# Creating LaTeX and HTML documents from within Stata using texdoc and webdoc

## Example 3

Ben Jann
University of Bern, ben.jann@soz.unibe.ch

Swiss Stata Users Group meeting
Bern, November 17, 2016

## Contents

## 1 Introduction

This example illustrates how to implement literate programming using `texdoc`. The procedure is as follows:

1. Within the ado file containing your program, use `/*** ***/` blocks to include notes and explanations. Stata will treat these blocks as comments when loading the program (i.e. they will not affect the definition and functionality of the program). However, do *not* use any `texdoc` commands within the ado file!
2. Maintain a do file containing `texdoc` commands to produce the documentation of your program. Within this do file, include a `texdoc do` command to process the the ado file, using options `logall` (to document all Stata code found in the ado file), `cmdlog` (to document code instead of output), and `nodo` (to prevent Stata from running the commands found in the ado file). Within this do file, you may also include examples illustrating the usage of your program.
3. Process the do file by `texdoc do` and compile the resulting LaTeX file.

## 2 The annotated ado file

— *myxtsum.ado* —

```
*! version 1.0.0  11nov2016  Ben Jann
program myxtsum, eclass
    version 9.2
/***
    We have to specify the \stcmd{eclass} option because we want to return
    results in \stcmd{e()}. Version control is set to 9.2 so that also users of
    older versions of Stata can use the program.

    With respect to syntax, all we need to allow is a variable list and the
    if-qualifier. The sample is restricted to all observations that satisfy the
    if-qualifier and are non-missing for all specified variables (list-wise
    exclusion).
***/
    syntax varlist [if]
    marksample touse
/***
    Next the list of statistics to be collected is defined and a temporary name
    is created for each statistic. These names will be used below as names for
    the matrices to collect the results.
***/
    local stats mean sd min max sd_b min_b max_b sd_w min_w max_w
    foreach s of local stats {
        tempname `s'
    }
/***
    Now we can run \stcmd{xtsum} for each variable and collect the results from
    \stcmd{r()} by writing them into matrices. Matrix function
    \stcmd{nullmat()} allows appending results to a matrix that does not yet
    exist. This is necessary to prevent error when processing the first
    variable.
***/
    foreach v of local varlist {
        quietly xtsum `v' if `touse'
        foreach s of local stats {
            matrix ``s'' = nullmat(``s''), r(`s')
        }
    }
/***
    Technical note: Appending results to matrices is bad programming
    style as it is computationally inefficient. A much better approach would be
    to first define all matrices and then fill them in element-by-element.

    At the end we collect some additional overall statistics. Since these
    statistics are the same for all variables, we just use the results from the
    last variable.
***/
    foreach s in N n Tbar {
        local `s' = r(`s')
    }
/***
    The results matrices have no column names yet, so let's add some names.
***/
    foreach s of local stats {
        matrix coln ``s'' = `varlist'
    }
/***
    Now we copy the means into an additional matrix to be posted in
    \stcmd{e(b)}. This is not strictly needed, but it may be convenient to have
    \stcmd{e(b)} defined.
***/
    tempname b
    matrix `b' = `mean'
/***
    In the last section of the program the collected results are posted in
    \stcmd{e()}.
***/
    eret post `b', obs(`N') esample(`touse')
    eret local cmd "myxtsum"
    eret scalar n = `n'
    eret scalar Tbar = `Tbar'
    foreach s of local stats {
        eret matrix `s' = ``s''
    }
end
```

— *end of file* —

## 3 The texdoc source file

— *myxtsum.texdoc* —

```
clear all
texdoc init myxtsum.tex, logdir(log) replace
```

```
/***
\documentclass[a4paper]{article}
\usepackage{fullpage,stata}
\usepackage{textcomp}
\makeatletter
\g@addto@macro\stLogSetup{\let\'=\textquotesingle}
\makeatother
\let\OLDstlog\stlog \def\stlog{\par\OLDstlog}
\parindent0pt \parskip0.6\baselineskip

\begin{document}
\title{The myxtsum command}
\author{Ben Jann}
\maketitle

\begin{abstract}
    The \stcmd{myxtsum} program applies \stcmd{xtsum} to the specified
    variables and collects the results in \stcmd{e()} so that they can be
    tabulated using \stcmd{esttab}.
\end{abstract}

\section*{Explanation of the program}

The code of \stcmd{myxtsum} starts as follows:

***/

texdoc do myxtsum.ado, logall cmdlog nodo noltrim

/***

\section*{Example usage}

The following example illustrates how \stcmd{myxtsum} can be used to tabulate
some variables from the \stcmd{nlswork} dataset:

***/

texdoc stlog, certify
webuse nlswork
myxtsum hours birth_yr age grade race
esttab, stats(N n Tbar) cells((mean sd min max sd_b sd_w)) ///
    nomtitles nonumbers compress
texdoc stlog close

/***

\end{document}
***/
```

— *end of file* —

Note the usage of option `certify` for the code of the example application. The option will check whether the output is still the same when you rerun the do file (e.g. after making some changes to your ado file). It will display an error if the output changed.

## 4 The resulting LaTeX source file

Applying

```
. texdoc do myxtsum.texdoc
```

generates to the following LaTeX file.

— *myxtsum.tex* —

```
\documentclass[a4paper]{article}
\usepackage{fullpage,stata}
\usepackage{textcomp}
\makeatletter
\g@addto@macro\stLogSetup{\let\'=\textquotesingle}
\makeatother
\let\OLDstlog\stlog \def\stlog{\par\OLDstlog}
\parindent0pt \parskip0.6\baselineskip

\begin{document}
\title{The myxtsum command}
\author{Ben Jann}
\maketitle

\begin{abstract}
    The \stcmd{myxtsum} program applies \stcmd{xtsum} to the specified
    variables and collects the results in \stcmd{e()} so that they can be
    tabulated using \stcmd{esttab}.
\end{abstract}

\section*{Explanation of the program}

The code of \stcmd{myxtsum} starts as follows:

\begin{stlog}\input{log/1.log.tex}\end{stlog}
    We have to specify the \stcmd{eclass} option because we want to return
    results in \stcmd{e()}. Version control is set to 9.2 so that also users of
    older versions of Stata can use the program.

    With respect to syntax, all we need to allow is a variable list and the
    if-qualifier. The sample is restricted to all observations that satisfy the
    if-qualifier and are non-missing for all specified variables (list-wise
    exclusion).
\begin{stlog}\input{log/2.log.tex}\end{stlog}
    Next the list of statistics to be collected is defined and a temporary name
    is created for each statistic. These names will be used below as names for
    the matrices to collect the results.
\begin{stlog}\input{log/3.log.tex}\end{stlog}
    Now we can run \stcmd{xtsum} for each variable and collect the results from
    \stcmd{r()} by writing them into matrices. Matrix function
    \stcmd{nullmat()} allows appending results to a matrix that does not yet
    exist. This is necessary to prevent error when processing the first
    variable.
\begin{stlog}\input{log/4.log.tex}\end{stlog}
    Technical note: Appending results to matrices is bad programming
    style as it is computationally inefficient. A much better approach would be
    to first define all matrices and then fill them in element-by-element.

    At the end we collect some additional overall statistics. Since these
    statistics are the same for all variables, we just use the results from the
    last variable.
\begin{stlog}\input{log/5.log.tex}\end{stlog}
    The results matrices have no column names yet, so let's add some names.
\begin{stlog}\input{log/6.log.tex}\end{stlog}
    Now we copy the means into an additional matrix to be posted in
    \stcmd{e(b)}. This is not strictly needed, but it may be convenient to have
    \stcmd{e(b)} defined.
\begin{stlog}\input{log/7.log.tex}\end{stlog}
    In the last section of the program the collected results are posted in
    \stcmd{e()}.
\begin{stlog}\input{log/8.log.tex}\end{stlog}

\section*{Example usage}

The following example illustrates how \stcmd{myxtsum} can be used to tabulate
some variables from the \stcmd{nlswork} dataset:

\begin{stlog}\input{log/9.log.tex}\end{stlog}

\end{document}
```

— *end of file* —

## 5 The resulting PDF

The following pages display the resulting PDF after compiling the LaTeX source file.

# The myxtsum command

Ben Jann

November 17, 2016

**Abstract**

The myxtsum program applies xtsum to the specified variables and collects the results in e() so that they can be tabulated using esttab.

## Explanation of the program

The code of myxtsum starts as follows:

```
*! version 1.0.0  11nov2016  Ben Jann
program myxtsum, eclass
    version 9.2
```

We have to specify the eclass option because we want to return results in e(). Version control is set to 9.2 so that also users of older versions of Stata can use the program.

With respect to syntax, all we need to allow is a variable list and the if-qualifier. The sample is restricted to all observations that satisfy the if-qualifier and are non-missing for all specified variables (list-wise exclusion).

```
    syntax varlist [if]
    marksample touse
```

Next the list of statistics to be collected is defined and a temporary name is created for each statistic. These names will be used below as names for the matrices to collect the results.

```
    local stats mean sd min max sd_b min_b max_b sd_w min_w max_w
    foreach s of local stats {
        tempname `s'
    }
```

Now we can run xtsum for each variable and collect the results from r() by writing them into matrices. Matrix function nullmat() allows appending results to a matrix that does not yet exist. This is necessary to prevent error when processing the first variable.

```
    foreach v of local varlist {
        quietly xtsum `v' if `touse'
        foreach s of local stats {
            matrix ``s'' = nullmat(``s''), r(`s')
        }
    }
```

Technical note: Appending results to matrices is bad programming style as it is computationally inefficient. A much better approach would be to first define all matrices and then fill them in element-by-element.

At the end we collect some additional overall statistics. Since these statistics are the same for all variables, we just use the results from the last variable.

```
        foreach s in N n Tbar {
            local `s' = r(`s')
        }
```

The results matrices have no column names yet, so let's add some names.

```
        foreach s of local stats {
            matrix coln ``s'' = `varlist'
        }
```

Now we copy the means into an additional matrix to be posted in `e(b)`. This is not strictly needed, but it may be convenient to have `e(b)` defined.

```
        tempname b
        matrix `b' = `mean'
```

In the last section of the program the collected results are posted in `e()`.

```
        eret post `b', obs(`N') esample(`touse')
        eret local cmd "myxtsum"
        eret scalar n = `n'
        eret scalar Tbar = `Tbar'
        foreach s of local stats {
            eret matrix `s' = ``s''
        }
    end
```

## Example usage

The following example illustrates how `myxtsum` can be used to tabulate some variables from the `nlswork` dataset:

```
. webuse nlswork
(National Longitudinal Survey.  Young Women 14-26 years of age in 1968)
. myxtsum hours birth_yr age grade race
. esttab, stats(N n Tbar) cells((mean sd min max sd_b sd_w)) ///
>     nomtitles nonumbers compress
----------------------------------------------------------------------
                mean         sd        min        max       sd_b       sd_w
----------------------------------------------------------------------
hours        36.5574   9.870955          1        168   7.851648   7.521153
birth_yr    48.08579   3.012239         41         54   3.052058          0
age         29.03931   6.701764         14         46   5.487027   5.170993
grade       12.53469   2.324229          0         18   2.565939          0
race        1.303224    .482062          1          3   .4855582          0
----------------------------------------------------------------------
N              28441
n               4707
Tbar        6.042277
----------------------------------------------------------------------
```