CIVET
Contentious Incident Variable Entry Template
- - - DRAFT - - - *

Philip A. Schrodt
Parus Analytics
Charlottesville, VA
schrodt735@gmail.com

Version 0.2 : March 20, 2015

*Acknowledgements: The development of CIVET is funded by the U.S. National Science Foundation Office of Multidisciplinary Activities in the Directorate for Social, Behavioral & Economic Sciences, Award 1338470 and the Odum Institute at the University of North Carolina at Chapel Hill with additional assistance from Parus Analytics.

1 Introduction

This is the documentation for a prototype of the CIVET[1]—Contentious Incident Variable Entry Template—customizable data entry system. CIVET is being developed by the NSF-sponsored project titled "A Method for Leveraging Public Information Sources for Social Science Research" which is creating tools to improve the efficiency of data generation in the social sciences. The project has an initial focus on coding event data in the domain of contentious politics, but we expect that these tools will be relevant in a number of data-generation domains.

The core objective of CIVET is to provide a reasonably simple—yes, simple—set of commands that will allow a user to set up a web-based coding environment without the need to master the likes of HTML, CSS and Javascript. As currently implemented, the system is a rather ugly prototype and primarily a testbed of testing the functionality of some of various elements we expect to have in the final system; it will also be evolving as we add additional elements. Nonetheless, the template-based system should now be useable for coding.

CIVET is written using the Flask "microframework"—`http://flask.pocoo.org/`—a Python-based system which is widely available on various cloud platforms. The complete CIVET code will be licensed as open source—we are currently using the MIT license—and provided on GitHub in the event you wish to create your own local or cloud versions. The prototype is currently deployed in the cloud on the "Google App Engine"; we currently expect the operational version will be deployed on a University of North Carolina server.

CIVET is currently implemented in two modes:

Coding form template: This is a fully functioning template-based coding form which implements several of the common HTML data entry formats.

Text-extraction demonstration: This is a demonstration of a coding form which displays a block of text and does simple automatic text highlighting and extraction.

We are very interested in feedback on this system, including any bugs you encounter (please let us know what operating system (e.g. Windows, OS-X) and browser (e.g. FireFox, Explorer, Chrome) you were using), aspects of the manual that are unclear (and features that appear too complex), and additional features that would be useful. Please send any suggestions to `schrodt735@gmail.com`.

Accessing CIVET

- Working version on Google App Engine: `http://ace-element-88313.appspot.com/`
- Source code: `https://github.com/philip-schrodt/CIVET`

[1] `http://en.wikipedia.org/wiki/Civet`

2 Template-Based Data Entry Form Operating Instructions

From the home page, click the link in the line *Read a coding template by clicking here*

1. Choose a template file then click the `Read file` button or just click the `Use demo template` button without selecting a file. You can also enter a coder ID if you want to use the `_coder_` variable. The `Download demo template` will download a copy of the template.
2. Enter data using the usual HTML data entry widgets
3. To save a set of coded fields, click one of the buttons which follow the title `Options after saving:`

   Code another case: Save, then return to the same form

   Download data: Save, then download data as a .csv file
4. The "Download CIVET data" page provides a text box for a file name, and the `Download file` button downloads the coded data. Use the *Start new data file* link to re-start the coding and the *Continue coding with this file* link to continue adding to the existing records.
   - The .csv file is tab-delimited and contains the variable names in the first line.
   - If the file name does not end in ".csv," this suffix will be added.
5. To quit the program, just close the window.[2]

3 Introduction to Templates

A CIVET template file specifies the individual components of the form: these are the familiar components from web forms but the syntax used to specify them is simpler than what you will find in HTML.

CIVET is simply adding these controls to an HTML <`form`> and, as with all things HTML, most of the placement of the fields is handled by the browser.[3] CIVET provides some limited formatting through the insertion of text and line breaks, and with some experimenting you should be able to keep the form from being too ugly.

[2]This, it turns out, is a HTML/Javascript security feature which prevents rogue websites from closing windows unless they have created the window.

[3]Writing in HTML5 and CSS, one can actually exercise a very fine degree of control over the placement, but if you are comfortable with that sort of code, you presumably aren't using CIVET in the first place. That said, you can see the HTML generated by CIVET by using the *View source* option in your browser, then save it as a file using *Save Page As...* and that could provide a starting point for creating prettier code.

The template file should be a simple text file—most systems are happier if this ends in the suffix `.txt`—similar to that used in an *R* or *Stata* script (that is, not a formatted file such as that produced by Word). Appendix 1 gives an example of a template file, and the code for this can also be downloaded from a link in the program.

At present the program does only a very limited amount of error checking; more of this will be added in the future. If the template does contain one or more errors, the system will display this on a web page.

3.1 Command formats

Commands generally have the following format

```
command: entry-title [var-name] options
comma-delimited list
```

Commands vary in how many of these components they have, but all follow this general pattern.

Each command ends with a blank line (or, if you prefer, the commands are separated by blank lines.)

Commands can also be cancelled by adding a "-" in front of the command: this will cancel the entire command, that is, all of the lines associated with the command, not just the first line. For visual symmetry, a "+" in front of the command "activates" it, though the command will also be active without the plus.

"#" denotes a comment: anything following a "#" is ignored, so lines beginning with "#" are completely ignored.

3.2 Items in template specification

The commands involve one or more of the following items:

entry-title : This is the title of data entry field. If this ends with `/` a line-break (<br>) is inserted after the text. The titles are escaped: at present the characters <, >and the single and double quotes are replaced with the equivalent HTML entities `<`, `>` `&quot;` and `&rsquo;`.[4] The **entry-title** field cannot contain the characters "[" or "]"—if these are present they will be interpreted as bounding the **var-name** field—but the escaped versions "\[" and "\]" are allowed.

[4] In the current implementation, named HTML entities such as `&copy;` and `&euro;` can be included and should produce the correct character. At present numbered entities such as `&#91;`—the HTML equivalent of ']'—do not work since the # is interpreted as a comment delimiter: depending on whether there is demand for this feature, the system could provide a way around this.

var-name : The text of the variable name for this field; this will be used in the first line of the .csv output file

comma-delimited-option-list : a list of the items that can be selected, separated by commas. A '*' at the beginning of the item means that it will be initially selected.

comma-delimited-var-name-list : a list of items which appear in **var-name** fields, separated by commas.

page-text : Any text

number : An integer

4 Templates: Specifying variables

4.1 Specifying variables to save

This command gives the variables that will be saved; these can be in any order but each of these must correspond to a `var-name` somewhere in the form, or are one of the special variables discussed below. A tab-delimited version of this list will be the first line of the output file. The command can occur anywhere in the file.

save: comma-delimited-var-name-list

Example:

```
save:  worldregion, eyewit, groupname, comments
```

4.2 constant

Sets the value of a variable to a constant; this can be used in a `save:`

constant: page-text [varname]

Example:

```
constant:  Data set 0.2 [data_id]
```

4.3 filename

Sets the default file name for the downloads: this can be changed before downloading

filename: page-text

Example:

```
filename:  our_wonderful_data.csv
```

4.4 Special variables

coder : Text entered in the *CIVET template selection* page

date : Current date. this is currently in the form DD-mmm-YYYY but later versions of the system will allow other formats

time : Current time in hh:mm:ss format

5 Templates: Data Entry Fields

5.1 Checkbox

A simple binary check-box. The value of the variable will be first item in the list when the box is not checked; the second item when the box is checked. The * notation on the second item can be used to specify whether or not the box is initially checked.

select: entry-title [var-name]
comma-delimited-option-list

Example:

```
select:  Eyewitness report?  [eyewit]
no,*yes
```

5.2 Select from pull-down menu

Pull-down menus—which are called a "select" in HTML—are specified with the syntax

select: entry-title [var-name]
comma-delimited-option-list

Example:

```
select:  Region [worldregion]
North America, South America, Europe, *Africa, Middle East, Asia
```

5.3 Radio buttons

A series of radio buttons are specified with the syntax

radio: entry-title [var-name]
comma-delimited-option-list

The entry `/` in the option list causes a line-break (<br>) to be inserted

Example:

```
radio:  Region/ [worldregion]
North America, South America, Europe, *Africa, /,Middle East, Asia
```

5.4 Enter single line of text

This creates a box for a single line of text (HTML `type="text"`). The `width = number` is optional and specifies the size of the text entry box in characters: the default is `width = 32`

textline: entry-title [var-name] width = number
initial-text

Example:

```
select:  Name of group [groupname]
<enter name>
```

5.5 Enter multiple lines of text

This corresponds to an HTML "TEXTAREA" object. The `rows = number cols = number` is optional and specifies the size of the text entry box in characters: the default is `rows = 4 cols = 80`

textarea: entry-title [var-name] rows = number cols = number
initial-text

Example:

```
textarea:  Comments [comments] rows = 2 cols = 64
-- put any additional comments here --
```

Templates: Additional Web Page Formatting

5.6 Set page title

Sets the title of the web page: that is, the HTML $< title > ... < /title >$ section of the header.

title: page-text

Example:

```
title:  CIVET-based coding form
```

5.7 Insert text

Adds text to the form: the various options follow the usual HTML formats. In interests of simplicity, text is "escaped" so that special characters are not interpreted as HTML: note that this means that in-line mark-up such as $< i >, < b >$ and $< tt >$ will not work, so if you need this activate and use the **html:** command. Also keep in mind that these commands need to be separated by a blank line.

h1: page-text
h2: page-text
h3: page-text
h4: page-text
p: page-text

Example:

```
h1: Primary data set coding form

p:Please enter data in the fields below, and be really, really careful!
```

The simple command

```
p:
```

is useful for putting some space between form elements.

5.8 Insert HTML

[This command may or may not be included in the operational version of the system, as it provides some opportunities for mischief. Stay tuned. It is in the code but currently

deactivated; if you are installing your own version of the system, it can be activated by changing a single character in the source code.]

Adds arbitrary HTML code without escaping.

html: page-text

5.9 Insert a line break

Adds a new line in the form

newline:

6 Text-Extraction Demonstration Form

The second major component of CIVET will be a system for extracting information from texts: this will include mark-up, automated highlighting of certain types of information (e.g. names, dates, numbers, user-specified sets of words and phrases), and a method for rapidly identifying these and putting them into a data entry field. In the current implementation, we only have a proof-of-concept prototype of this system: this is not ready for use in actual coding.

To activate the demo, from the home page, click the link in the line *See a demo of the text-highlighting system by clicking here*

1. Select a text file to edit: you can use either the pull-down menu or radio boxes, then click the `Edit the file button`.
2. Click one of the text entry boxes will highlight the relevant words in text: For demonstration purposes these are words beginning with the letters 'a', 'c', 'd', 'e' and 's'. The 'tab' key cycles between these options, or an option can be selected using the mouse.
3. When a text entry box is active, the first relevant word in the text is highlighted. The right-arrow key will cycle the highlighted word. To copy a highlighted word into the text box, use the down-arrow key.
4. Text can also be selected using the mouse: To copy the selected text into the text box, use the left-arrow key.
5. Text can also be entered manually.
6. To save a set of coded fields, click one of the buttons along the bottom. At present, all three buttons save; we will be adding "cancel" and "reset" options. The options are:

   Return to this case: Save, then return to the same text

   Select new case: Save, then return to the same text

   Download data: Save, then download data as a .csv filr
7. The "CIVET Download" page provides a text box for a file name, and the `Download file` button downloads the coded data. Use the *Start new data file* link to re-start the coding and the *Continue coding with this file* link to continue adding to the existing records.
   - The .csv file contains the variable names in the first line.
   - If the file name does not end in ".csv", this will be added.
8. To quit the program, just close the window.[5]

[5]This, it turns out, is a HTML/Javascript security feature which prevents rogue websites from closing windows unless they have created the window.

7 Projected Features

- We are currently developing this based on a fixed frame that is 960 pixels wide and 700 pixels high: this will fit easily on a 1024x768 screen. For contemporary equipment this is probably quite conservative but it is not unheard of for data labs to have older equipment so we're going with this at the moment

Appendix: Sample Template File

```
# CIVET template demonstration file

h1:Ministry of Magic Hogwarts Incident Report

radio: House where incident occurred: [house]
Gryffindor, Hufflepuff, Ravenclaw, *Slytherin

p:

select:Nature of incident [natincid]
*Minor mischief, Unauthorized absence, Accident, Major infraction, Unforgivable Curses, Other

p:If "Other", provide details in the report section

checkbox: Was incident reported to school authorities? [authreport]
No,*Yes

checkbox: Did incident involve muggles? [muggles]
No,Yes

p:

textline: Name of student(s) [names] width=80
Enter names here

p:

textarea:Brief description of incident [descrp] cols = 80
Enter brief description here

p:

textline:Reporting official [reporter] width=40
Enter your name here

h3:Thank you for your assistance; we will contact you by owl should we require more information

save: _date_, house, natincid, authreport, muggles, names, descrp, reporter
```