Title

stata.com

_docx*() — Generate Office Open XML (.docx) file

Description

_docx*() provides a set of Mata functions to generate Office Open XML (.docx) files compatible with Microsoft Word 2007 and later.

Syntax

Syntax is presented under the following headings:

Create and save .docx file

real scalar _docx_new()

In the rest of the manual entry, *dh* is the value returned by _docx_new().

real scalar _docx_save(*dh*, *string scalar filename* [, *real scalar replace*])

real scalar _docx_append(*dh*, *string scalar filename*)

real scalar _docx_close(*dh*)

void _docx_closeall()

Add paragraph and text

real scalar _docx_paragraph_new(*dh*, *string scalar s*)

real scalar _docx_paragraph_new_styledtext(*dh*, *string scalar s*, *style*)

real scalar _docx_paragraph_add_text(*dh*, *string scalar s*
[, *real scalar nospace*])

real scalar _docx_text_add_text(*dh*, *string scalar s*
[, *real scalar nospace*])

Add image file

real scalar _docx_image_add(*dh*, *string scalar filepath* [, *real scalar link*, *cx*, *cy*])

Add table

real scalar _docx_new_table(*dh*, *real scalar row*, *col* [, *noadd*])

real scalar _docx_add_matrix(*dh*, *string scalar name*, *fmt*, *real scalar colnames*, *rownames* [, *noadd*])

real scalar _docx_add_mata(*dh*, *real matrix m*, *string scalar fmt* [, *real scalar noadd*])

real scalar _docx_add_data(*dh*, *real scalar varnames*, *obsno*, *real matrix i*, *rowvector j* [, *real scalar noadd*, *scalar selectvar*])

_docx_new_table(), _docx_add_matrix(), _docx_add_mata(), and _docx_add_data() return an integer ID *tid* of the new table for future use. In the rest of the manual entry, *tid* is used to denote the integer ID of the table returned by _docx_new_table(), _docx_add_matrix(), _docx_add_mata(), or _docx_add_data().

Edit table

After a table is successfully created, that table identified by *tid* can be edited using the following functions:

real scalar _docx_table_add_row(*dh*, *tid*, *real scalar i*, *count*)

real scalar _docx_table_del_row(*dh*, *tid*, *real scalar i*)

real scalar _docx_table_add_cell(*dh*, *tid*, *real scalar i*, *j* [, *string scalar s*])

real scalar _docx_table_del_cell(*dh*, *tid*, *real scalar i*, *j*)

real scalar _docx_cell_set_colspan(*dh*, *tid*, *real scalar i*, *j*, *count*)

real scalar _docx_cell_set_rowspan(*dh*, *tid*, *real scalar i*, *j*, *count*)

real scalar _docx_cell_set_span(*dh*, *tid*, *real scalar i*, *j*, *rowcount*, *colcount*)

real scalar _docx_table_mod_cell(*dh*, *tid*, *real scalar i*, *j*, *string scalar s* [, *real scalar append*])

real scalar _docx_table_mod_cell_table(*dh*, *tid*, *real scalar i*, *j*, *append*, *src_tid*)

real scalar _docx_table_mod_cell_image(*dh*, *tid*, *real scalar i*, *j*, *string scalar filepath* [, *real scalar link*, *append*, *cx*, *cy*])

Query routines

real scalar	`_docx_query(`*real matrix doc_ids*`)`
real scalar	`_docx_query_table(`*dh*`, `*tid*`)`
real scalar	`_docx_table_query_row(`*dh*`, `*tid*`, `*real scalar i*`)`

Remarks and examples

stata.com

The following sections describe the purpose, input parameters, and return codes of the Mata functions.

Remarks are presented under the following headings:

Detailed description

`_docx_new()` creates an empty `.docx` document in memory.

`_docx_save(`*dh*`, `*filename*` [ , `*replace*` ])` saves the document identified by ID *dh* to file *filename* on disk. The file *filename* is overwritten if *replace* is specified and is not `0`.

`_docx_append(`*dh*`, `*filename*`)` appends the document identified by ID *dh* to file *filename* on disk.

`_docx_close(`*dh*`)` closes the document identified by ID *dh* in memory.

`_docx_closeall()` closes all `.docx` documents in memory.

`_docx_paragraph_new(`*dh*`, `*s*`)` creates a new paragraph with the content specified in *string scalar s*.

`_docx_paragraph_new_styledtext(`*dh*`, `*s*`, `*style*`)` creates a new paragraph with the content specified in *string scalar s*. The text has the style specified in *style*. The styles can be `"Title"`, `"Heading1"`, `"Heading2"`, etc. See http://www.stata.com/docx_styles.html for more discussion on styles.

_docx_paragraph_add_text(*dh*, *s* [, *nospace*]) adds text *s* to the current paragraph. If *nospace* is specified and is not 0, the leading spaces in *s* are trimmed; otherwise, the leading spaces in *s* are preserved.

_docx_text_add_text(*dh*, *s* [, *nospace*]) adds text *s* to the current text. If *nospace* is specified and is not 0, the leading spaces in *s* are trimmed; otherwise, the leading spaces in *s* are preserved.

_docx_image_add(*dh*, *path* [, *link*, *cx*, *cy*]) adds an image file to the document. The *filepath* is the path to the image file.

_docx_new_table(*dh*, *row*, *col* [, *noadd*]) creates an empty table of size *row* by *col*.

_docx_add_matrix(*dh*, *name*, *fmt*, *colnames*, *rownames* [, *noadd*]) adds a matrix in a table to the document and returns the table ID *tid* for future use.

_docx_add_mata(*dh*, *m*, *fmt* [, *noadd*]) adds a Mata matrix in a table to the document and returns the table ID *tid* for future use.

_docx_add_data(*dh*, *varnames*, *obsno*, *i*, *j* [, *noadd*, *selectvar*]) adds the current Stata dataset in memory in a table to the document and returns the table ID *tid* for future use.

_docx_table_add_row(*dh*, *tid*, *i*, *count*) adds a row with *count* columns to the table ID *tid* right after the *i*th row.

_docx_table_del_row(*dh*, *tid*, *i*) deletes the *i*th row from the table.

_docx_table_add_cell(*dh*, *tid*, *i*, *j* [, *s*]) adds a cell to the table ID *tid* right after the *j*th column on the *i*th row.

_docx_table_del_cell(*dh*, *tid*, *i*, *j*) deletes the cell of the table ID *tid* on the *i*th row, *j*th column.

_docx_cell_set_colspan(*dh*, *tid*, *i*, *j*, *count*) sets the cell of the *j*th column on the *i*th row to span horizontally *count* cells to the right.

_docx_cell_set_rowspan(*dh*, *tid*, *i*, *j*, *count*) sets the cell of the *j*th column on the *i*th row to span vertically *count* cells downward.

_docx_cell_set_span(*dh*, *tid*, *i*, *j*, *rowcount*, *colcount*) sets the cell of the *j*th column on the *i*th row to span vertically *rowcount* cells downward and span horizontally *colcount* cells to the right.

_docx_table_mod_cell(*dh*, *tid*, *i*, *j*, *s*[, *append*]) modifies the cell on the *i*th row and *j*th column with text *s*.

_docx_table_mod_cell_table(*dh*, *tid*, *i*, *j*, *append*, *src_tid*) modifies the cell on the *i*th row and *j*th column with a table identified by ID *src_tid*.

_docx_table_mod_cell_image(*dh*, *tid*, *i*, *j*, *filepath* [, *link*, *append*, *cx*, *cy*]) modifies the cell on the *i*th row and *j*th column with an image. The *filepath* is the path to the image file.

_docx_query(*doc_ids*) returns the number of all documents in memory. It stores document IDs in *doc_ids* as a row vector.

_docx_query_table(*dh*, *tid*) returns the total number of rows of table ID *tid* in document ID *dh*.

_docx_table_query_row(*dh*, *tid*, *i*) returns the number of columns of the *i*th row of table ID *tid* in document ID *dh*.

Error codes

Functions can only abort if one of the input parameters does not meet the specification; for example, a string scalar is used when a real scalar is required. Functions return a negative error code when there is an error. The codes specific to _docx_*() functions are the following:

Negative code	Meaning
−16510	an error occurred; document is not changed
−16511	an error occurred; document is changed
−16512	an error occurred
−16513	document ID out of range
−16514	document ID invalid
−16515	table ID out of range
−16516	table ID invalid
−16517	row number out of range
−16518	column number out of range
−16519	no current paragraph
−16520	invalid property value
−16521	too many open documents
−16522	last remaining row of the table cannot be deleted
−16523	last remaining column of the row cannot be deleted
−16524	invalid image file format or image file too big
−16525	function is not supported on this platform
−16526	too many columns
−16527	no current text

Any function that takes a document ID *dh* as an argument may return error codes −16513 or −16514.

Any function that takes a table ID *tid* as an argument may return error codes −16515 or −16516.

Any function that takes a row number as an argument may return error code −16517. Any function that takes a column number as an argument may return error code −16518.

Error code −16511 means an error occurred during a batch of changes being applied to the document. For example, an error may occur when adding a matrix to the document. When this happens, the document is changed, but not all the entries of the matrix are added.

Functions

Create and save .docx file

_docx_new() creates an empty .docx document in memory. The function returns an integer ID *dh* that identifies the document for future use. The function returns a negative error code if an error occurs. The function returns −16521 if there are too many open documents. If this happens, you may use _docx_close() or _docx_closeall() to close one or all documents in memory to fix the problem.

_docx_save(*dh*, *string scalar filename* [, *real scalar replace*])
saves the document identified by ID *dh* to file *filename* on disk. The file *filename* is overwritten if *replace* is specified and is not 0.

Besides the error codes −16513 and −16514 for invalid or out of range document ID *dh*, the function may return the following error codes if *replace* is not specified or if specified as 0:

Code	Meaning
−602	file already exists
−3602	invalid filename

The function may return the following error codes if *replace* is specified and is not 0:

Code	Meaning
−3621	attempt to write read-only file
−3602	invalid filename

_docx_append(*dh*, *string scalar filename*)
appends the document identified by ID *dh* to file *filename* on disk.

Besides the error codes −16513 and −16514 for invalid or out of range document ID *dh*, the function may return the following error codes:

Code	Meaning
−601	file cannot be found or read
−3621	attempt to write read-only file
−3602	invalid filename

_docx_close(*dh*)
closes the document identified by ID *dh* in memory. The function returns error code −16513 if the ID *dh* is out of range.

_docx_closeall()
closes all .docx documents in memory.

Add paragraph and text

_docx_paragraph_new(*dh*, *string scalar s*)
creates a new paragraph with the content specified in *string scalar s*. The function returns 0 if it is successful or returns a negative error code if it fails.

_docx_paragraph_new_styledtext(*dh*, *string scalar s*, *style*)
creates a new paragraph with content *string scalar s*. The text has the style specified in *style*. The styles can be "Title", "Heading1", and "Heading2", etc. See http://www.stata.com/docx_styles.html for more discussion on styles.

After _docx_paragraph_new() and _docx_paragraph_new_styledtext(), the newly created paragraph becomes the current paragraph.

_docx_paragraph_add_text(*dh*, *string scalar s* [, *real scalar nospace*])
adds text *s* to the current paragraph. If *nospace* is specified and is not 0, the leading spaces in *s* are trimmed; otherwise, the leading spaces in *s* are preserved. The function returns 0 if it is successful and returns a negative error code if it fails. It may return −16519 if there is no current paragraph. This case usually happens if this function is called before a _docx_paragraph_new() or _docx_paragraph_new_styledtext() function.

_docx_text_add_text(*dh*, *string scalar s* [, *real scalar nospace*])
adds text *s* to the current text. If *nospace* is specified and is not 0, the leading spaces in *s* are trimmed; otherwise, the leading spaces in *s* are preserved. The function returns 0 if it is successful and returns a negative error code if it fails. It may return −16527 if there is no current text. This case usually happens if this function is called before a _docx_paragraph_add_text() function.

This is a convenience routine so the newly added text can have the same styles as the current text.

Add image

_docx_image_add(*dh*, *string scalar filepath* [, *real scalar link*, *cx*, *cy*])
adds an image file to the document. The *filepath* is the path to the image file. It can be either the full path or the relative path from the current working directory. If *link* is specified and is not 0, the image file is linked; otherwise, the image file is embedded.

The width of the image is controlled by *cx*. The height of the image is controlled by *cy*. *cx* and *cy* are measured in twips. A twip is 1/20 of a point, 1/1440 of an inch, or approximately 1/567 of a centimeter.

If *cx* is not specified or is less than or equal to 0, the default size, which is determined by the image information and the page width of the document, is used. If the *cx* is larger than the page width of the document, the page width is used. Otherwise, the width is *cx* in twips.

If *cy* is not specified or is less than or equal to 0, the height of the image is determined by the width and the aspect ratio of the image; otherwise, the added image has height *cy* in twips.

The function returns 0 if it is successful and returns a negative error code if it fails. The function may return error code −601 if the image file specified by *filepath* cannot be found or read. The function may return error code −16524 if the type of the image file specified by *filepath* is not supported or the file is too big.

The function is not supported on Stata for Mac running on OS X 10.9 (Mavericks) or console Stata for Mac. The function returns error code −16525 if specified on the above platforms.

Add table

_docx_new_table(*dh*, *real scalar row*, *col* [, *noadd*])
creates an empty table of size *row* by *col*. If it is successful, the function returns the table ID *tid*, which is an integer greater than or equal to 0 for future use. The function returns a negative error code if it fails. If *noadd* is specified and is not 0, the table is created but not added to the document. This is useful if the table is intended to be added to a cell of another table.

Microsoft Word 2007/2010 allows a maximum of 63 columns in a table. The function returns error code −16526 if *col* is greater than 63.

_docx_add_matrix(*dh*, *string scalar name*, *fmt*, *real scalar colnames*, *rownames* [, *noadd*])
adds a matrix in a table to the document and returns the table ID *tid* for future use. The elements of the matrix are formatted using *fmt*. If *fmt* is not a valid Stata numeric format, %12.0g is used. If *colnames* is not 0, the first row of the table is filled with matrix colnames. If *rownames* is not 0, the first column of the table is filled with matrix rownames. If *noadd* is specified and is not 0, the table is created but not added to the document. This is useful if the table is intended to be added to a cell of another table.

The function returns a negative error code if it fails. The function may return −111 if the matrix specified by *name* cannot be found. The function may return error code −16511 if the table has been added and an error occurred while filling the table. In this case, the document is changed but the operation is not entirely successful. The function returns error code −16526 if the number of columns of the matrix is greater than 63.

_docx_add_mata(*dh*, *real matrix m*, *string scalar fmt* [, *real scalar noadd*])
adds a Mata matrix in a table to the document and returns the table ID *tid* for future use. The elements of the Mata matrix are formatted using *fmt*. If *fmt* is not a valid Stata numeric format, %12.0g is used. If *noadd* is specified and is not 0, the table is created but not added to the document. This is useful if the table is intended to be added to a cell of another table.

The function returns a negative error code if it fails. The function may return error code −16511 if the table has been added and an error occurred while filling the table. In this case, the document is changed, but the operation is not entirely successful. The function returns error code −16526 if the number of columns of the matrix is greater than 63.

_docx_add_data(*dh*, *real scalar varnames*, *obsno*, *real matrix i*, *rowvector j* [, *real scalar noadd*, *scalar selectvar*])
adds the current Stata dataset in memory in a table to the document and returns the table ID *tid* for future use. If there is a value label attached to the variable, the variable is displayed according to the value label. Otherwise, the variable is displayed according to its format. *i*, *j*, and *selectvar* are specified in the same way as with st_data(). Factor variables and time-series-operated variables are not allowed. If *varnames* is not 0, the first row of the table is filled with variable names. If *obsno* is not 0, the first column of the table is filled with observation numbers. If *noadd* is specified and is not 0, the table is created but not added to the document. This is useful if the table is intended to be added to a cell of another table.

The function returns a negative error code if it fails. The function may return error code −16511 if the table has been added and an error occurred while filling the table. In this case, the document is changed, but the operation is not entirely successful. The function outputs missing . or empty string "" if *i* or *j* is out of range; it does not abort with an error. The functions returns error code −16526 if the number of variables is greater than 63.

Edit table

_docx_table_add_row(*dh*, *tid*, *real scalar i*, *count*)
adds a row with *count* columns to the table ID *tid* right after the *i*th row. The range of *i* is from 0 to *r*, where *r* is the number of rows of the table. Specifying *i* as 0 adds a row before the first row, which is equivalent to adding a new first row; specifying *i* as *r* adds a row right after the last row, which is equivalent to adding a new last row. The function returns error code −16517 if *i* is out of range.

_docx_table_del_row(*dh*, *tid*, *real scalar i*)
deletes the *i*th row from the table. The range of *i* is from 1 to *r*, where *r* is the number of rows of the table. The function returns error code −16517 if *i* is out of range. If the table has only one row, the function returns error code −16522, and the row is not deleted. This is to ensure the document can be properly displayed in Microsoft Word.

_docx_table_add_cell(*dh*, *tid*, *real scalar i*, *j* [, *string scalar s*])
adds a cell to the table ID *tid* right after the *j*th column on the *i*th row. The range of *i* is from 1 to *r*, where *r* is the number of rows of the table. The range of *j* is from 0 to *c*, where *c* is the number of columns of the *i*th row. Specifying *j* as 0 adds a cell to the first column on the row; specifying *j* as *c* adds a cell to the last column on the row. The function returns error code −16517 if *i* is out of range. The function returns error code −16518 if *j* is out of range.

_docx_table_del_cell(*dh*, *tid*, *real scalar i*, *j*)
deletes a cell from the table *tid* on the *i*th row, *j*th column. The range of *i* is from 1 to *r*, where *r* is the number of rows of the table. The range of *j* is from 1 to *c*, where *c* is the number of columns of the *i*th row. The function returns error code −16517 if *i* is out of range. The function returns error code −16518 if *j* is out of range. If the row has only one column, the function returns error code −16523, and the column is not deleted. This is to ensure the document can be properly displayed in Microsoft Word.

_docx_cell_set_colspan(*dh*, *tid*, *real scalar i*, *j*, *count*)
sets the cell of the *j*th column on the *i*th row to span horizontally *count* cells to the right. This is equivalent to merging *count* − 1 cells right of the cell on the same row into that cell. If *j*+*count* − 1 is larger than *c*, where *c* is the total number of columns of the *i*th row, the span stops at the last column. The function returns error code −16517 if *i* is out of range. The function returns error code −16518 if *j* is out of range or if *count* is less than 1.

_docx_cell_set_rowspan(*dh*, *tid*, *real scalar i*, *j*, *count*)
sets the cell of the *j*th column on the *i*th row to span vertically *count* cells downward. This is equivalent to merging *count* − 1 cells below the cell on the same column into it. If *i*+*count* − 1 is larger than *r* where *r* is the total number of rows of the table, the span stops at the last row. The function returns error code −16517 if *i* is out of range or if *count* is less than 1. The function returns error code −16518 if *j* is out of range.

_docx_cell_set_span(*dh*, *tid*, *real scalar i*, *j*, *rowcount*, *colcount*)
sets the cell of the *j*th column on the *i*th row to span vertically *rowcount* cells downward and span horizontally *colcount* cells to the right. The function returns error code −16517 if *i* is out of range. The function returns error code −16518 if *j* is out of range.

_docx_table_mod_cell(*dh*, *tid*, *real scalar i*, *j*, *string scalar s*[, *real scalar append*])
modifies the cell on the *i*th row and *j*th column with text *s*. If *append* is specified and is not 0, text *s* is appended to the current content of the cell; otherwise, text *s* replaces the current content of the cell. The function returns 0 if it is successful and returns a negative error code if it fails.

_docx_table_mod_cell_table(*dh*, *tid*, *real scalar i*, *j*, *append*, *src_tid*) modifies the cell on the *i*th row and *j*th column with a table identified by ID *src_tid*. If *append* is not 0, table *src_tid* is appended to the current content of the cell; otherwise, table *src_tid* replaces the current content of the cell. The function returns error code −16515 or −16516 if *src_id* is out of range or invalid.

_docx_table_mod_cell_image(*dh*, *tid*, *real scalar i*, *j*, *string scalar filepath* [, *real scalar link*, *append*, *cx*, *cy*])
modifies the cell on the *i*th row and *j*th column with an image. The *filepath* is the path to the image file. It can be either the full path or the relative path from the current working directory. If link is specified and is not 0, the image file is linked; otherwise, the image file is embedded.

cx and *cy* specify the width and the height of the image. *cx* and *cy* are measured in twips. A twip is 1/20 of a point, is 1/1440 of an inch, or approximately 1/567 of a centimeter.

If *cx* is not specified or is less than or equal to 0, the width of cell (i, j) is used; otherwise, the image has width *cx* in twips.

If *cy* is not specified or is less than or equal to 0, the height of the image is determined by the width and the aspect ratio of the image; otherwise, the image has height *cy* in twips.

If *append* is not 0, the image is appended to the current content of the cell; otherwise, the image replaces the current content of the cell.

The function returns error code −601 if the image file specified by *filepath* cannot be found or read. The function may return error code −16524 if the type of the image file specified by *filepath* is not supported or the file is too big.

The function is not supported on Stata for Mac running on OS X 10.9 (Mavericks) or console Stata for Mac. The function returns error code −16525 if specified on the above platforms.

Query routines

_docx_query(*real matrix doc_ids*)
The function returns the number of all documents in memory. It stores document IDs in *doc_ids* as a row vector. If there is no document in memory, the function returns 0 and *doc_ids* is not changed.

_docx_query_table(*dh*, *tid*)
returns the total number of rows of table ID *tid* in document ID *dh*. The function returns a negative error code if either *dh* or *tid* is invalid.

_docx_table_query_row(*dh*, *tid*, *real scalar i*)
returns the number of columns of the *i*th row of table ID *tid* in document ID *dh*. The function returns a negative error code if either *dh* or *tid* is invalid. The function returns a negative error code if *i* is out of range.

Save document to disk file

A document in memory can be saved and resaved. It stays in memory and can be modified as long as the document is not closed. We do not support loading and modifying existing Word documents from disk into memory at this time.

Current paragraph and text

When a paragraph is added, it becomes the current paragraph. The subsequent _docx_paragraph_add_text() call will add text to the current paragraph. When _docx_paragraph_add_text() is called, the newly added text becomes the current text. Functions changing paragraph styles are applied to the current paragraph. Functions changing text styles are applied to the current text.

When the current paragraph and text change, there is no way to go back. We do not have functions to move around the document. The only exception to this is the table. The table is identified by its ID and can be accessed at any time.

A new paragraph is always created when you add a table or an image. The table is added to the new paragraph, and this paragraph becomes the current paragraph.

Supported image types

Images of types `.emf`, `.png`, and `.tiff` are supported. Images of types `.wmf`, `.pdf`, `.ps`, and `.eps` are not supported.

Linked and embedded images

The image is linked into the document if the *link* parameter is specified and is not 0 in `_docx_image_add()` and `_docx_table_mod_cell_image()`; otherwise, the image is embedded.

If the image is embedded, it becomes a part of the document and has no further relationship with the original image on the disk. If the image is linked, only a link to the image file is inserted into the document. The image file must be present so that the Word document can display the image.

If the Word document is moved to a different machine, all embedded images will display fine; all linked images will require the image files to be moved to the same directory of the Word document on the new machine to be displayed correctly.

If the original image file on the disk is updated, the linked image in the Word document will reflect the change; the embedded image will not.

Styles

A wide range of styles—for example, font, color, text size, table width, justification—is supported. For a list of functions related to styles, see http://www.stata.com/docx_styles.html.

Performance

Creating a new document in a new session of Stata can cause some noticeable delay, usually several seconds.

Examples

Create a .docx document in memory

In the following example, we create a Microsoft Word document using Stata data, results from a Stata estimation command, and Stata graphs.

We create a new `.docx` document in memory by calling `_docx_new()`.

```
mata:
dh = _docx_new()
end
```

It is good practice to check if *dh* is negative, which means the document has not been successfully created.

Add paragraphs and text

After the document is successfully created, we can add paragraphs and text to it. We start by adding a title, a subtitle, and a heading.

```
mata:
_docx_paragraph_new_styledtext(dh, "Sample Document", "Title")
_docx_paragraph_new_styledtext(dh, "by Stata", "Subtitle")
_docx_paragraph_new_styledtext(dh, "Add", "Heading1")
end
```

The document ID *dh* is returned from previously calling `_docx_new()`.

Each function returns a real scalar. A negative return code indicates the function failed with error. If you would like to suppress the display of the return code, simply put `(void)` in front of the function.

Now we add a regular paragraph and some text to the document.

```
mata:
_docx_paragraph_new(dh, "Use auto dataset. ")
_docx_paragraph_add_text(dh, "Use -regress- with ")
_docx_paragraph_add_text(dh, "variables -mpg price foreign-.")
end
```

The function `_docx_paragraph_add_text()` can be used to break long sentences into pieces.

Display data

We use `auto.dta` in the rest of the examples.

```
. sysuse auto
```

We may use `_docx_add_data()` to display observations 1–10 of variables `mpg`, `price`, and `foreign` to the document as a table.

```
mata:
_docx_add_data(dh, 1, 1, (1,10), ("mpg", "price", "foreign"))
end
```

The table's first row contains variable names, and the first column contains observation numbers.

Display regression results

After running a regression,

```
. regress mpg price foreign
```

the output contains the following in the header:

```
Number of obs =      74
F(  2,    71) =   23.01
Prob > F      =  0.0000
R-squared     =  0.3932
Adj R-squared =  0.3761
Root MSE      =  4.5696
```

The regression table looks like this:

mpg	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
price	-.000959	.0001815	-5.28	0.000	-.001321	-.000597
foreign	5.245271	1.163592	4.51	0.000	2.925135	7.565407
_cons	25.65058	1.271581	20.17	0.000	23.11512	28.18605

We want to replicate the output in the document as two tables.

First, we replicate the header. We add an empty 6×2 table by using `_docx_new_table()`, then modify each cell of the table by using `_docx_table_mod_cell()` with the stored results in `e()` to replicate the above output. Note that `Prob > F` is not stored but computed by

```
Ftail(e(df_m), e(df_r), e(F))
```

Also note the use of `sprintf()` to format the numeric values to string.

```
mata:
tid = _docx_new_table(dh, 6, 2)
_docx_table_mod_cell(dh, tid, 1, 1, "Number of obs")
result = sprintf("%g", st_numscalar("e(N)"))
_docx_table_mod_cell(dh, tid, 1, 2, result)
result = sprintf("F(%g, %g)",
                          st_numscalar("e(df_m)"),
                          st_numscalar("e(df_r)"))
_docx_table_mod_cell(dh, tid, 2, 1, result)
result = sprintf("%8.2g", st_numscalar("e(F)"))
_docx_table_mod_cell(dh, tid, 2, 2, result)
_docx_table_mod_cell(dh, tid, 3, 1, "Prob > F")
prob = Ftail(st_numscalar("e(df_m)"),
                          st_numscalar("e(df_r)"),
                          st_numscalar("e(F)"))
result = sprintf("%10.4g", prob)
_docx_table_mod_cell(dh, tid, 3, 2, result)
_docx_table_mod_cell(dh, tid, 4, 1, "R-squared")
result = sprintf("%10.4g", st_numscalar("e(r2)"))
_docx_table_mod_cell(dh, tid, 4, 2,result)
_docx_table_mod_cell(dh, tid, 5, 1, "Adj R-squared")
result = sprintf("%10.4g", st_numscalar("e(r2_a)"))
_docx_table_mod_cell(dh, tid, 5, 2, result)
_docx_table_mod_cell(dh, tid, 6, 1, "Root MSE")
result = sprintf("%10.4g", st_numscalar("e(rmse)"))
_docx_table_mod_cell(dh, tid, 6, 2, result)
end
```

To replicate the regression table, we store the numeric values in `r(table)`. But `r(table)` is in the transposed form and contains extra rows, and all row and column names are not what we want. We extract the stored results from `r(table)` by typing

```
mat define r_table = r(table)'
mat r_table = r_table[1..3, 1..6]
```

Then we add the extracted matrix `r_table` to the document by using `_docx_add_matrix`:

```
mata:
tid = _docx_add_matrix(dh, "r_table", "%10.0g", 1, 1)
end
```

Notice that we are including the row and column names although they are not what we want. We modify them by using _docx_table_mod_cell():

```
mata:
_docx_table_mod_cell(dh, tid, 1, 1, "mpg")
_docx_table_mod_cell(dh, tid, 1, 2, "Coef.")
_docx_table_mod_cell(dh, tid, 1, 3, "Std. Err.")
_docx_table_mod_cell(dh, tid, 1, 4, "t")
_docx_table_mod_cell(dh, tid, 1, 5, "P>|t|")
_docx_table_mod_cell(dh, tid, 1, 6, "[95% Conf. Interval]")
end
```

We set the last column of the first row in the regression table to have a column span of 2 to match the Stata output by typing

```
mata:
_docx_cell_set_colspan(dh, tid, 1, 6, 2)
end
```

Add an image

To add a graph to the document, we first need to export the Stata graph to an image file of type .emf, .png, or .tif.

```
. scatter mpg price
. graph export auto.png
```

Then we can add the image to the document by using _docx_image_add():

```
mata:
_docx_image_add(dh, "auto.png")
end
```

Display nested table

If we want to output something like the table below to the document,

	mpg
price	−0.001
	(5.28)**
foreign	5.245
	(4.51)**
_cons	25.651
	(20.17)**
R^2	0.39
	74

* $p < 0.05$; ** $p < 0.01$

we can either create a 10×2 table and fill in the content or build it in pieces and combine the pieces.

Notice that the middle part of the table for each variable has a similar pattern. First, we run the regression and get the saved table.

```
. regress mpg price foreign
```

Then in Mata, we can build a 2×2 table for each variable by coding

```
mata:
mr_table = st_matrix("r(table)")
colnames = st_matrixcolstripe("r(table)")
tids = J(1, cols(mr_table), .)

for(i=1; i<=cols(mr_table); i++) {
        tids[i] = _docx_new_table(dh, 2, 2, 1)
        _docx_table_mod_cell(dh, tids[i], 1, 1, colnames[i, 2])
        output = sprintf("%10.0g", mr_table[1, i])

        _docx_table_mod_cell(dh, tids[i], 1, 2, output)
        if(mr_table[1, i]<0) {
output = sprintf("(%10.0g)", mr_table[3, i])
        }
        else {
                output = sprintf("%10.0g", mr_table[3, i])
        }

        if(mr_table[4, i]<0.05) {
                output = output +"*"
        }

        if(mr_table[4, i]<0.01) {
                output = output +"*"
        }

        _docx_table_mod_cell(dh, tids[i], 2, 2, output)
        _docx_cell_set_rowspan(dh, tids[i], 1, 1, 2)
}
end
```

Now we can combine them with the header and bottom three rows.

```
mata:
tid = _docx_new_table(dh, cols(mr_table)+4, 2)
_docx_table_mod_cell(dh, tid, 1, 2, "mpg")

for(i=2; i<=cols(mr_table)+1; i++) {
        _docx_cell_set_colspan(dh, tid, i, 1, 2)
        _docx_table_mod_cell_table(dh, tid, i, 1,
                0, tids[i-1])
}

_docx_table_mod_cell(dh, tid, cols(mr_table)+2, 1, "R2")
output = sprintf("%10.4g", st_numscalar("e(r2)"))
_docx_table_mod_cell(dh, tid, cols(mr_table)+2, 2, output)

_docx_table_mod_cell(dh, tid, cols(mr_table)+3, 1, "N")
output = sprintf("%10.4g", st_numscalar("e(N)"))
_docx_table_mod_cell(dh, tid, cols(mr_table)+3, 2, output)

_docx_table_mod_cell(dh, tid, cols(mr_table)+4, 1,
        "* p<0.05; ** p<0.01")
_docx_cell_set_colspan(dh, tid, cols(mr_table)+4, 1, 2)
end
```

Add images to table cells

We can also add an image to a table cell. First, we create the images:

```
. histogram price, title("Prices")
. graph export prices.png
. histogram mpg, title("MPG")
. graph export mpg.png
```

We can add auto0.png and auto1.png to different cells of a table by using _docx_table_mod_cell_image().

```
mata:
tid = _docx_new_table(dh, 1, 2)
_docx_table_mod_cell_image(dh, tid, 1, 1, "prices.png")
_docx_table_mod_cell_image(dh, tid, 1, 2, "mpg.png")
end
```

Save the .docx document in memory to a disk file

We use _docx_save() to save the document to a disk file:

```
mata:
res = _docx_save(dh, "example.docx")
end
```

Notice that we did not specify the third parameter *replace*; hence, the function may fail if example.docx already exists in the current working directory. It is always good practice to check the return code of _docx_save().

Diagnostics

See *Remarks and examples*.

References

Gallup, J. L. 2012. A programmer's command to build formatted statistical tables. *Stata Journal* 12: 655–673.

Lo Magno, G. L. 2013. Sar: Automatic generation of statistical reports using Stata and Microsoft Word for Windows. *Stata Journal* 13: 39–64.

Quintó, L. 2012. HTML output in Stata. *Stata Journal* 12: 702–717.

Also see

[M-4] **io** — I/O functions

[M-5] **Pdf*()** — Create a PDF file

[M-5] **xl()** — Excel file I/O class

[P] **putdocx** — Generate Office Open XML (.docx) file

[P] **putexcel** — Export results to an Excel file

[P] **putpdf** — Create a PDF file