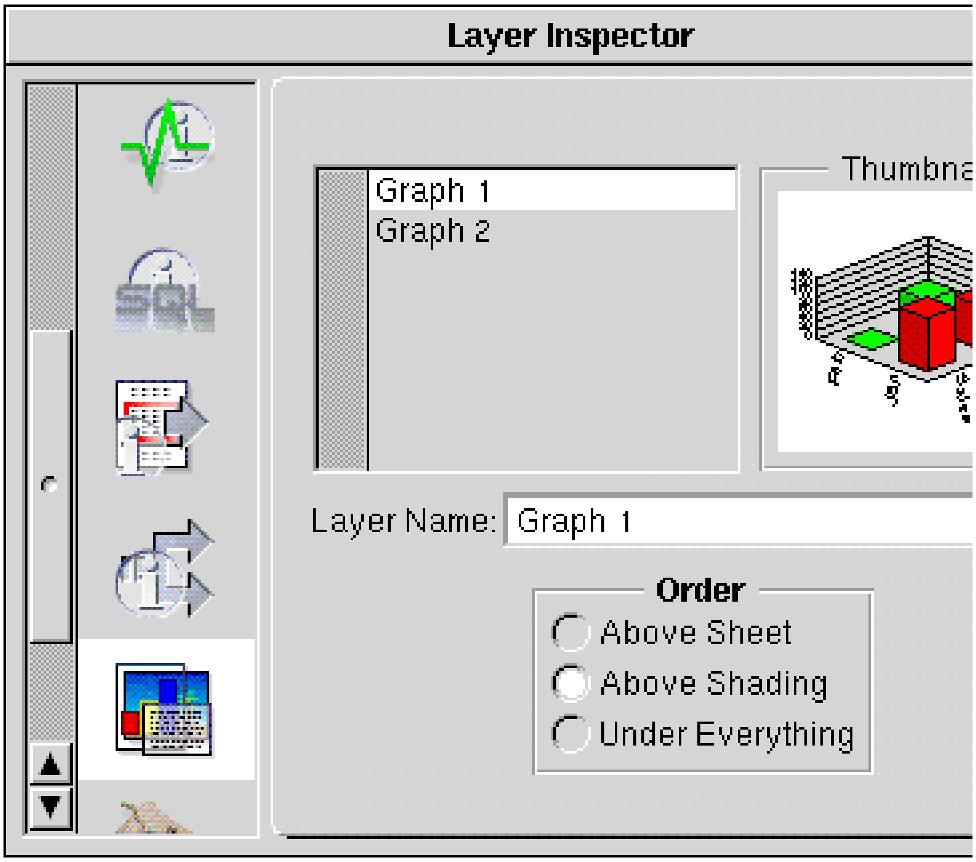
Graphic objects can be selected and moved just like the native Mesa Charts and Shapes.

✔ To put a Graphic image on a Report, first drag a Graphic Tag from the edge of the Report Layout onto the correct position on the page. Then drag a TIFF or EPS icon from the workspace onto that Graphic Tag. You’ll see the cursor turn green when you’ve reached the corretc ’dropping’ position.

Graphics on Reports can also be moved and resized.

Layer Inspector

The Layer Inspector allows you to look at Graphic objects (Graphs, Shapes and Icons) on the worksheet, and adjust their relative positions.



All Graphic objects can be moved 'up' and 'down' relative to other objects, selected areas on the sheet (*shading*) and the cell gridlines (*sheet*).

You cannot change the name of a Graphic object in this Inspector.

Graphic images in Reports

To put a Graphic image on a Report, first drag a Graphic Tag from the edge of the Report Layout onto the correct position on the page.

Then drag a TIFF or EPS icon from the workspace onto that Graphic Tag. You’ll see the cursor turn green when you’ve reached the correct ’dropping’ position.

Graphics on Reports can also be moved and resized

CoWorking Techniques

If you share worksheets with other users, or if you create worksheets which you intend to give to other people to use, then you can do quite a lot to make your and other users lives easier. Mesa allows you to:

- Put passwords on your sheet so that only authorised user can open them

- Protect cells on your sheet so that no-one can type into them (useful for stopping yourself or others from typing over formulas by mistake).

- Give meaningful names to parts of the worksheet and use those names in Reports, Formulas and GOTOs

- Designate cells on your worksheet to accept only particular types of input.

- Delete data only from cells containing designated data types.

- Design report layouts which ensure that all users will produce the same well-presented printouts.

All this without using Macros at all.

See also

Protecting Worksheets

Within your worksheet, you can designate protected and unprotected cells. Typically you'll do this if you have a sheet where you don't want users to overwrite formulas and headings by mistake.

When you create a new worksheet, protection is off by default. After you have built a structure that requires some ranges to be protected and others open for data entry, you should do the following:

In the **Sheet Inspector** select the ''Protection'' aspect and click the *Protection* button to display the check mark. The entire worksheet is now protected.

Next, highlight a range you want to unprotect. In the Cell Style Inspector select the ''Input Type'' aspect. Select the *Unprot'd* radio button.

The range you selected is now unprotected. This is now the only area in which you are allowed to type.

✔ You can mark lots of different areas as Unprotected.

✔ Anyone using the sheet can turn off the Sheet Protection. Setting protection is not a guarantee that no-one will ever delete your formulas or overwite constant data.

See also

Controlling data input
Range names 239
Using TAB to skip non-protected calls

Clear Special

Use Edit->Clear Special to clear various aspects of a cell or range of cells.

Format clears all formatting options from the cell or cell range that you selected, returning it to Base Style 1.

Data clears all data from the cell or cell range that you selected, leaving only the formatting options.

Numeric Input clears all cells containing numeric data in the selected cell range.

✓ This is useful for worksheets that are used repeatedly, and must be cleared of all prior numeric content before each use. Formatting options, formulas, and string are not affected by this command.

String Input clears only cells containing string data in the selected cell range.

✓ This is useful for reusing a worksheet that needs to have only its captions updated. Formatting options, formulas, and number are not affected by this command.

Reset clears the values of cells containing formulas. This is useful when working with cells containing CURRENTVALUE and CURRENTNUMBER constants.

See also

Protecting cells
Range names

Using Range Labels

You can give English-like names to cells or ranges of cells.

You can then GOTO or Select the area by referring to it's name rather than having to remember the cell references. This makes it easier for both you and other users of your worksheet to navigate around the sheet.

Create, change and Use Range Labels in the Label Inspector (see below for more information)

✓ Use Range Labels in any situation where you would otherwise use cell-addresses.

- in the ReportWriter
- in Formulas
- for Copying

See also

The Label Inspector *240*

The Label Inspector

The Label Inspector displays an inspector panel that lists all the range labels you have assigned in your worksheet. The inspector panel allows you to Goto, Delete, Run a Macro for, or Update any of the labels on the list.

To add a new label, highlight a range on the worksheet and then open the Label Inspector. The new range address will be displayed and the name window will be blank. Type in the name of the new label and click *Add* or press ↵. The label of the new range will be added to the list.

Insert Label Index displays a list of all labels and their range addresses in a column on the worksheet, starting in the current cell. This is useful if you frequently refer to the ranges that you have labelled.

Label Range To: Right, Left, Top, Bottom. These four commands let you specify the direction of your label range, depending on the label you specify before invoking one of the commands.

Delete All Labels removes all the label names from the worksheet.

✓ If you enter a name in a cell, you can assign it as a range label to any adjacent cell by using the **Range** → **Labels** menu. For example, if you enter *Sales* into a cell and select **Range** → **Labels** → **Label Range to Right** the label Sales will now refer to the adjacent cell on the right. You can also choose *Left, Top,* or *Bottom*.

See also

Range names 239

The Language

MScript is a scripting language designed for the Mesa user. It is a very simple and straight-forward language that is similar to BASIC. In short, it was designed for ease of use by people who aren't programmers. MScript is ideal for writing simple scripts and macros, but AddIns or MOLI should be used for more complex tasks. See the *Mesa Programmer's Guide* for more information about AddIns and MOLI.

See also

Creating and Editing a Script

MScript scripts can be created and edited using the MScript Inspector. The MScript Inspector can be opened with the **Tools** → **MScript Inspector...** command.

Creating a Script

You can type MScript commands directly into the editing window of the MScript Inspector. For example, type:
`a1 := "Hello World"`

A script needs a name before it can be added and run. Type `MyScript` into the field marked “Script Name:” and click the *Add* button. Choose *MyScript* from the browser. Click on the *Script:MyScript* button to execute your script.

The *Script:MyScript* button can be *Command-dragged* to one of the Tool Holders on your worksheet or onto the worksheet itself, if you so desire.

Editing a Script

To change a script, select the appropriate script name from the browser and make your changes in the editing window. When you’ve finished your editing, click on the *Change* button and run your script as normal.

See also

MScript - The Language *241*
Errors *243*
Basic Operations *244*
Functions *254*

Errors

If there's an error in your script, a message will appear below the editing window, and the line containing the error will be highlighted.

See also

Basic Operations

Each statement in an MScript script must be on its own line.
Comments can be placed in scripts by starting the line with a `%`. Comments can be useful for making your scripts more readable and more easily maintainable by others and by the author at a later date.

```
% This line is a comment and won't affect
% the script.

a1 := "Hello World"

% The above line of code will put the string
% "Hello
%      World" in the cell A1.
```

See also

Manipulating Values

The values in cells can be set easily from an MScript script. Numbers, strings, cell references, range labels, and formulas must be treated differently.

```
% This will set the value of A1 to the
% number 10.
a1 := 10

% This will set the value of A1 to the
% string "Hello"
% Note that strings need to be surrounded by
% quotes.
a1 := "Hello"

% This will set the value of A1 to the value
%       of B1
a1 := b1

% This will set the value of A1 to the value
%       of the range label "mylabel". If
there
%       is no range labelled "mylabel", an
error
%       will occur.
a1 := mylabel

% This will set the input string of A1 to
%       the formula "=1+2"
% Note that strings that should be
%       interpreted as if they were typed in
by
%       a user must be treated in this
manner.
setinput(a1,"=1+2")
```

For more information about **setinput()**, see “SETINPUT” *293*.

Variables

Variables can be used to facilitate the manipulation of information within an MScript script. Variable names must be preceded by a `?`, and can contain upper- and lowercase letters (MScript variables are case-insensitive), numbers, or the symbols “_”, “`$`”, “`?`” and “`.`”. Examples of valid MScript variable names include `?x`, `?value37`, `?cat`, `?tax_rate`, and `?overtime.hours`.

Variables can have string or number values, cell references, or labels assigned to them, and can be used in the place of those values or references. Values can be assigned to variables with the `LET()` function or with the `:=` operator. For example:

```
% This sets the variable ?x to the value 10
LET("?x",10)

% This does the same thing.
?x := 10

% This sets the variable ?z to the string
%       "hello"
?z := "hello"
```

Variables can also be set to the values contained in a cell. When a variable is set to the value of a cell, the variable's value will change as the value of the cell changes. In order to set a variable to the current value of the cell (such that its value will not change), use the **freeze()** function. For example:

```
% This sets the variable ?y to the value in
%       cell A1
% If the value in cell A1 changes, the value
%       of ?y will
%       change as well.
?y := a1

% This sets the variable ?y to the current
%       value in the
%       cell A1
% The value of ?y will not change if the
%       value in cell A1
%       changes
?y := freeze(a1)
```

Conditionals

IF-ELSE statements

One way to execute code conditionally is with an IF/ELSE statement. If the expression following the IF statement is true, the next line will executed. Otherwise the next statement is executed. The ELSE lines are skipped if the IF expression was true. In the example below, since the condition is true, the next line is executed.

```
IF (1 = 1)
a1 := "One does in fact equal One"
ELSE
a1 := "Whoops..."
```

If you want to execute more than one function in an IF/ELSE statement, lines can be grouped after the IF and ELSE statements with BEGIN and END statements.

```
IF (1 = 1)
       BEGIN
       a1 := "One does in fact equal One"
       a2 := "And MScript tells us so!"
       END
ELSE
       BEGIN
       a1 := "Whoops..."
       a2 := "Something’s wrong!"
       END
```

Conditionals can be used to check values and execute specific functions. More complex conditions can be checked using operators like the ones listed in the following table.

Conditional Operators

The following conditional statements are some simple examples:

```
if (a1 > 0 AND a1 < 10)
b1 := "A1 is between 0 and 10"

if (a2 < 10 OR a2 > 20)
b2 := "b1 is either less than 10 or greater
than 20"

if (a3 = 10 XOR a4 > 10)
b3 := "Either a3 is equal to 10 or a4 is
greater than 10, but not both."
```

WHILE statements

Another way to execute code conditionally is with a WHILE statement. A WHILE statement will execute the line following it as long as the expression following the WHILE keyword is true. In the following example, the value in cell b1 will be set to 10.

```
% make ?x equal to 1
?x := 1

% As long as ?x is less than 10, keep adding
%       1 to it.
WHILE ?x < 10
?x := ?x + 1

% set the value of the cell b1 to the value
%       of ?x
b1 := ?x
```

Lines after a WHILE statement can also be grouped with BEGIN and END.

Looping and Branching

Looping and branching can be accomplished with a GOTO statement. "GOTO *label*" passes control unconditionally to another section of the script, marked by a *label*. Labels begin with a colon (:).

```
a1 := "This is an example of branching."

GOTO "mylabel"

a2 := "Program control never gets to this
line."

:mylabel
a3 := "...but it does get here."
```

Looping can be accomplished with the use of a conditional and a GOTO. The example below will put the string "This is an example of looping" in cells A1:A20.

```
% set ?x to 0
?x := 0

% set a label
:mylabel
% Set the string value of the cell at row
%       ?x, column 0
address(?x,0) := "This is an example of
looping."
% add 1 to ?x
?x := ?x + 1

% As long as x is less than 20, do it all
%       again.
if ?x < 20
       GOTO "mylabel"
```

For more information about the **address()** function, see "ADDRESS or @ADDRESS" .

Subroutines

Subroutines can be used to divide a large script up into smaller parts for the sake of readability and functionality. They can also be used to perform small tasks that have to be repeated numerous times.

Subroutines are called with “GOSUB *label*”. GOSUB is similar to GOTO, but will return control to the original location in the script upon reaching a RETURN statement.

```
% Check if A1 is 1, 2, or 3
if (a1 = 1 or a1 = 2 or a1 = 3)
begin
        % Concatenate a string for the
appropriate
        % subroutine name
        gosub "subroutine" & fixed(a1,0)
        % Set cell B1 to the message set by
the subroutine
        b1 := ?message
end
else
begin
        % Set cell B1 to an error message
telling the user
        % that they didn’t pick one of our
subroutines
        b1 := "Not a valid subroutine."
end

:subroutine1
?message := "This is subroutine #1"
return

:subroutine2
?message := "This is subroutine #2"
return

:subroutine3
?message := "This is subroutine #3"
```

For more information about the **fixed()** function, see “FIXED or @STRING” .

Getting user input

MScript provides three different ways to get input from the user. The user can be prompted to type something, can be offered a menu with choices, or can be given an alert panel with buttons.

Prompting for input

The user can be prompted for input using the **prompt()** function. The **prompt()** function will cause a window to appear on the screen displaying the specified text, and will return the string typed in by the user.

```
% This will set the value of A1 to the
string typed in by
%       the user.
a1 := prompt("Type in any string")
```

For more information on the **prompt()** function, see "PROMPT" *291*).

Menus

The user can be prompted to make a choice from a menu using the **menu()** command. For a detailed description, see "MENU"*290*.

If the worksheet contained the following information:

```
        A
          B                 C
1       One
          The First Option  subroutine1
2       Two
          The Second Option subroutine2
3       Three  The Third Option  subroutine3
```

then the following code would create a menu of the options and execute the appropriate subroutines.

```
?foo := menu(a1:c3,menubyrow)
IF (?foo != -1)
        BEGIN
        gosub ?foo
        e1 := ?message
        END
ELSE
        e1 := "No choice was made"

:subroutine1
?message := "This is subroutine #1"
return
```

```
:subroutine2
?message := "This is subroutine #2"
return

:subroutine3
?message := "This is subroutine #3"
```

Alert Panels
The user can also be prompted for input via an Alert panel with buttons. For more information about the **alert()** function, see “ALERT”288.

```
?foo := alert("Options","Choose one.",
"One","Two", "Three")
IF (?foo != -1)
BEGIN
       gosub "subroutine" & fixed(?foo+1,0)
       e1 := ?message
END
ELSE
       e1 := "No choice was made"

:subroutine1
?message := "This is subroutine #1"
return

:subroutine2
?message := "This is subroutine #2"
return

:subroutine3
?message := "This is subroutine #3"
```

245 Manipulating Values
247 Conditionals
249 Looping and Branching
250 Subroutines
254 Functions

Functions

MScript can use all Mesa functions, although some functions take different arguments in the scripting language. The following sections describe these differences as well as additional MScript-specific functions.

For more information about standard Mesa formulas, see ‘‘Entering Formulas’’ *67*

File Operations

NAMESHEET

Names the current worksheet.

NAMESHEET(*name*)

name the new name of the worksheet

This function changes the name of the current worksheet to *name*. The name *name* must be a string.

Returns: *the name of the sheet if success, -1 if failure*

```
namesheet("MySheet")
```

SAVE

Saves the current worksheet.

SAVE([*name*])

name the [optional] new name of the worksheet

This function saves the current worksheet. If *name* is specified, the worksheet's name is changed before the save. The name *name* must be a string.

Returns: *the full pathname of the sheet if success, -1 if failure*

```
save()
```

```
save("NewName")
```

SETSHEETPROTECTION

Turns sheet protection on or off.

SETSHEETPROTECTION(TRUE|FALSE)

Editing Operations

CLR or **CLEAR**

Clears the specified cell or range.

CLR(*cell*) or CLR(*range*)

cell	any cell reference
range	any worksheet range

This function clears all the information from the *cell* or *range* specified, including both data and formatting information.

Returns: *cell or range*

```
clear(a1)

clear(a1:b7)

clear(myrange)
```

CLRDATA or **CLEARDATA**

Clears any data from the specified cell or range.

CLRDATA(*cell*) or CLRDATA(*range*)

cell	any cell reference
range	any worksheet range

This function clears the data (but not the formatting information) from *cell* or *range*.

Returns: *cell or range*

```
clrdata(a1)

clrdata(a1:b7)

clrdata(myrange)
```

CLRFMT, **CLEARFMT** or **CLEARFORMAT**

Clears any formatting information from the specified cell or range.

CLRFMT(*cell*) or CLRFMT(*range*)

cell any cell reference
range any worksheet range

This function clears formatting information (but not the data) from *cell* or *range*.

Returns: *cell or range*

```
clrfmt(a1)

clrfmt(a1:b7)

clrfmt(myrange)
```

CLRNUM or **CLEARNUM**

Clears numerical data from the specified cell or range.

CLRNUM(*cell*) or CLRNUM(*range*)

cell any cell reference
range any worksheet range

This function clears the numerical input from *cell* or *range*, but not strings, formulas, or formatting information.

Returns: *cell or range*

```
clrnum(a1)

clrnum(a1:b7)

clrnum(myrange)
```

CLRSTR or **CLEARSTR**

Clears string data from the specified cell or range.

CLRSTR(*cell*) or CLRSTR(*range*)

cell any cell reference
range any worksheet range

This function clears any string data from *cell* or *range*, but not numerical input, formulas, or formatting information. It returns *cell* or *range*.

Returns: *cell or range*

```
clrstr(a1)
```

```
clrstr(a1:b7)

clrstr(myrange)
```

COPY

Copies a cell or range from one location to another.

COPY(*source*,*dest*)

source	any cell reference or worksheet range
dest	any cell reference or worksheet range

The COPY function copies the information from *source* into *dest*. If a cell reference is given for *dest*, it will be used as the upper left hand corner of the destination range. Any information in the destination range will be overwritten.

Returns: *dest*

```
copy(a1:b7,d12)

copy(myrange,d12:e14)
```

COPYFMT or **COPYFORMAT**

Copies formatting information of a cell or range from one location to another.

COPYFMT(*source*,*dest*)

source	any cell reference or worksheet range
dest	any cell reference or worksheet range

This function copies formatting information only (but not the values) from *source* into *dest*. If a cell reference is given for *dest*, it will be used as the upper left hand corner of the destination range. Any information in the destination range will be overwritten.

Returns: *dest*

```
copyfmt(a1:b7,d12)

copyfmt(myrange,d12:e14)
```

COPYVAL

Copies the values of a cell or range from one location to another.

COPYVAL(*source*,*dest*)

source	any cell reference or worksheet range
dest	any cell reference or worksheet range

This function copies the values only (but not the formatting) from *source* into *dest*. If a cell reference is given for *dest*, it will be used as the upper left hand corner of the destination range. Any information in the destination range will be overwritten.

Returns: *dest*

```
copyval(a1:b7,d12)
```

```
copyval(myrange,d12:e14)
```

FORMTOVAL or FORMULASTOVAL

Converts formulas to their current values.

FORMTOVAL(*cell*) or FORMTOVAL(*range*)

cell	any cell reference
range	any worksheet range

This function converts any formulas in the *cell* or *range* to their current values. Any cell references within the formulas will be converted to the values of the cells referenced, and the values of the formulas will be calculated.

Returns: *cell or range*

```
formtoval(a1)
```

```
formtoval(a1:b7)
```

```
formtoval(myrange)
```

LOWER

Converts strings in a cell or range to lower case.

LOWER(*cell*) or LOWER(*range*)

cell	any cell reference
range	any worksheet range

Converts the strings in the *cell* or *range* to lower case, regardless of their original case.

Returns: *cell or range*

```
lower(a1)

lower(a1:b7)

lower(myrange)
```

MOVE

Moves the information in a cell or range.

MOVE(*source*,*dest*)

source	any cell reference or worksheet range
dest	any cell reference or worksheet range

Moves the information from *source* into *dest*. If a cell reference is given for *dest*, it will be used as the upper left hand corner of the destination range. Any information in the destination range will be overwritten.

Returns: *dest*

```
move(a1:b7,d12)

move(myrange,d12:e14)
```

NUMTOSTR

Converts numbers in the specified cell or range to strings.

NUMTOSTR(*cell*) or NUMTOSTR(*range*)

cell	any cell reference
range	any worksheet range

This function any converts numerical values in *cell* or *range* to strings.

Returns: *cell or range*

```
lower(a1)

lower(a1:b7)

lower(myrange)
```

PROPER

Converts the strings in the specified cell or range to Proper Case.

PROPER(*cell*) or PROPER(*range*)

cell	any cell reference
range	any worksheet range

This function converts the strings in the *cell* or *range* to Proper Case, regardless of their original case. The first letter of each word is capitalized, and all remaining letters are converted to lower case.

Returns: *cell or range*

```
proper(a1)

proper(a1:b7)

proper(myrange)
```

SELECT

Selects a cell or range in the worksheet.

SELECT(*cell*) or SELECT(*range* [, *selectedcell*])

cell	any cell reference
range	any worksheet range
selectedcell	an [optional] cell reference within

This function highlights *cell* or *range*. If *selectedcell* is specified and is within *range*, that cell is selected; otherwise, the upper left cell is selected.

Returns: *cell or range*

```
select(b17)

select(a10:g19,b13)
```

SELECTALL

Selects all cells in the sheet.

SELECTALL([*selectedcell*])

selectedcell	an [optional] cell reference

This function highlights the entire sheet. If *selectedcell* is specified, that cell is selected; otherwise, the upper left cell (A1) is selected.

Returns: *the range reference of the selected area*

```
selectall()

selectall(g29)
```

STRTONUM

Converts strings in the specified cell or range to numbers.

STRTONUM(*cell*) or STRTONUM(*range*)

cell	any cell reference
range	any worksheet range

This function converts any strings in *cell* or *range* to their numerical equivalent.

Returns: *cell or range*

```
strtonum(a1)

strtonum(a1:b7)

strtonum(myrange)
```

TRANSPOSE

Transposes the data from a range, switching rows and columns.

TRANSPOSE(*source*,*dest*)

source	any cell reference or worksheet range
dest	any cell reference or worksheet range

This function transposes the information from *source* into *dest*, such that rows and columns are transposed. If a cell reference is given for *dest*, it will be used as the upper left hand corner of the destination range. Any information in the destination range will be overwritten.

Returns: *dest*

```
transpose(a1:b7,d12)

transpose(myrange,d12:e14)
```

UPPER

Converts strings in the specified cell or range to UPPER CASE.

UPPER(*cell*) or UPPER(*range*)

cell	any cell reference
range	any worksheet range

This function converts any strings in the *cell* or *range* to UPPER CASE, regardless of their original case.

Returns: *cell or range*

```
upper(a1)

upper(a1:b7)

upper(myrange)
```

Formatting Operations

BOLD

Toggles the data in the specified range between **bold** and regular weight.

BOLD(*cell*) or BOLD(*range*)

cell	any cell reference
range	any worksheet range

This function changes the format of the selected cells to **bold** if they are regular weight, and to regular weight if they are **bold**.

Returns: *cell or range*

```
bold(a1)

bold(a1:b7)

bold(myrange)
```

CLRHBREAK or CLEARHBREAK

Clears horizontal page breaks from the specified cell or range.

CLRHBREAK(*cell*) or CLRHBREAK(*range*)

cell	any cell reference
range	any worksheet range

This function clears any horizontal page breaks within the *cell* or *range*, including breaks following the last cell in the range, but not breaks preceding the first cell.

Returns: *cell or range*

```
clrhbreak(a1)

clrhbreak(a1:b7)

clrhbreak(myrange)
```

CLRVBREAK or CLEARVBREAK

Clears horizontal page breaks from the specified cell or range.

CLRVBREAK(*cell*) or CLRVBREAK(*range*)

cell	any cell reference
range	any worksheet range

This function clears any vertical page breaks within the *cell* or *range*, including breaks following the last cell in the range, but not breaks preceding the first cell.

Returns: *cell or range*

```
clrvbreak(a1)
```

```
clrvbreak(a1:b7)
```

```
clrvbreak(myrange)
```

DEFAULTS

Sets the formatting of the specified cell or range to its defaults.

DEFAULTS(*cell*) or DEFAULTS(*range*)

cell	any cell reference
range	any worksheet range

This function changes the format of the selected *cell* or *range* to the defaults for its current style template. For more information about style templates, see “Style Template submenu” .

Returns: *cell or range*

```
defaults(a1)
```

```
defaults(a1:b7)
```

```
defaults(myrange)
```

HBREAK

Sets a horizontal page break.

HBREAK(*cell*) or HBREAK(*range*)

cell	any cell reference

range	any worksheet range

Sets a horizontal page break immediately after *cell* or *range*.

Returns: *cell or range*

```
hbreak(a1)
```

```
hbreak(a1:b7)
```

```
hbreak(myrange)
```

HIDDENCOL

Tests for hidden columns.

HIDDENCOL(*cell*)

cell any cell reference

This functions tests if *cell* is in a hidden column.

Returns: *1 if cell is hidden, 0 if not*

```
hiddencol(h19)
```

HIDDENROW

Tests for hidden rows.

HIDDENROW(*cell*)

cell any cell reference

This function tests if *cell* is in a hidden row.

Returns: *1 if cell is hidden, 0 if not*

```
hiddenrow(h19)
```

HIDECOL

Hides the specified columns.

HIDECOL(*cell*) or HIDECOL(*range*)

cell	any cell reference
range	any worksheet range

This function hides the columns that contain *cell* or *range*. Column A cannot be hidden.

Returns: *cell or range*

```
hidecol(b5)

hidecol(b5:c7)

hidecol(myrange)
```

HIDEROW

Hides the specified rows.

HIDEROW(*cell*) or HIDEROW(*range*)

cell	any cell reference
range	any worksheet range

This function hides the rows that contain *cell* or *range*. Row 1 cannot be hidden.

Returns: *cell or range*

```
hiderow(b5)

hiderow(b5:c7)

hiderow(myrange)
```

ITALIC

Toggles the data in the specified range between *italic* and standard text.

cell	any cell reference
range	any worksheet range

This function changes the format of the cells specified by *cell* or *range* to *italic* if they are standard, and to standard if they are *italic*.

Returns: *cell or range*

```
italic(a1)

italic(a1:b7)

italic(myrange)
```

OUTLINE

Outlines the specified cells with the specified border.

OUTLINE(*cell*, *border*) or OUTLINE(*range*, *border*)

cell	any cell reference
range	any worksheet range
border	the specified border type: 0, 1 or 2

This function outlines each of the specified cells with a border of type *border*. Possible borders are 0 (no border), 1 (single border) or 2 (double border).

Returns: *cell or range*

```
outline(a1,1)

outline(a1:b7,0)

outline(myrange,2)
```

PLAIN

Changes the format of the specified cell or range to plain text.

PLAIN(*cell*) or PLAIN(*range*)

cell	any cell reference
range	any worksheet range

This function changes the text formatting of *cell* or *range* to plain (i.e. non-**bold**, non-*italic)*.

Returns: *cell or range*

```
plain(a1)

plain(a1:b7)

plain(myrange)
```

RESETROWSIZE

Resets the size of the specified rows.

RESETROWSIZE(*cell*) or RESETROWSIZE(*range*)

cell	any cell reference
range	any worksheet range

This function sets the row(s) that contain *cell* or *range* to the default for the worksheet.

Returns: *cell or range*

```
resetrowsize(a1)

resetrowsize(a1:b7)

resetrowsize(myrange)
```

RESETCOLSIZE

Resets the size of the specified columns.

RESETCOLSIZE(*cell*) or RESETCOLSIZE(*range*)

cell	any cell reference
range	any worksheet range

This function sets the column(s) that contain *cell* or *range* to the default for the worksheet.

Returns: *cell or range*

```
resetcolsize(a1)

resetcolsize(a1:b7)

resetcolsize(myrange)
```

SETALIGN

Sets the text alignment of the specified cell or range.

SETALIGN(*cell*, *h-align* [, *v-align*])
SETALIGN(*range*, *h-align* [, *v-align*])

cell	any cell reference
range	any worksheet range
h-align	a horizontal alignment constant
v-align	an [optional] vertical alignment constant

This function aligns the text in *cell* or *range* according to *h-align* and an optional *v-align*. For a list of alignment constants, see “Alignment Constants” .

Returns: *cell or range*

```
setalign(a1,LEFTALIGN)

setalign(a1:b7,RIGHTALIGN,TOPALIGN)

setalign(myrange,RIGHTALIGN,BOTTOMALIGN)
```

SETALTCOLOR

Sets the alternate text color of the specified cell or range.

SETALTCOLOR(*cell*, *color* [, *pattern*]),
SETALTCOLOR(*range*, *color* [, *pattern*]),
SETALTCOLOR(*cell*, *r*, *g*, *b* [, *pattern*]), or
SETALTCOLOR(*range*, *r*, *g*, *b* [, *pattern*])

cell	any cell reference
range	any worksheet range
color	a color constant
r	the amount of red
g	the amount of green
b	the amount of blue
pattern	a pattern number

This function sets the alternate text color of *cell* or *range* to the specified *color* and optional *pattern*. The color can be specified by a color constant *color* or with red, green, and blue values (*r*, *g*, *b*) between 0 and 1. For a list of color constants, see “Color constants” . The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setaltcolor(a1,BLACKCOLOR)

setaltcolor(a1:b7,0,0.5,0.5)

setaltcolor(myrange,MAGENTACOLOR,53)
```

SETALTPATTERN

Sets the alternate text pattern of the specified cell or range.

SETALTPATTERN(*cell*, *pattern*) or
SETALTPATTERN(*range*, *pattern*)

cell	any cell reference
range	any worksheet range
pattern	a pattern number

This function sets the alternate text pattern of *cell* or *range* to *pattern*, using the current alternate color. The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setaltpattern(a1,23)

setaltpattern(a1:b7,17)

setaltpattern(myrange,59)
```

SETBKGCOLOR

Sets the background color of the specified cell or range.

SETBKGCOLOR(*cell*, *color* [, *pattern*]),
SETBKGCOLOR(*range*, *color* [, *pattern*]),
SETBKGCOLOR(*cell*, *r*, *g*, *b* [, *pattern*]), or
SETBKGCOLOR(*range*, *r*, *g*, *b* [, *pattern*])

cell	any cell reference
range	any worksheet range
color	a color constant
r	the amount of red
g	the amount of green
b	the amount of blue
pattern	a pattern number

This function sets the background color of *cell* or *range* to the specified *color* and optional *pattern*. The color can be specified by a color constant *color* or with red, green, and blue values (*r*, *g*, *b*) between 0 and 1. For a list of color constants, see “Color constants” . The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setbkgcolor(a1,BLACKCOLOR)

setbkgcolor(a1:b7,0,0.5,0.5)

setbkgcolor(myrange,MAGENTACOLOR,53)
```

SETBKGPATTERN

Sets the background pattern of the specified cell or range.

SETBKGPATTERN(*cell*, *pattern*) or
SETBKGPATTERN(*range*, *pattern*)

cell	any cell reference

range	any worksheet range
pattern	a pattern number

This function sets the background pattern of *cell* or *range* to *pattern*, using the current background color. The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setbkgpattern(a1,23)

setbkgpattern(a1:b7,17)

setbkgpattern(myrange,59)
```

SETBORDER

Sets the border for a specified cell or range.

SETBORDER(*range*, *borders*, *bool*)

range	any worksheet range
borders	a border constant
bool	whether the border should be visible

This function places a border in *range* around the sides of the cells specified by *borders*, which can be one of: `LEFTBORDER`, `RIGHTBORDER`, `TOPBORDER`, `BOTTOMBORDER`. These constants can be OR'd together. The *bool* argument should be TRUE or FALSE, depending on whether the border should be visible.

Returns: *range*

```
setborder(a1,TOPBORDER,TRUE)

setborder(a1:b7,LEFTBORDER |
RIGHTBORDER,TRUE)

setborder(myrange,TOPBORDER |
BOTTOMBORDER,FALSE)
```

SETBORDERCOLOR

Sets the border color for a specified cell or range.

SETBORDERCOLOR(*cell*, *color* [, *pattern*]),
SETBORDERCOLOR(*range*, *color* [, *pattern*]),
SETBORDERCOLOR(*cell*, *r*, *g*, *b* [, *pattern*]), or
SETBORDERCOLOR(*range*, *r*, *g*, *b* [, *pattern*])

cell	any cell reference
range	any worksheet range
color	a color constant
r	the amount of red
g	the amount of green
b	the amount of blue
pattern	a pattern number

This function sets the border color of *cell* or *range* to the specified *color* and optional *pattern*. The color can be specified by a color constant *color* or with red, green, and blue values (*r*, *g*, *b*) between 0 and 1. For a list of color constants, see "Color constants" . The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setbordercolor(a1,BLACKCOLOR)
```

```
setbordercolor(a1:b7,0,0.5,0.5)
```

```
setbordercolor(myrange,MAGENTACOLOR,53)
```

SETBORDERPATTERN

Sets the border pattern for the specified cell or range.

SETBORDERPATTERN(*cell*, *pattern*) or
SETBORDERPATTERN(*range*, *pattern*)

cell	any cell reference
range	any worksheet range
pattern	a pattern number

This function sets the border pattern of *cell* or *range* to *pattern*, using the current border color. The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setborderpattern(a1,23)
```

```
setborderpattern(a1:b7,17)
```

```
setborderpattern(myrange,59)
```

SETBORDERTYPE

Sets the border type for a specified cell or range.

SETBORDERTYPE(*cell*, *border*) or

SETBORDERTYPE(*range*, *border*)

cell	any cell reference
range	any worksheet range
border	a border type: 0, 1, or 2

This function changes the border for *cell* or *range* to border type *border*. Possible borders are 0 (no border), 1 (single border) or 2 (double border). This does not place a border around the cell.

Returns: *cell or range*

```
setbordertype(a1,1)

setbordertype(a1:b7,0)

setbordertype(myrange,2)
```

SETCOLOR

Sets the text color for a specified cell or range.

SETCOLOR(*cell*, *color* [, *pattern*]),
SETCOLOR(*range*, *color* [, *pattern*]),
SETCOLOR(*cell*, *r*, *g*, *b* [, *pattern*]), or
SETCOLOR(*cell*, *r*, *g*, *b* [, *pattern*])

cell	any cell reference
range	any worksheet range
color	a color constant
r	the amount of red
g	the amount of green
b	the amount of blue
pattern	a pattern number

This function sets the text color of *cell* or *range* to the specified *color* and optional *pattern*. The color can be specified by a color constant *color* or with red, green, and blue values (*r*, *g*, *b*) between 0 and 1. For a list of color constants, see “Color constants” . The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setcolor(a1,BLACKCOLOR)

setcolor(a1:b7,0,0.5,0.5)

setcolor(myrange,MAGENTACOLOR,53)
```

SETCOLSIZE

Sets the size of the specified columns.

SETCOLSIZE(*cell*, *num*) or SETCOLSIZE(*range*, *num*)

cell	any cell reference
range	any worksheet range
num	the size of the column(s) in points

This function sets the column(s) that contain *cell* or *range* to a size of *num* points.

Returns: *cell or range*

```
setcolsize(a1,72)

setcolsize(a1:b7,100)

setcolsize(myrange,53)
```

SETFMT or SETDISPLAYFMT

Sets the display format of the specified cell or range.

SETFMT(*cell*, *format* [, *precision*]) or
SETFMT(*range*, *format* [, *precision*])

cell	any cell reference
range	any worksheet range
format	the size of the column(s) in points
precision	the [optional] number of decimal places.

This function sets the display format of *cell* or *range* to the format specified by *format* and, optionally, to *precision* decimal places. For a list of formatting constants, see “Formatting Constants” .

Returns: *cell or range*

```
setfmt(a1:b10,CURRENCYFMT,2)
```

SETFONT

Sets the font of the specified cell or range.

SETFONT(*cell*, *fontname*, *size*) or
SETFONT(*range*, *fontname*, *size*)

cell	any cell reference
range	any worksheet range
fontname	the name of a font

size	the point size

This function changes the font in *cell* or *range* to the font *fontname* and point size *size*.

Returns: *cell or range, or -1 on error*

```
setfont(a1:b7,"ZapfChancery-MediumItalic",12)
```

SETPATTERN

Sets the text pattern for the specified cell or range.

SETPATTERN(*cell*, *pattern*) or
SETPATTERN(*range*, *pattern*)

cell	any cell reference
range	any worksheet range
pattern	a pattern number

This function sets the text pattern of *cell* or *range* to *pattern*, using the current text color. The pattern can be specified with a number between 0 and 87.

Returns: *cell or range*

```
setpattern(a1,23)

setpattern(a1:b7,17)

setpattern(myrange,59)
```

SETPRECISION

Sets the precision of the specified cell or range.

SETPRECISION(*cell*, *precision*) or
SETPRECISION(*range*, *precision*)

cell	any cell reference
range	any worksheet range
precision	the number of decimal places.

Sets the number of decimal places for *cell* or *range* to *precision*.

Returns: *cell or range*

```
setprecision(a1,2)

setprecision(a1:b7,3)
```

```
setprecision(myrange,0)
```

SETROWSIZE

Sets the size of the specified rows.

SETROWSIZE(*cell*, *num*) or SETROWSIZE(*range*, *num*)

cell	any cell reference
range	any worksheet range
num	the size of the row(s) in points

This function sets the row(s) that contain *cell* or *range* to a size of *num* points.

Returns: *cell or range*

```
setrowsize(a1,72)

setrowsize(a1:b7,100)

setrowsize(myrange,53)
```

SETTEMPLATE

Sets the style template of the specified cell or range.

SETTEMPLATE(*cell*, *template*) or SETTEMPLATE(*range*, *template*)

cell	any cell reference
range	any worksheet range
template	a style template number

Sets the style template for *cell* or *range* to *template*. *template* can be an integer between 1 and 8, inclusive. For more information about style templates, see "Style Template submenu" .

Returns: *cell or range*

```
setrowsize(a1,1)

setrowsize(a1:b7,6)

setrowsize(myrange,3)
```

SETVERTALIGN

Sets the vertical text alignment of the specified cell or range.

SETVERTALIGN(*cell*, *alignment*) or
SETVERTALIGN(*range*, *alignment*)

cell	any cell reference
range	any worksheet range
alignment	a vertical alignment constant

This function vertically aligns the text in *cell* or *range* according to *alignment*. For a list of alignment constants, see “Alignment Constants” .

Returns: *cell or range*

```
setalign(a1,TOPALIGN)

setalign(a1:b7,TOPALIGN)

setalign(myrange,BOTTOMALIGN)
```

SMARTCOLSIZE

Smartsizes the columns of the specified cell or range.
SMARTCOLSIZE(*cell*) or SMARTCOLSIZE(*range*)

cell	any cell reference
range	any worksheet range

This function sets the column(s) that contain *cell* or *range* to the best column size for the data they contain.

Returns: *cell or range*

```
smartcolsize(a1)

smartcolsize(a1:b7)

smartcolsize(myrange)
```

SMARTROWSIZE

Smartsizes the rows of the specified cell or range.

SMARTROWSIZE(*cell*) or SMARTROWSIZE(*range*)

cell	any cell reference
range	any worksheet range

This function sets the row(s) that contain *cell* or *range* to the best row size for the data they contain.

Returns: *cell or range*

```
smartrowsize(a1)
```

```
smartrowsize(a1:b7)

smartrowsize(myrange)
```

UNHIDECOL

Unhides the specified columns.

UNHIDECOL(*cell*) or UNHIDECOL(*range*)

cell	any cell reference
range	any worksheet range

This function unhides any hidden columns in *cell* or *range*.

Returns: *cell or range*

```
unhidecol(a1)

unhidecol(a1:b7)

unhidecol(myrange)
```

UNHIDEROW

Unhides the specified rows.

UNHIDEROW(*cell*) or UNHIDEROW(*range*)

This function unhides any hidden rows in *cell* or *range*.

Returns: *cell or range*

```
unhiderow(a1)

unhiderow(a1:b7)

unhiderow(myrange)
```

VBREAK

Sets a vertical page break.

VBREAK(*cell*) or VBREAK(*range*)

cell	any cell reference
range	any worksheet range

Sets a vertical page break immediately after *cell* or *range*.

Returns: *cell or range*

```
vbreak(a1)

vbreak(a1:b7)

vbreak(myrange)
```

Graphics Operations

ABOVESHADING

Places a graphic above the shading on the sheet.

ABOVESHADING(*name*)

name the name of a graphic

This function places the graphic named by *name* above the shading but below the text on the sheet. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
aboveshading("Graph 1")

c1 := "MyGraphic"
aboveshading(c1)
```

ABOVESHEET

Places a graphic about the sheet.

ABOVESHEET(*name*)

name the name of a graphic

This function places the graphic named by *name* above the sheet. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
abovesheet("Graph 1")

c1 := "MyGraphic"
abovesheet(c1)
```

BRINGTOFRONT

Brings a graphic to the front of its sheet layer.

BRINGTOFRONT(*name*)

name the name of a graphic

This function brings the graphic named by *name* to the front of other graphics in the same layer on the sheet. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
bringtofront("Graph 1")

c1 := "MyGraphic"
bringtofront(c1)
```

CLEARGRAPH

Clears a graph.

CLEARGRAPH(*name*)

name	the name of a graph

This function clears the graph named by *name* from the sheet. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
cleargraph("Graph 1")

c1 := "MyGraphic"
cleargraph(c1)
```

GRAPH

Creates a graph.

GRAPH(*range*, *type* [, *name*])

range	any worksheet range
type	a graph type constant
name	the [optional] graph name

This function graphs *range* in a graph of type *type*. For a list of graph type constants, see "Graph Constants" . Optionally, the graph can be given a name of *name*. *name* must be a string. If a *name* is not chosen, the name assigned is "Graph *n*", where *n* starts at 1 and is incremented with each successive graph.

Returns: *name, or -1 if there's an error*

```
graph(a1:c10,BARGRAPH)

graph(d10:f17,COLUMNGRAPH,"MyGraph")
```

SENDTOBACK

Sends a graphic to the back of its sheet layer.

SENDTOBACK(*name*)

name the name of a graphic

This function sends the graphic named by *name* to the back of other graphics in the same layer on the sheet. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
sendtoback("Graph 1")

c1 := "MyGraphic"
sendtoback(c1)
```

SETGRAPHELEMENTCOLOR

Sets the color of an element (row, column, line, etc) in a graph.

SETGRAPHELEMENTCOLOR(*graph*,*element*,*color)*

graph the name of the graph.
element the element to be affected (starting at 0).
color a color constant.

SETGRAPHITEMCOLOR

Sets the color of a graphic item.

SETGRAPHITEMCOLOR(*graph*,*item*,*color*)

graph the name of the graph.
item the item to be affected (see below).
color a color constant.

Item:

bkgColorGraphItem - the background color of the graph
borderColorGraphItem - the border color of the graph
legendColorGraphItem - the color of the legend
threeDColorGraphItem - the color of the 3D background
baseColorGraphItem - the color of the text and the axises

SETGRAPHITEMVALUE

Sets the numeric value of an item in a graph.

SETGRAPHITEMVALUE(*graph*,*item*,*value*)

graph	the name of the graph.
item	the item to be affected (see below).
value	the new value of the item.

Item:

hasClearBkgGraphItem - 1 if the graph has a clear background, 0 if it is colored

borderWidthGraphItem - width of the border around the graph. 0 if no border

hasClearLegendGraphItem - 1 if the legend is clear. 0 if colored

hasClear3DGraphItem - 1 if the 3D backgeound is clear. 0 if colored.

labelPositionGraphItem - the position of the Y labels. 0 if centered. 1 if grouped.

topRowLabelsGraphItem - 1 if the top row has labels. 0 if the top row has graphed data.

leftColumnLabelsGraphItem - 1 if the left column has labels. 0 if it has graphed data.

displayFormatGraphItem - the display format of the labels on the graph axis.

precisionGraphItem - sets the precision of the graph labels

xTitleIsCellRefGraphItem - 1 if the X Title is a cell reference, 0 if it is text

yTitleIsCellRefGraphItem - 1 if the Y Title is a cell reference, 0 if it is text

firstTitleIsCellRefGraphItem - 1 if the first Title is a cell reference, 0 if it is text

secondTitleIsCellRefGraphItem - 1 if the second Title is a cell reference, 0 if it is text

orientationGraphItem - 0 Horizantal data orientation. 1 Vertical data orientation

yMajTickGraphItem
yMajTickSizeGraphItem
xMajTickGraphItem
xMajTickSizeGraphItem
yMinTickGraphItem
yMinTickSizeGraphItem
xMinTickGraphItem
xMinTickSizeGraphItem
autoScaleGraphItem

logScaleGraphItem
xMajHairGraphItem
xMinHairGraphItem
yMajHairGraphItem
yMinHairGraphItem
hasXBorderGraphItem
hasYBorderGraphItem
barColumnSizeGraphItem
hasBarColumnLineGraphItem
barColumnLineWidthGraphItem
scatterGraphTypeGraphItem
singleXRangeGraphItem
candleStickGraphItem
wireFrameGraphItem
legendLocationGraphItem
xyDisplayFormatGraphItem
xyPrecisionGraphItem
xLabelRotGraphItem
yLabelRotGraphItem
xMaxGraphItem
xMinGraphItem
yMaxGraphItem
yMinGraphItem
xRotGraphItem
yRotGraphItem
strikeThroughGraphTickMark
innerGraphTickMark
outerGraphTickMark
scatterGraphPointsOnly
scatterGraphPointsAndLine
scatterGraphLineOnly
noMesaLegend
leftMesaLegend
rightMesaLegend
topMesaLegend
bottomMesaLegend

SETGRAPHITEMSTRING

Sets the string value of an item in a graph.

SETGRAPHITEMSTRING(*graph*,*item*,*string*)

graph	the name of the graph.
item	the item to be affected (see below).
string	the new value of the item.

Item:

xTitleGraphItem - the X Title

yTitleGraphItem - the Y Title
firstTitleGraphItem - the First Title
secondTitleGraphItem - the Second Title
topTitleGraphItem - the top title
sideTitleGraphItem - the side title

SETGRAPHITEMRANGE

Sets an item in a graph to refer to a range.

SETGRAPHITEMRANGE(*graph*,*item, range*)

graph	the name of the graph.
item	the item to be affected (see below).

Item:

xLabelGraphItem - the X Label range (accessed only if Top Row has Labels is not checked)

yLabelGraphItem - the Y Label range (accessed only if Left Column has Labels is not checked)

baseRangeGraphItem - The base range of the graph information.

baseRangeGraphItem1 - An additional range of graph information.

baseRangeGraphItem2 - An additional range of graph information.

baseRangeGraphItem3 - An additional range of graph information.

baseRangeGraphItem4 - An additional range of graph information.

baseRangeGraphItem5 - An additional range of graph information.

baseRangeGraphItem6 - An additional range of graph information.

SETGRAPHSIZE

Changes the size of the specified graph.

SETGRAPHSIZE(*name*, *width*, *height*)

name	the name of a graphic
width	the width of the graph in points
height	the height of the graph in points

This function sets the *width* and *height* of the graph *name*, in points.

Returns: *name, or -1 if name isn't found*

```
setgraphsize("Graph 1",520,350)
```

SETGRAPHTYPE

Sets *graph* to *type*.

GETGRAPHITEMRANGE(*graph*,*item*)

graph — the name of the graph.

type — a graph type constant (see GRAPH()).

SETGRAPHXY

Sets the X and Y coordinates of a graph.

SETGRAPHXY(*name*, *x*, *y*)

name	the name of a graphic
x	the width of the graph in points
y	the height of the graph in points

This function sets the X and Y coordinates, in points, of the graph *name* to *x* and *y*, respectively. The X coordinate is measured from the left of the worksheet, and the Y coordinate is measured from the top of the worksheet. The name *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
setgraphxy("Graph 1",100,200)
```

UNDERALL

Places a graphic under everything on a sheet.

UNDERALL(*name*)

name	the name of a graphic

This function places the graphic named by *name* under everything in the sheet, including text and shading. *name* must be a string.

Returns: *name, or -1 if name isn't found*

```
underall("Graph 1")

c1 := "MyGraphic"
underall(c1)
```

Other Operations

ADDIN

Calls an EventAddIn with optional data.

ADDIN(*name* [, *data*...])

name	the name of an EventAddIn
data	the [optional] data to be passed to the AddIn

This function calls the EventAddIn *name*, optionally passing it the specified *data*. The name *name* must be expressed as a string.

Returns: *the return value from the AddIn*

```
addin("MyAddIn")
```

ALERT

Opens an alert panel.

ALERT(*title*, *message* [, *button1* [, *button2* [, *button3*]]])

title	the title of the alert panel
message	the message of the alert panel
*button*N	the titles of the buttons

This function opens an alert panel with the *title*, *message*, and *button*s specified. The *title, message*, and *button*N names must be strings. If no buttons are specified, one button will appear, labeled *OK*.

Returns: *the number of the button pressed, with the first button numbered 0*

```
alert("Warning","The number entered
exceeds the maximum.", "Redo","Accept")
```

DISPLAY

Re-displays the worksheet.

DISPLAY()

This function re-displays the worksheet. By default, the sheet is only re-displayed at the end of a script. This is useful for re-displaying the worksheet during the execution of a long script.

Returns: *0*

```
display()
```

FREEZE

Freezes the value of a cell reference.

FREEZE(*value*)

value any cell reference

This function returns the current value of *cell*. This is useful for assignment of the value in a cell to a local variable.

Returns: *the current value of value*

```
b1 := freeze(a1)
```

```
?x := freeze(g7)
```

INVERT

Inverts a matrix.

INVERT(*range*, *dest*)

range any worksheet range representing a matrix
dest the destination of the inverted matrix

This function inverts the matrix in *range* and places the inverse in *dest*. Both ranges must be square and of the same size.

Returns: *0 if successful and -1 if unsuccessful*

```
invert(a1:c3,d1:f3)
```

```
invert(mymatrix,g7:j10)
```

LET

Assigns a value to a variable.

LET(*variable*, *value*)

variable	a variable name
value	a number, string, label, or cell reference

This function sets a *variable* to a *value*. The *variable* must be expressed as a string. Variables can also be assigned with the := operator.

Returns: *value*

```
LET("?x",10).
% is the same as
?x := 10
```

MENU

Displays a menu of choices.

MENU(*range* [, MENUBYROW|MENUBYCOL])

This function gives the user a menu of choices.
If MENUBYCOL is used (this is default), for each column in *range*, the first row is the short name, the second row is the long description, and the third row is the string to return (this can be used for a subsequent "goto" or "gosub").
If MENUBYROW is used, the orientation of the table is horizontal rather than vertical.

Returns: *the return string, or -1 if the Cancel button is clicked*

```
?foo := menu(a1:c3,MENUBYROW)
```

MULTIPLY

Multiplies two matrices.

MULTIPLY(*Xrange*, *YRange*, *dest*)

XRange	a worksheet range representing a matrix
YRange	a worksheet range representing a matrix
dest	the destination range

This function multiplies the two matrices specified by *XRange* and *YRange* and places the result in *dest*.

Returns: *0 if successful and -1 if unsuccessful*

```
multiply(A1:C3,E1:F3,H1:I3)
```

PRINT

Prints a report or a part of the worksheet.

PRINT([*range*]) or PRINT([*report*])

This function prints the *range* or the *report* specified. Report names must be expressed as strings. If nothing is specified, PRINT() will print all the information on the sheet.

Returns: *1 if success, -1 if failure*

```
print()
```

```
print("MyReport")
```

```
print(a1:g27)
```

PROMPT

Prompts the user for input.

PROMPT(*text*)

text the user prompt string

This function prompts the user for input with the string *text*. The message *text* must be expressed as a string.

Returns: *the user's input, as a string*

```
a1 := prompt("Type in any word.")
```

```
a2 := value(prompt("Type in any number"))
```

QUERY

Executes an SQL query.

QUERY(*name*)

name the query name

This function executes the pre-defined SQL query *name*. *name* must be expressed as a string. The query must first be defined in the Query Inspector. For more information, see "Query Inspector" . This MScript command is not available in Sunrise.

Returns: *0*

```
query("myquery")
```

```
query(c7)
```

RECALC

Recalculates the worksheet.

RECALC(), RECALC(*cell*) or RECALC(*range*)

This function recalculates the *range* or, if no range is specified, the whole worksheet.

Returns: *cell or range*

```
recalc()
```

```
recalc(b7:z99)
```

REGRESSION

Performs a linear regression on the specified ranges.

REGRESSION(*Xrange*, *YRange*, *dest*)

XRange	a worksheet range representing a matrix
YRange	a worksheet range representing a matrix
dest	the destination range

Performs linear regression on the input ranges specified by *XRange* and *YRange* and places the result in the range *dest*.

Returns: *0 if successful and -1 if unsuccessful*

```
regression(A1:C3,D1:D3,F1:F3)
```

REPORT

Prints a specified report.

REPORT(*name*)

name	the name of a report

This function prints the pre-defined report *name*. *name* must be expressed as a string. The report must first be defined in the Report Layout Inspector. For more information about reports, see “Report Layout

Inspector" .

Returns: *0*

```
report("myreport")
```

```
report(c18)
```

SCENARIO

Runs a scenario.

SCENARIO(*name*)

name the scenario name

This function runs the pre-defined scenario *name*. *name* must be expressed as a string. The scenario must first be defined in the Scenario Inspector. For more information about scenarios, see "Scenario Inspector" .

Returns: *0*

```
scenario("myscenario")
```

```
scenaro(c9)
```

SETADDRESS

Sets the cell address for the macro execution.

SETADDRESS(*cell*)

This function sets the address of the *cell* that the macro is executing in. This is useful for the SAME() command. The default is A1.

```
setaddress(g17)
```

SETINPUT

Sets the input of a specified cell.

SETINPUT(*cell*, *string*)

cell any cell reference
string a string

This function sets the input of the cell *cell* to *string*. The *string* will be interpreted by the cell as if typed by the user.

the evaluated *string*

```
setinput(c1,"=a1+b1")
```

SETLABEL

Assigns a label name to a range of cells.

SETLABEL(*label*, *range*)

label	any string
range	any worksheet range

This function assigns *range* to the name *label*. *label* must be expressed as a string.

Returns: *0*

```
setlabel("myrange",a1:g13)
```

SETPROTECTION

Sets the protection for a given range.

SETPROTECTION(*range*,TRUE|FALSE)

SOLVE

Finds an input value such that a specified formula will give a desired result.

SOLVE(*var*, *comp*, *value* [, *prec*])

var	any cell reference
comp	any cell reference containing a formula
value	the desired result
prec	the [optional] digits of precision

This function tries to find the correct value of the cell *var* such that the formula in *comp* yields *value* (a number). It places the result in *var*. Optionally, the number of digits of precision can be specified with *prec*. The default is 6.

Returns: *0 if successful, -1 if not*

```
solve(A1,B1,2.5)
```

```
solve(A1,B1,2.5,8)
```

SORT

Sorts a range.

SORT(*range*, *titles*, SORTBYROW | SORTBYCOL, *key1*, *sorttype* [, *key2*, *sorttype* [, *key3*, *sorttype*]])

range	any worksheet range
titles	a boolean stating if the range has titles
*key*N	the offset of the column to sort by
sorttype	a sort-type constant (see below)

This functions sorts the given *range*. If *titles* is true, the top row of the range is ignored. To sort by rows, use the SORTBYROW constant; to sort by columns, use the SORTBYCOL constant. The *key* is the number of rows or columns that the key row or column is offset from the top or left side of the range. Use ASCENDINGSORT or DESCENDINGSORT to specify the order of the sort. *key2* and *key3* are optional.

Returns: *0*

```
sort(a1:b10,false,SORTBYROW,0,ASCENDINGSORT)
```

SYSTEM

Sends a string to the shell.

SYSTEM(*string*)

string any string that can be interpreted by the shell
This function sends the string *string* to the shell for execution.

Returns: *the return value of the shell command*

Other Operations

ADDIN

Calls an EventAddIn with optional data.

ADDIN(*name* [, *data*...])

name the name of an EventAddIn
data the [optional] data to be passed to the AddIn

This function calls the EventAddIn *name*, optionally passing it the specified *data*. The name *name* must be expressed as a string.

Returns: *the return value from the AddIn*

```
addin("MyAddIn")
```

ALERT

Opens an alert panel.

ALERT(*title*, *message* [, *button1* [, *button2* [, *button3*]]])

title the title of the alert panel
message the message of the alert panel
*button*N the titles of the buttons

This function opens an alert panel with the *title*, *message*, and *button*s specified. The *title, message*, and *button*N names must be strings. If no buttons are specified, one button will appear, labeled *OK*.

Returns: *the number of the button pressed, with the first button numbered 0*

```
alert("Warning","The number entered
exceeds the maximum.", "Redo","Accept")
```

DISPLAY

Re-displays the worksheet.

DISPLAY()

This function re-displays the worksheet. By default, the sheet is only re-displayed at the end of a script. This is useful for re-displaying the worksheet during the execution of a long script.

Returns: *0*

```
display()
```

FREEZE

Freezes the value of a cell reference.

FREEZE(*value*)

value any cell reference

This function returns the current value of *cell*. This is useful for assignment of the value in a cell to a local variable.

Returns: *the current value of value*

```
b1 := freeze(a1)
```

```
?x := freeze(g7)
```

INVERT

Inverts a matrix.

INVERT(*range*, *dest*)

range any worksheet range representing a matrix
dest the destination of the inverted matrix

This function inverts the matrix in *range* and places the inverse in *dest*. Both ranges must be square and of the same size.

Returns: *0 if successful and -1 if unsuccessful*

```
invert(a1:c3,d1:f3)
```

```
invert(mymatrix,g7:j10)
```

LET

Assigns a value to a variable.

LET(*variable*, *value*)

variable	a variable name
value	a number, string, label, or cell reference

This function sets a *variable* to a *value*. The *variable* must be expressed as a string. Variables can also be assigned with the := operator.

Returns: *value*

```
LET("?x",10).
% is the same as
?x := 10
```

MENU

Displays a menu of choices.

MENU(*range* [, MENUBYROW|MENUBYCOL])

This function gives the user a menu of choices.
If MENUBYCOL is used (this is default), for each column in *range*, the first row is the short name, the second row is the long description, and the third row is the string to return (this can be used for a subsequent "goto" or "gosub").
If MENUBYROW is used, the orientation of the table is horizontal rather than vertical.

Returns: *the return string, or -1 if the Cancel button is clicked*

```
?foo := menu(a1:c3,MENUBYROW)
```

MULTIPLY

Multiplies two matrices.

MULTIPLY(*Xrange*, *YRange*, *dest*)

XRange	a worksheet range representing a matrix
YRange	a worksheet range representing a matrix
dest	the destination range

This function multiplies the two matrices specified by *XRange* and *YRange* and places the result in *dest*.

Returns: *0 if successful and -1 if unsuccessful*

```
multiply(A1:C3,E1:F3,H1:I3)
```

PRINT

Prints a report or a part of the worksheet.

PRINT([*range*]) or PRINT([*report*])

This function prints the *range* or the *report* specified. Report names must be expressed as strings. If nothing is specified, PRINT() will print all the information on the sheet.

Returns: *1 if success, -1 if failure*

```
print()

print("MyReport")

print(a1:g27)
```

PROMPT

Prompts the user for input.

PROMPT(*text*)

text the user prompt string

This function prompts the user for input with the string *text*. The message *text* must be expressed as a string.

Returns: *the user's input, as a string*

```
a1 := prompt("Type in any word.")

a2 := value(prompt("Type in any number"))
```

QUERY

Executes an SQL query.

QUERY(*name*)

name the query name

This function executes the pre-defined SQL query *name*. *name* must be expressed as a string. The query must first be defined in the Query Inspector. For more information, see "Query Inspector" . This MScript command is not available in Sunrise.

Returns: *0*

```
query("myquery")
```

```
query(c7)
```

RECALC

Recalculates the worksheet.

RECALC(), RECALC(*cell*) or RECALC(*range*)

This function recalculates the *range* or, if no range is specified, the whole worksheet.

Returns: *cell or range*

```
recalc()
```

```
recalc(b7:z99)
```

REGRESSION

Performs a linear regression on the specified ranges.

REGRESSION(*Xrange*, *YRange*, *dest*)

XRange	a worksheet range representing a matrix
YRange	a worksheet range representing a matrix
dest	the destination range

Performs linear regression on the input ranges specified by *XRange* and *YRange* and places the result in the range *dest*.

Returns: *0 if successful and -1 if unsuccessful*

```
regression(A1:C3,D1:D3,F1:F3)
```

REPORT

Prints a specified report.

REPORT(*name*)

name	the name of a report

This function prints the pre-defined report *name*. *name* must be expressed as a string. The report must first be defined in the Report Layout Inspector. For more information about reports, see “Report Layout

Inspector” .

Returns: *0*

```
report("myreport")
```

```
report(c18)
```

SCENARIO

Runs a scenario.

SCENARIO(*name*)

name the scenario name

This function runs the pre-defined scenario *name*. *name* must be expressed as a string. The scenario must first be defined in the Scenario Inspector. For more information about scenarios, see “Scenario Inspector” .

Returns: *0*

```
scenario("myscenario")
```

```
scenaro(c9)
```

SETADDRESS

Sets the cell address for the macro execution.

SETADDRESS(*cell*)

This function sets the address of the *cell* that the macro is executing in. This is useful for the SAME() command. The default is A1.

```
setaddress(g17)
```

SETINPUT

Sets the input of a specified cell.

SETINPUT(*cell*, *string*)

cell any cell reference
string a string

This function sets the input of the cell *cell* to *string*. The *string* will be interpreted by the cell as if typed by the user.

the evaluated *string*

```
setinput(c1,"=a1+b1")
```

SETLABEL

Assigns a label name to a range of cells.

SETLABEL(*label*, *range*)

label	any string
range	any worksheet range

This function assigns *range* to the name *label*. *label* must be expressed as a string.

***Returns:** 0*

```
setlabel("myrange",a1:g13)
```

SETPROTECTION

Sets the protection for a given range.

SETPROTECTION(*range*,TRUE|FALSE)

SOLVE

Finds an input value such that a specified formula will give a desired result.

SOLVE(*var*, *comp*, *value* [, *prec*])

var	any cell reference
comp	any cell reference containing a formula
value	the desired result
prec	the [optional] digits of precision

This function tries to find the correct value of the cell *var* such that the formula in *comp* yields *value* (a number). It places the result in *var*. Optionally, the number of digits of precision can be specified with *prec*. The default is 6.

***Returns:** 0 if successful, -1 if not*

```
solve(A1,B1,2.5)
```

```
solve(A1,B1,2.5,8)
```

SORT

Sorts a range.

SORT(*range*, *titles*, SORTBYROW | SORTBYCOL, *key1*, *sorttype* [, *key2*, *sorttype* [, *key3*, *sorttype*]])

range	any worksheet range
titles	a boolean stating if the range has titles
*key*N	the offset of the column to sort by
sorttype	a sort-type constant (see below)

This functions sorts the given *range*. If *titles* is true, the top row of the range is ignored. To sort by rows, use the SORTBYROW constant; to sort by columns, use the SORTBYCOL constant. The *key* is the number of rows or columns that the key row or column is offset from the top or left side of the range. Use ASCENDINGSORT or DESCENDINGSORT to specify the order of the sort. *key2* and *key3* are optional.

Returns: *0*

```
sort(a1:b10,false,SORTBYROW,0,ASCENDINGSORT)
```

SYSTEM

Sends a string to the shell.

SYSTEM(*string*)

string any string that can be interpreted by the shell
This function sends the string *string* to the shell for execution.

Returns: *the return value of the shell command*

Constants

Color Constants

Constant	Red	Green	Blue
BLACKCOLOR	0	0	0
BLUECOLOR	0	0	1
CYANCOLOR	0	1	1
DKGRAYCOLOR	0.333	0.333	0.333
GREENCOLOR	0	1	0
LTGRAYCOLOR	0.666	0.666	0.666
MAGENTACOLOR	1	0	1
WHITECOLOR	1	1	1
YELLOWCOLOR	1	1	0
REDCOLOR	1	0	0

Alignment Constants

Constant	Effect
RIGHTALIGN	Aligns text to right side of cell
LEFTALIGN	Aligns text to left side of cell
CENTERALIGN	Centers text in cell
SMARTALIGN	Aligns strings left, numbers and dates right
FILLALIGN	Makes copies of cell contents to fill the cell
TOPALIGN	Aligns text to top of cell
BOTTOMALIGN	Aligns text to bottom of cell
WRAPALIGN	Wraps text to fit inside width of cell

Formatting Constants

Constant	Effect
GENERALFMT	Displays numbers as typed
FIXEDFMT	Displays numbers in fixed decimal format with the default number of decimal places
SCIENTIFICFMT	Displays numbers in scientific notation
CURRENCYFMT	Displays numbers in the default currency format
COMMAFMT	Displays numbers using commas to separate groups of three digits

CHARTFMT	Displays numbers as a series of plus signs of length equal to the number
PERCENTFMT	Displays numbers in percentage format with the default number of decimal places
TEXTFMT	Displays formulas directly, not as values
HIDDENFMT	Hides numbers
DMYFMT	Treats number as a serial number of a date and displays the date in day-month-year format
DMFMT	Displays the date in day-month format
MYFMT	Displays the date in month-year format
HMSFMT	Number must be fractional; treats number as a time and displays it in twelve-hour hours:minutes-second format
HMFMT	Displays the time in twelve-hour hours:minutes format
DATE1FMT	Displays date in month/day/year format
DATE2FMT	Displays date in month/day format
TIME1FMT	Displays time in 24-hour hour:minutes:seconds format
TIME2FMT	Displays time in 24-hour hour:minutes format

Graph Type Constants

Constant	Meaning
BARGRAPH	Bar chart
STACKEDBARGRAPH	Stacked bar chart
COLGRAPH	Column chart
STACKEDCOLGRAPH	Stacked column chart
SCATTERGRAPH	Scatter chart
XYGRAPH	X-Y chart
AREAGRAPH	Area chart
PIEGRAPH	Pie chart
HILOGRAPH	High/Low chart
HILOCLOSEGRAPH	Hi/Low/Close chart
BAR3DGRAPH	3D bar chart
RIBBONGRAPH	Ribbon chart
PYRAMIDGRAPH	Pyramid chart
SURFACEGRAPH	Surface chart

Error Constants

Constant	Meaning

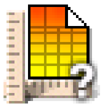
ERR	General error
ERRORCONST	General error

Preferences

The Preferences options allow you to customise Mesa behaviour.

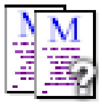
Use *Info->Preferences* to bring up the Preferences panel. Select the Area you want to change.

You can change the overall behaviour of:



Worksheets

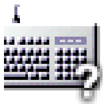
- Currency symbol and position
- Sheet dimensions
- Measurement units



Scripts

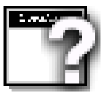
- Autoexecution on startup

Licensing Information



Key functions

- Behaviour of Return
- Behaviour of Enter
- Behaviour of Escape
- Behaviour of cursor keys during editing



Addins

- add or delete addins from Mesa

File and Copying preferences

- copying outside Mesa as EPS graphic or as Ascii
- whether or not to create a Backup file
- how dates look when Exported or Copied

Database Queries

- Show or Hide the Database Password in Queries

QUICKHELP QuickHelp usage

- QuickHelp enabled or disabled
- time before the Help ’balloon’ appears

Excel and 1-2-3 Import/Export

Mesa currently allows export to and import from 1-2-3 WK1 files, and import from Excel V3 files.

P&L Systems is working on an updated Excel import and Export facility.

Registered users of Mesa 2 will be notified of this feature when it is released.

Delimited Text

Use the Worksheet Import/Export Delimited Text options to allow ASCII text delimited in many different ways to be brought into the Mesa sheet.

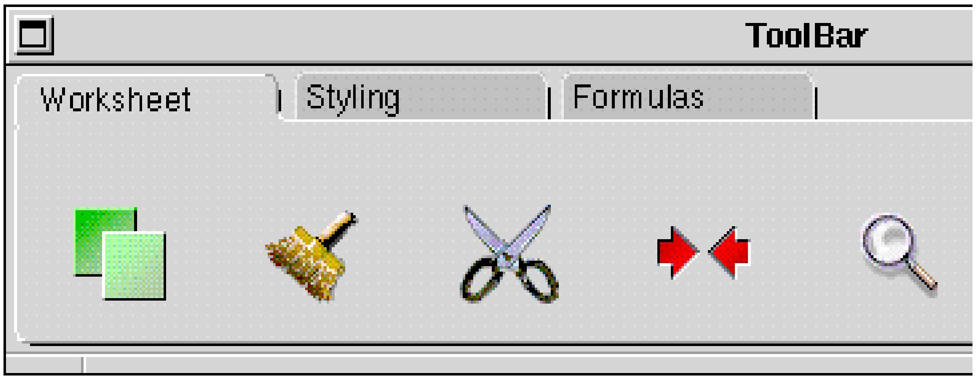
See also

dBASE Addin

ToolBox

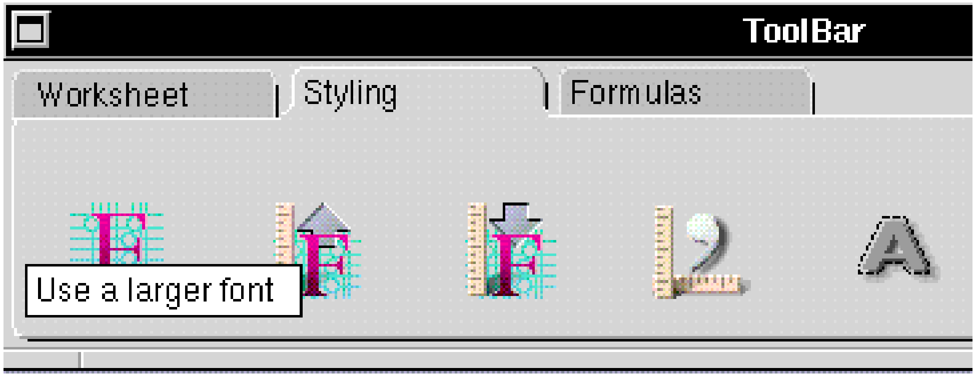
The ToolBox

The ToolBox lets you keep short-cut icons for anything you'd find useful. Bring up the Toolbox with *Tools->Toolbox*, or by selecting the ToolBox icon on the worksheet.



The Toolbox is fully customisable so that you can set up the shortcuts that *you* need. You can also resize it and position it anywhere you like on the screen.

QuickHelp works on all icons in your ToolBox.



Setting up the Toolbox

Adding an icon

Use Command-Drag to drag an icon off a menu. Drag it to the position on the toolbar you want, release the Command key, and then release the mouse button.

Deleting a icon

Drag it outside the ToolBox onto the workspace and let go with the mouse.

Moving an icon

Drag it to the new location. Note that you can't drag icons between Tabs.

Adding a Tab

Click on a Tab name twice (so that the tab area turns dark grey), and press Return. A new tab with a name like 'New2' will be created.

Renaming a Tab

Doubleclick on the Tab Name, and type in the name you want

Deleting a Tab

Click on a Tab name twice (so that the tab area turns dark grey), and press Backspace.   The Tab and all the icons it contains will be removed from your toolbar.

Working with the Toolbox

Setting up the Toolbox

Adding an icon

Use Command-Drag to drag an icon off a menu. Drag it to the position on the toolbar you want, release the Command key, and then release the mouse button.

Deleting a icon

Drag it outside the ToolBox onto the workspace and let go with the mouse.

Moving an icon

Drag it to the new location. Note that you can't drag icons between Tabs.

Adding a Tab

Click on a Tab name twice (so that the tab area turns dark grey), and press Return. A new tab with a name like 'New2' will be created.

Renaming a Tab

Doubleclick on the Tab Name, and type in the name you want

Deleting a Tab

Click on a Tab name twice (so that the tab area turns dark grey), and press Backspace.   The Tab and all the icons it contains will be removed from your toolbar.

Tools on the Worksheet

Just as you can *Command-Drag* an item onto the ToolBar, you can place an item onto the worksheet. This is useful if you have a command or a script that you use at a certain place in the worksheet. For example, if you have a script that validates data in a data entry in a certain range on the worksheet, you can place an MScript button near that range so that you can run the script by clicking on the button.

To place a command or a script on the worksheet, *Command-Drag* the item and drop it onto the worksheet just like you would drop the item onto the ToolBar. Now the tool works just like it does on the menu or on a ToolBar.

To remove a tool from the worksheet, *Command-Control-Drag* a blank tool and drop it onto the tool on the worksheet. To move a tool on the worksheet, *Command-Drag* the tool to a new location.

Drag & Drop Ranges

Mesa allows you to perform many tasks by direct manipulation: grabbing the thing that you want to change and changing it. Drag & Drop ranges are an example of this. You can copy of move a range of cells by dragging them to a new location.

The range A1 through A12 should still be selected from the previous SmartFill operation. Let's move this range to B2 through B13. To do this, move the mouse cursor above cell A1; press and hold the *Control* key; press and hold the mouse button. Mesa will display a shadow of the range. Move the mouse and watch as the shadow range follows the mouse movements. Move the mouse above cell B2; release the mouse button and then release the *Control* key. Mesa will move the range from A1 through A12 to B2 through B13.

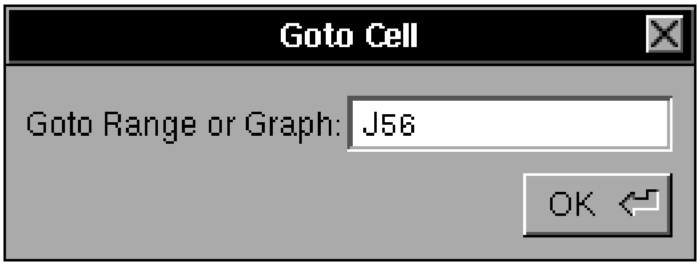
If you want to abort the Drag & Drop operation, release the *Control* key and Mesa will stop the operation.

You can copy a range to another location by *Alt-Dragging* it rather than *Control-Dragging* it. Mesa will make a second copy of the range in the new location.

Special Features
Toolbar
Tools on the Worksheet
SmartFill
Formula Inheritance
Graphing

GoTo

Use *Range->GoTo* to jump directly to any part of the worksheet or to a chart.



You can enter:

- a cell reference (eg J56)
- a Range Label (eg SalesTotal)
- a Chart name (eg FinalFigures)

Formula Inheritance

Mesa has a unique and useful feature called Formula Inheritance. Formula Inheritance applies object oriented concepts to spreadsheets by allowing a set of cells to inherit their formula from a master cell and automatically change the way they calculate when that master cell changes its formula.

Select cells C2 through H13 by clicking in cell C2 and dragging to cell H13 and releasing the mouse button. The range C2 through H13 should be selected and the current cell should be cell C2. Type *=INT(RAND * 100)* and press *Shift-*↵. This will SmartFill the range with this formula that generates a random number between 0 and 99.

When a range is SmartFilled with a formula, Mesa sets the other cells in the range to inherit their formula for the formula in the base cell which is cell C2 in this case. Select cell C2 and change the formula to *=B2 & " Hello"* and press ↵. Note that all the cells in the range changed the **way** that they calculated. Change cell C2 to *=INT(RAND * 1000)*. All the cells in the range C2 through H13 should be displaying a random number between 0 and 999.

You have just performed a SmartFill operation with formulas and taken advantage of Mesa's formula inheritance feature. Formula inheritance makes it easier to build spreadsheets because you only need to change a single formula and the other cells that inherit their formula from the base formula will automatically change the way that they calculate. Formula Inheritance also makes spreadsheet maintenance easier because you do not have to remember all the places that the formula was copied to.

The AddIn Menu

The AddIn menu contains ’extras’ that add to the Mesa core functionality.

Two AddIns come as standard:

- HTML Table Export

and

- DBASE Import/Export

HTML Table Export

Select a range in the worksheet. Choose *Addin->HTML->Export*, and save the range to a file. The extension .html will be added automatically.

Such HTML tables can be accessed directly by any Web browser, or incorporated into larger pages.

NB. Currently Styling features are not exported.

dBASE Export/Import

You can output ranges to dBASE format files, and Import dBASE database files with this Addin.

Note that dates are exported as Strings.

Note also that this addin only works fully on Intel-based systems. If you own a Motorola NeXT or a Sun Sparc, then you can Export to dBASE format but cannot read in a dBASE-created file.

Informant! Addin

Mesa also includes the Informant! Datafeed Addin, allowing dynamic access to DataFeeds using P&L Systems’ Informant! DataFeed Display product.

Other Addins may become available from P&L Systems or Third Parties.

Event Inspector

The Event Inspector is used by some Addins to display or accept information required by the Addin.

Most users will never use this Inspector.

Technical Support

Commercial versions of Mesa include free unlimited technical support. P&L System's technical support lines are open from 9:15 am to 5:30 pm Greenwich Mean Time, Monday through Friday, except holidays.

The technical support phone number is +44-1494-432422. Please have your license number handy when calling for technical support.

You can also request technical support via e-mail. The e-mail address for technical support is mesa@plsys.co.uk, or you can use the **Info** → **Feedback** command to compose and send e-mail directly from Mesa. Mesa automatically includes the version of Mesa and your license number with mail sent from **Info** → **Feedback**.

Tutorial

Introduction

This is a short tutorial that will step you through building a simple check-book worksheet. This worksheet will take advantage of many of Mesa's features including AutoGrow ranges, pre-defined Reports, formulas, formatting, data entry validation, and other Mesa features.

There are six lessons in this chapter. The resulting worksheet at the end of each lesson is located in the *MesaSampler/Tutorial* directory. This will allow you to compare your work to the result of the lesson. It also lets you skip lessons that cover material that you are already familiar with.

Tutorial

AutoGrow Ranges

This lesson will show you how to create an AutoGrow range and do direct data entry. An AutoGrow range is a data entry range that automatically grows when you need to enter more data into it. This lesson will also demonstrate how to do direct data entry rather than using the formula bar at the top of the worksheet window.

First, move to cell A2. Open the Label Inspector via the Inspector Icon on the IconBar. Select the label named *Checking* by clicking on it in the browser. Click on the check-box marked *AutoGrow*. You have just designated this range as an AutoGrow range. Select the *Checking* label and click on the button marked *Goto*. This will select the *checking* range on the worksheet.

Drag the Worksheet Input Area Resize Dimple fully upwards. This allows you to enter and edit data directly in the cell, not in the larger formula bar in the top of the worksheet. Direct Input is useful when doing data entry because you can watch the data that you are entering rather than shifting your eyes between the formula bar and the place that you are doing the data entry in the worksheet.

This lesson has taught you how to mark a range as AutoGrow, how to select a range from the Label Inspector, and how to select Direct Input rather than Formula Bar input. Please save your worksheet. Your worksheet should look similar to the worksheet in *Lesson5.Mesa*.

Tutorial - Creating the Worksheet

The first step is to create a new worksheet. A worksheet is the place that you work on in Mesa. It is a collection of cells organized in rows an columns. Each cell can be individually addressed with its row and column address. Rows are numbered 1 through 16384 and columns are labeled A through XFD. When you first create a worksheet, it will have 32 columns and 40 rows. It will automatically expand as you need more space. You can also make a worksheet larger or smaller by changing the sheet size in the Sheet Inspector.

To create a new worksheet, select the **Worksheet** → **New** (*Command-n*) command. To do this, move the mouse cursor above the **Worksheet** selection on the main menu. Click and release the mouse button. The **Worksheet** submenu will appear. Move the mouse cursor above the **New** selection, click and release the mouse button. Mesa will create a new, blank worksheet.

Many Mesa commands have *Command-Key* equivalents. These commands can be issued by clicking on the desired menu item or by pressing the *Command* key along with another key on the keyboard. Command-Key equivalents are useful shortcuts once you become familiar with an application. The Command-Key equivalent will be listed after each command so that they will become familiar.

Once you have created a new, blank worksheet, you can save it on your computer's disk. Saving a worksheet transfers the information contained in the worksheet from the computer's internal memory to a hard disk or file server. Once the information is on *permanent* storage, it will remain until it is changed or erased.

It is a good idea to save your work from time to time in case there is an unexpected problem with your computer. If you have saved the file, even if the information in the computer's memory is lost, you will still have the information on disk.

To save a worksheet, select the **Worksheet** → **Save** (*Command-s*) command. The first time you save a worksheet, Mesa asks you for the name of the worksheet. Select an appropriate name, for example, *Checking*, enter it into the name field in the Save panel and press ↵.

You can also use the Save icon on the WorkSheet IconBar to save your worksheet.

329 Entering the Column Headings
327 Entering the Formulas
325 Formatting
322 AutoGrow Ranges
331 Reports
333 Summary

Tutorial - Formatting

This lesson will demonstrate some of Mesa's formatting capabilities. Formatting changes the visually appearance of the worksheet to make it more pleasing and more effective at communicating information. Mesa allows you to change the format by direct manipulation, via menu commands, and via inspector panels. This lesson has examples of using the menu items and inspectors to change formatting.

Select cells E2 through G3. Select the **Format->Quick->Display Format** → **Currency** (*Command-Shift-$*) command. This makes all the cells in the range display their numbers in currency format with two decimal places. Select cells B2 through C3 and apply the currency format to those cells as well.

Select cells B4 and C4. Open the Cell Style Inspector using the **IconBar** command. Change the display format from *General* to *Currency* using the pop-up list in the middle of the inspector. You have just changed the formatting of these cells the same way you would have by using the **Display Format** → **Currency** (*Command-Shift-$*) command.

Next, change the view in the Inspectors to the *Border* view by clicking on the button marked *Border* on the left side of the inspector. This will allow you to inspect the border attributes of the selected range. There are four check-boxes organized in a diamond shape in the inspector. These boxes control the border around the cell. Click on the top-center check-box. This will place a thin border on top of the selected range.

Mesa allows you to set the type of input that a cell will accept. This is valuable in order to make sure that data entry is done correctly. By default, Mesa tries to figure out the type of information that is entered into a cell (i.e., number, date, formula, etc.) You can control the type of information entered into a cell by defining the cell's Input Type using the Cell Style Inspector.

Select cells A2 and A3. Change the view on the Cell Style

Inspector to *Input Type*. Change the input type to *Date* by clicking on the date radio button. You have designated this cell as date only. Note that the cell type designation *Date* is in light gray in below the cell number indicator in the upper left corner of the worksheet. Set cells B2 through C3 to *Number* format. Set cells D2 and D3 to *String* format. Set cells E2 and E3 to *Number* format. Finally set cells F2 through G3 to *Formula* format.

You have entered the formatting for this worksheet. You have changed the display format of number cells to display their contents as currency. You have put borders on cells and set the input type of cells. Please save your worksheet. It should look like the *Lesson4.Mesa* worksheet.

322 AutoGrow Ranges
331 Reports
333 Summary

Tutorial - Entering the Formulas

323 Creating the Worksheet
329 Entering the Column Headings

This lesson will teach you about entering formulas and creating labels on the worksheet. Formulas are algebraic statements that can be evaluated by Mesa to return a value in a cell. The formula in one cell can refer to the value in another cell. When the value in a cell changes, the spreadsheet automatically recalculates and updates each cell that relies on a value that was changed.

The first thing that we will do is create a label or a named range called *Checking*. A label can be referred to in a formula by its title rather than the cells addresses. Referring to a label makes formulas more readable. Select cells A2 through G3. Open the Label Inspector by selecting the Inspector Icon on the Worksheet Iconbar and choosing the Label Inspector.   Type *checking* and press ↵. Mesa will create a label called *checking* and assign it to cells A2 through G3.

The next step is to enter the formulas that calculate the balance in the checking account. Move to cell F2 and enter the formula *=B2-C2*. This formula subtracts the amount withdrawn from the amount deposited and places the result in cell F2. Next, move to cell F3, and enter the formula *=F2 + B3 - C3*. This formula added the current transaction to the prior balance, thus accumulating the balance in the checking account.

The next step is to sum the total amount deposited and withdrawn. Select cells B4 and C4. Type *=SUM(checking[][this])* and press *Shift*-↵. This will SmartFill this formula into the selected range. The formula sums, adds up, all the values in the range *checking*, but constrains the range to *this* column. By SmartFilling the range, this formula is automatically propagated into cell C4.

The next step is to enter the formula into the column that alerts you if there is a difference between the amount in the checkbook and the amount that should be in the checkbook. Select cells G2 and G3. Enter the formula

=setcolor(setbkgcolor(if(#NOT#isempty(e2),e2-f2),if(if(#NOT#isem
whitecolor))
,if(if(#NOT#isempty(e2),e2-f2)<>0,whitecolor,blackcolor)) and press *Shift*-↵.

This formula sets the foreground color to white and the background to red if the value in the checkbook is different from the running total.

You have entered all the formulas into the worksheet. Please save your work. Your worksheet should be similar to *Lesson3.Mesa* worksheet.

325 Formatting
322 AutoGrow Ranges
331 Reports
333 Summary

Tutorial - Entering the Column Headings

323 Creating the Worksheet

Now that you have created the worksheet and saved it, you can put information into the worksheet. This section will show you how to put labels on the top of column of data and then *freezing* these titles so they remain on the worksheet even if you scroll to a different section on the worksheet.

The current cell should be cell A1. If it is not, move the mouse cursor on top of cell A1 and press and release the mouse button. Enter *Date* and press the *Tab* key. The string *Date* will appear in cell A1 and Mesa will change the current cell to B1. Enter *Deposit* into cell B1, *Withdrawl* into cell C1, *Reason* into cell D1, *In Book* into cell E1, *Balance* into cell F1, and *Off By* into cell G1. You have entered the text of the titles.

Next, you will put underlines beneath each of the titles to set them off. Select cells A1 through G1. To do this, position the mouse cursor over cell A1. Press and hold the mouse button. Drag the mouse to the right. Notice that Mesa is highlighting each of the cells under the mouse. Move the mouse until cells A1 through G1 are highlighted. Release the mouse button. You have selected cells A1 through G1.

Open the Cell Style Inspector by clicking on the Inspector Icon on the Worksheet Iconbar, and choosing the Cell Inspector. The Cell Style Inspector allows you to examine and change styling elements for the cells in the range that you have selected. The Cell Style Inspector contains a series of views that allow you to view different aspects of the cell's style.

Click on the button marked *Font* on the left side of the inspector. This view allows you to change the font and also change the underlining of the font in the cells. Click on the radio button marked *Wide*. This will place a wide underline beneath the text in the selected cells. Note the change on the screen.

The next step is to make the width of the column containing *Reason* a little wider. Move the mouse so that the mouse cursor is positioned over the line between the column titles *D* and *E*. Note that the cursor changes from an arrow to a vertical bar with two smaller arrows pointing to the left and right. Press and hold the mouse button. Note that Mesa displays the width of the column. Drag the mouse to the right until the column is 1.75 inches wide and release the mouse button. You have made the column wider.

The final step is to freeze the titles so that they will remain at the top of the worksheet even as you scroll through the data on the worksheet. Position the mouse cursor on top of the thick black line that is just beneath the column labels. Note that the cursor has changed to a horizontal bar with a single down arrow. Press and hold the mouse button. Drag the mouse down until the dark grey line is below row 1. Release the mouse button. Mesa has frozen the first row at the top of the worksheet.

Move to cell A2 and save the worksheet by issuing the **Worksheet** → **Save** (*Command-s*) command. You have completed the second lesson in the tutorial. Your worksheet should look like *Lesson2.Mesa* in the *Tutorial* folder.

327 Entering the Formulas
325 Formatting
322 AutoGrow Ranges
331 Reports
333 Summary

Tutorial

Reports

This lesson will show you how to build a simple report and how to print it.

Open the Report Inspector by selecting the ReportWriter icon on the Worksheet IconBar. Mesa will display the Report Selection Panel. Tis allows you to name and define a report that can be printed with the push of a button.

First, we will name the report. Click in the *Name* field in the lower right part of the Report Inspector and type *Checking* and click on the *Add* button. This will add a report named *Checking* to the worksheet.

Next, change the orientation of the report by clicking on the *Landscape* radio button in the *Page Layout* section of the inspector. Select the first line in the *Range* section and type *A2:G999 T=A1:G1*, and click on the *Smart Width* button on that line. This will set the print range to cells A2 through G999 and use the first row as titles on each page. You may question the designation of almost 1,000 lines in the report if you have only filled in a few lines. Mesa automatically ignores rows that do not contain any information when printing, so only the lines that contain information will be printed.

Now, we will add *Tags* to the report. The Report Inspector contains a *virtual piece of paper* onto which you can place *tags* which contain information such as print time and more. You can reduce the scale of the piece of paper using the Scaler at the bottom right of the Page Area.. Drag the tag marked *File Name* from the tags and drop in on the top part of the piece of paper. Mesa will now print the file name on the top of this report every time you print it.

Now click *Save* to add this report to the Report List.

We will now add a print button to the worksheet so that you have easy access to printing this report. Press and hold the *Command* key. Click on the button on the Report List marked *Print*. Drag the button and drop it onto one of the two tool holders in the upper right corner of your worksheet. Every time you click on this button, Mesa will print out the checking report.

This lesson has shown you how to add a report to a worksheet. Please save this worksheet. It should be similar to the *Lesson6.Mesa* worksheet.

333 Summary

Tutorial

Summary

This has been a short tutorial that has stepped you through building a worksheet. You created a new worksheet, entered information and formulas, formatted the information, and created a report.

Mesa has many other features that will help you enter data and present it in a pleasant and informative fashion.

If you have any questions about this tutorial or about using Mesa in general, please contact P&L Systems Technical Support at +44 1494 432422 or by e-mail at feedback@plsys.co.uk.

Index (Click a topic to learn about it)

A

B

C

D

E

F

G

H

IJK

L

M

NO

P

Q

R

S

T

U

V

W

XYZ