# TA2 Read Me

TAMS Analyzer 2.0 moves everything to a project focus. Note that this is still under development and soon will feature more options.

Converting from Version 1 to Version 2:

1. Run TA2.
2. You will see a work bench like window. This represents a project.
3. From a project window pick from the Project menu "Convert ver. 1 prefs to project files"
4. TA will save the project files on your desktop.
5. Double click a project file.
6. Select the code file from the list of files listed
7. Pick "Import codes and defs from code file" from the Project menu
8. Save the project. From now on use the project files to access TA.

TAMS Analyzer 2.0 Change list

1. the whole program is shifted to a "Project" focus. There can be multiple open projects now. Files do not actually have to be open to do searches. Codes and definitions are now stored in the project file that is a regular file rather than a plist file buried in a system folder. There are tools on a new project file for importing the old code files and exporting the project codes and definitions for external storage.
2. The regular expression engine was upgraded to AGRegex .3 and PCRE 4.0 which while slower plugs some bugs. NOTE REGEX IS VERY, VERY SLOW AND MEMORY INTENSE. BEST FOR SMALL DOCUMENTS AND SMALLER NUMBERS OF REPLACES
3. Escape characters \t, \r, \n can now be used in Non regular expression searches/replaces. I offer this as small compensation for the slowness of REGEX stuff.
4. To manage codes in the project file there is now a code browser window.
5. Code definitions can now be accessed from the code lists on search and workbench list views as well as the usual editor. These are now evoked through a "Code Definition" menu option from the "Coding" menu.
6. Repeat codes attached to a data code that crosses sections/ends can now be determined at the !end location of the open tag or the close tag of the coded passage. This is determined by a user preference or metatags !frontloadrepeat and !backloadrepeat. Consider this example:

```
{!repeat a, b}

{code}
{a}r code 1{/a} some text {!endsection}
{b}r code 2{/b} other text {/code} {!end}
```

If you front load the repeat, code “a” will have a value but not “b,” since the values will be ascertained at the !endsection; if you back load the setting of the repeats, {code} will have values set for “a” and “b.”

7. Cleaned up the save file format types a little, but still problematic
8. Improved international character set handling
9. Checking for legal characters for new codes in both the browser and the document.
10. “unlimited” (no code defined) non-simple searches are now possible
11. !zapuniversal and !nozapunversal metatags created for emptying or not emptying the universals at the end of the file; like {!dirty} and {!clean} with repeat values.
12. The files in a project can be saved using absolute paths (paths that are not calculated based on the location of the project file) and relative paths (file locations determined relative to project files). These are set by a “Project preference” panel selected from the Project menu.

    Use absolute paths when you want to move the project file relative to the source files.  Use relative paths (uncheck the absolute path setting in the project –not application—preference panel) when you want to move a whole directory tree including source files and the project file to a new location or to a different machine.

# TA 2.1 Release Notes

Prior to 2.1 searches could be exported to Excel or database software, but there was no native format which allowed the documents to be saved, reopened, and refined. The big new feature in TA 2.1 is the ability to save results of searches in a native format which preserves named selections and links to the source documents. To facilitate this, workbench windows now have a radio button to toggle between source files and the results you've created. These result files mean nothing by themselves, they need to be saved as part of a project file, i.e., a set of sources. Furthermore they are very delicate, in the sense that if you change the name of any of your source files the results will no longer be openable (actually you can change the name of a source file through save as, but only while the result file is open (i.e., visible), if the result file is closed it won't have access to the new name and will not open properly. Furthermore, you will then have to save (or resave) your result file for the change to take effect. Another side effect of all of this is that every source file needs a unique name under this new regime. At present TA does no checking regarding this.

In earlier versions of TA, the title of the result window was either result, the name of the limit string or a string you, the user, provided. Now windows are documents that are given titles by saving them. The window name field has, therefore, been removed. Also there is now a field that provides a history of your analysis of your results so that you can have some sense of what you are looking at, at any given time.

Saved search results are recalled by changing the mode of the file list to "Reports" from "Sources". Now double click the name of a saved search result. (or click one time and click the open button).

Users will notice some other changes to the project/workbench windows: The bottom contains status information: An "a" or an "r" is used to indicate whether the project is saving file locations as relative or absolute paths, and a count of files (either results or sources depending on which is showing in the file list) is also provided.

Some new options are available as well in the preference panel that should make users happy. First, users can now create universal button bars with the {!button} metatag.  Put these in the init file and make sure that "Scan init file when opening a data file is checked." The first time you may have to reload the button bar in open windows  by picking  "Rebuild button bar" from the coding menu.  Init files can also now be searched as part of inidividual file searches by means of a small check box in the searching tab view.

Finally, non-simple searches can either include or not include repeat variables in their results. By this I mean, Repeat variables will be treated as data in a non-simple search if the "Include repeat variables in non-simple search" preference item is checked. For instance:

```
    {!repeat a}
```

```
{a}this{/a}
{b}something{/b}
{!end}
```

Will return two records if it is turned on (one for “this” and one for “something”, but only one if it is turned off. Note that for both searches a column indicating the value of a will be provided.

Future developments: If you want to see a hint of the next phase of TA development and you have Interface Builder, you might want to look at myResults.nib.

Changes

1. fix to bug in NSString for codes that are one letter long

2. improved unlimited-non-simple searches so that repeats are returned properly

3. save results to file

4. results are now part of the projects and can be saved and reloaded

5. project/workbench windows now have a toggle to view results as well as sources

6. project/workbench windows now indicate the path saving mode (absolute or relative) and number of files in the file list pane

7. individual source search windows now have a switch which determines whether the init file is included in search

8. the program has a preference as to whether init files should be searched when looking for !buttons

9. The actions of the user in results (selections, etc.) are shown above the browser window

10. a preference is provided to indicate whether repeat values should be returned as results in non-simple searches

11. The window name field has been removed

The only change between the a1 and a2 versions of TA2.1 is that a serious bug fix has been done for save and save as results.

TAMS Analyzer  2.11
A niceties and bug fix release

o This release fixes a major bug that would not allow you to save your projects after a results file was closed.

o On the preference panel there is now a choice as to the character encoding  format in which you want to save your result files when you use “Save To”: unichar, UTF-8 or 7bit ASCII, Windows, NextSTEP and MacOS encodings.  ASCII  MacOS will allow you to use Excel and Filemaker for results but clobber  character palette symbols . Unichar preserves these but I’ve not found it compatible with Excel or other programs. Damned if you do or damned if you don’t: you decide.

o I’ve added a little definition button to both the workbench and individual file search windows so you can easily pull up code definitions in both places.

o Generate code list is now working again, and now produces the list in a new file rather than at the bottom of the open file.

## TA 2.12 release notes

This is mostly a fix it up release with a few odds and ends features added.

**Changes:**

- Fixed a major bug with recode when a shorter code was inserted.

- Fixed a bug with !name which would produce two FileName columns if !name was used in both the init file and the first file in the search list

- Coder string (this is the field where you limit search results to just certain coders) is now saved in results so reloading and refreshing restores the same subset.

- Fixed but with project window not checking state of files when the close button was clicked.

- Adding files to the search list now moves through the file list and checks so that you don't add a file twice

- Syntax checking routines on the coding menu ("Check for nested" and "Check for pairs") now can be executed from a project window, in which case it will check across all source files in the project.

- There is now a "Select exact string" in the result windows that will only return records that match this whole string exactly (not a substring).

- As a select, recode, add code and set operation sheets can now either be attached to a window or float above them through a preference option.

**Some notes about result files:**

Result files save more than just the data, they save the search string, with this release, the coder, the flags and the search list that was used to construct them. This has important consequences. Changing the search list and refreshing results will not produce any change in search results. Also, search files always assume that there are changes in source files and indicates this with a check by the refesh button. That's because there is no way of telling, if they were closed, whether or not you have made changes in the source files that feed it. (If I were a really sophisticated programmer I'd go get that information, but I'm not…). Finally, because of all of this you may want to refresh your results, but that will mean you'll lose named sets since the records that the named set is composed of will be erased and recreated. You decide whether the named sets or the refesh are more important. If you're still add code'ing and recoding you'll want to refresh, however.

TAMS Analyzer 2.13a2 Release Notes

It's been a good week: no new bugs (which isn't to say old bugs have all been dealt with). Three new features:

1. Named search lists: Now you can save the state of your search list (including init file) and instantly restore them from a little pop up menu. In most of my projects I have a single search list, but some people I know tweedle their search lists a lot.

2. {!block X} metatag.  Basically this tells the processor to ignore the named codes. Why would you do this. Well,  at different times I want different repeat values to be displayed. In my init file I can indicate which ones I don't want to show using something like the following:

{!repeat a, b, c, d}{!block c, d}

Codes c and d will now be entirely ignored, no column will be produced for them and no records created for them.

Note this is not really a new feature, but one restored from command line TAMS.

3. {!inner X} This is a real charmer. When you have a situation in which only one repeat value is changing over the course of a document, you can assign this value as an "inner repeat" and then each  incident of X is treated as though it has an {!endsection} in front of it. Note only one value can be an !inner. The following will not produce anything meaningful: {!inner a, b, c}. Also you will have really unexpected and unhelpful results if you have more than one repeat variable changing value over the course of the section (a section in this case being defined as a portion of a document ending with an {!end} metatag).

Example

The following

```
{!repeat location, date, time, name}
{!inner name}

{location}New York{/location}
{date}10/22/2044{/date}
{time}3:00pm{/time}

{name}Bob{/name}: {veggie}I love talking veggies{/veggie}

{name}Sam{/name}: You would you cretin, {meat}meat is the only proper
food{/meat} for hominids
```

```
{name}Mary{/name}: Sam get a life, {fruit}I've been a fruitarian for 30
years{/fruit}

{name}Ellen{/name}: Not me; I'm basically a {poultry}chicko-fisho
vegetarian{/poultry}

{!end}
```

is the same as the following:

```
{!repeat location, date, time, name}

{location}New York{/location}
{date}10/22/2044{/date}
{time}3:00pm{/time}

{name}Bob{/name}: {veggie}I love talking veggies{/veggie}

{!endsection}{name}Sam{/name}: You would you cretin, {meat}meat is the
only proper food{/meat} for hominids

{!endsection}{name}Mary{/name}: Sam get a life, {fruit}I've been a fruitarian
for 30 years{/fruit}

{!endsection}{name}Ellen{/name}: Not me; I'm basically a {poultry}chicko-
fisho vegetarian{/poultry}

{!end}
```

Notes:

1. Note that the first time the variable is set there is no endsection.
2. If there was more than one repeat variable changing this would not have worked. If you need to change more than one variable over the course of the section (ending with !end) you will need to insert the {!end}s or {!endsection}s yourself.
3. Note the location where the {!endsections} are virtually inserted. Make sure that that is the location you want the data dumped. If not, don't use !inner; just insert the !endsection 's yourself
4. You can change the "inner repeat" variable. Just use the {!inner} at the top of the section (before first use of the variable) you want the new value set to. You can also void out the "inner repeat" with a {!inner} call.

For many types of data this can make life simple, but only in the most basic cases. Even with interviews you may want to vary more than one value (name and question, for instance) in which case you will have to do the more arduous coding described in 2.

4. This feature has actually been in TAMS Analyzer for a few releases but it has been undocumented. You can set multiple conditional values with one !if statement:

```
{!if name=”bob”=>home=”NYC”=>gender=”male” => age=”25”}
```

This is the same as

```
{!if name=”bob=>home=”NYC”}
{!if name=”bob”=>gender=”male”}
{!if name= “bob” => age = “25”}
```

TAMS 2.14a1

Features:

1. added code browser button on the workbench (labeled c.b.—just pull up the code browser)
2. \n's and \t's and \r's are now interpreted before being displayed in the results window
3. improved coding: text does not change color when a code is applied—note the code does not necessarily come out the color you want; you still have to color the codes separately.

Bug fixes:

1. Save document bug fix that had disastrous effects on the state of closed documents
2. Saving results bug fix: bug introduced last release which did not load the saved results.

## TAMS Analyzer 2.15 Release Notes

1. Live coloring of tags: use the preference panel to indicate what color you'd like your tags to be! Be careful, however as you may find yourself typing in the new color if you try typing after one of these tags. As a work around to this use "Type to Black" on the Font menu or copy and paste font from the Font menu to apply the formats you actually want to be typing with.

2. "Type to Black" (just turns the current font color black) added to Font Menu to help with coding in color right before the insertion point.

3. Post-search tag hiding. Use the results menu to toggle whether your tags are visible. Only works for raw searches (obviously, I hope).

4. Now you can browse through your codes in the Code Browser using the up and down arrow key.

5. Bug fix kludge: somewhere in the bowels of my code is a bug which is clobbering the willCloseWindow: method of results windows. This has caused problems for various folks out there. I've got a kludge fix which gets the software to run seemingly unproblematically, but does not stamp out the problem at its source (unknown), which means there's no reason to think it wont rear it's ugly head in other guises. It's a tricky one.

6. For those willing to hack: I've included gmodel files of the nibs so that one might port this to gnuStep.

TAMSAnalzyer 2.16a4 Release Notes

TA2.16 adds two important features to TA:

1. An additional project preference option to help when projects are transferred to different machines. In addition to Relative and Absolute there is "Same Folder." This is best used if you move over a project in absolute mode. Simply put all the files in the same folder and then switch the project (in the project preferences) to "Same Folder" and save the project. You may have to close and reopen the project, but you should be able to access all of your files at that point.

2. Section searches. Section searches are triggered by using either the (new) section switch on the search panes or by using the {!sectionsearch} metatag as your very first tag in the document. In section search mode the program returns whole sections (demarcated by {!end} tags—or {!endsection} tags if "Report empty searches for each {!endsection}" is selected in the program preference panel. If you use a "limit" when searching for sections AND ("+") and OR (",") function differently than in other modes: they refer to the section rather than sections of code. so searching for ALPHA+BETA will return the full text of any section that contains both codes whether or not they overlap!

In addition there are bug fixes, some maybe due to changes since the last release however. Key bug fixes include the ability to close the preference panel using apple-W or the close menu.

Also the source code has been made much more GNUStep friendly for anyone wanting to do the tedious work of porting: all the references to cocoa.h have been removed, a #define COCOAVERSION has been added to isolate the cocoa specific sections, newlines have been added to the end of all header files. The name of the GNUStep project is GTAMSAnalyzer2, and a make file as well as up to date gmodel files are provided. At present, at least on my machine the program compiles and then segfaults under GNUStep.

TAMS Analyzer 2.17a2 Release Notes

New Features:

- Interface redesign so that the searches on the workbench windows are picked from a pull down menu

- Character find: this is a way of searching for a string across all the documents in your search list. Has a Regular Expression search as well: **warning—Regular expression searches are VERY slow; I mean hours slow; I mean you'll think your machine has crashed slow, so you may not want to do a regex search if you don't have to**. At some point I'll try rewriting the program with a different regex search engine, but that will not be easy. AGRegex is pretty integrated into the system at this point.

- Bug fixes for some elements of section search. Init files are now ignored in section searches and no warning is given.

Notes:

Let's say you want to look for the string "artichoke" in all your documents. Obviously you could open each and do a "Find" from the Find menu. Alternatively you could put the word "artichoke" into the workbench window's "Search" field and do a batch search of all the files in the search list!!! Use the Exact check box to indicate whether you want the search to be case insensitive (unchecked) or sensitive (checked). You will get a results window with a row for each instance of your search string. You can use the "Find record" button to move back to the original. There is also a "zoom factor" which controls how much context you want to see. Use the preference panel to set the zoom factor which is the number of characters on each side of the found string you want to see.

Two features appeared in the last release without comment: the collapse up and collapse down menu choices on the results menu. This is borrowed from dataBoiler. Often I want a single record for each value of a certain repeat value. For instance, If I look up a code (say "fruits>apple") in a bunch of interviews and I would like a list of everyone who mentioned apples I have a dilemma. Interviewee Bob may have 18 different records with apple mentioned. I only want to see bob once. That's where collapse comes in. Sort on the repeat value (in this example "interviewee") and then pick collapse up. Only one bob record will be there and only one for each of the other interviewees. Voila, I can now see a simple list of who mentioned apples. Collapse up takes the first record as representative; collapse down takes the last record as representative.

TAMS Analyzer 2.17a4 Release Notes

o Upgraded to PCRE library 4.3. Regex's are still a misery in terms of time, memory and diskspace however. Alternative regex libraries anyone?
o Squashed section search bug (again) so that init files don't count as a section (they were counting as a section, which is usually and now always –by definition—wrong).
o Squashed a bug which wouldn't allow projects to open if there were empty strings as file names.

That's all

TAMS Analyzer 2.17a7 A cleaning up some details release

New features:

- Better saving of projects. Projects can be saved with untitled result windows still open; it's just that those result files will not be included in the project. Once saved those result files will be added to the project and the project will be marked dirty. Result windows still need to be closed before you close a project window.

- Naming or deleting search lists will now mark the project window as dirty

- The preference panel is now tabbed so that the options don't appear in one long list but are separated into those preferences dealing with coding and those dealing with searching.

TAMS Analyzer 2.18a3 Read Me

New Features

- Delete tags from your results: you can now delete tags using the basic mark and recode system. Actually you mark and then pick delete codes in source. This only works for simple searches.  It deletes the code indicated in the _code column; not codes embedded in the actual passage you're examining

- The recode menu has been cleaned up a bit; well spaces have been added at any rate

Bug Fixes
- Since adding _doc to the result window users have not been able to export results to text or XML. Both export features have been restored.

TAMS Analyzer 2.20a1

The .2 designation is pretty arbitrary version change.

New features:

1. The key point of celebration is the new result window now is divided by a split pane that allows you to adjust the browser to rows of results ratio.
2. You can now "escape" brace characters so that they are NOT interpreted as tags. This way you can use { and } without them being interpreted. Escape is a technical Unix term for putting a "\" before a character. It changes the meaning of the following character. Just put a "\" before each so, to misquote Magritte, \{this\} is not a tag. It's just a bunch of characters. If you need to actually have a "\" in your document you need to escape it, i.e., \\ is turned into \ when you do a search.
    a. To facilitate this there is a menu option on the Coding menu which allows you to select a passage and it will automatically insert "\"'s before each { and }.
    b. There is a program preference as to whether you want to use this feature. (you may have a lot of "\"'s around which you do not want interpreted as escaping the next character, so you can turn this all off.
    c. There are two metatags to selectively turn on and off interpretation of "\" throughout the document. {!escapeon} and {!escapeoff}. Note, this only effects searches, not other activities that read through your file (e.g., checking for code pairs).
3. Bug fix to the parser that prevented opening of some rtf files at least
4. Modified apply code algorithm so that undo works closer to "as expected"

TAMS Analyzer 2.21a4 Read Me

TA2.21 adds the ability to set colors for individual tags and tag families. The default choice is “Inherited Color.” The idea is that tags normally inherit their color from their “parents” so aa>bb will inherit from aa which inherits the “default” color set in the preferences. You have option of changing any particular tag’s color, but by default it gets what it’s parent has, and it’s children (e.g., aa>bb>cc) will inherit the color you’ve assigned it. Any time a user can define a code, they can set its color: in the code browser, in the addcode/recode sheet (results windows), and in the new code sheet (document windows). The program will automatically recolor your documents if you have the preference set for automatic colors in the program’s “preference panel.” If you turn off automatic coloring you can “manually” color by choosing Coding->Color Tags menu option. There is a time cost to the automatic coloring. For some tasks (recoding, for instance), it might be substantial. Hence, automatic coloring is offered as a preference panel “option.”

Also in this release: previous and next code menu options now skip “escaped tags” (see 2.20 release notes)

Slightly different behavior with results windows. Creating results doesn’t automatically set the project’s dirty flag; only when files are saved (i.e., are attatched to a file) that they effect the project’s dirty status.

TAMS Analyzer 2.22 Release Notes

This release includes 3 MAJOR additions to TA's capacities.

1. TA now can create and work with RTFD files; these are actually folders that the finder and other programs treat as files. Basically this allows rtf (i.e., stylized text)+ graphics. TA does not actually allow you to code graphics; at least not in a meaningful way; but at least you can include graphics in your files.

2. TA now allows users to calculate inter-rater reliability (IRR). This includes Cohen's Kappa and a rough decimal/% of the number agreed upon. See the "IRR How-To included in this release's documentation.

3. The left and right halves of the document window now have a split panel divider so that you can adjust the width of the code list.

In addition to these additions to TAMS Analyzer itself I am including in this release a series of open source, auxillary unix, command line applications w/source that can help you line number your data sources. There is a brief tutorial on how to use these to create a line numbered source file you could use with TA.

## TAMS Analyzer 2.23b2 release notes

The major addition that this release of TA has is the ability to wrap and line number your document. It's all very kludgy. Line numbering is popular in other qualitative software packages. In 2.22 I just bundled a bunch of gpl'd software that could do this wrapping/line numbering for the user; here I've integrated the feature (see coding menu).

Nevertheless line numbering is very "unTAMS" which prefers to identify seemingly natural units of data (turns in conversation) and use repeat values to identify those chunks. Line numbers are arbitrary in comparison. All that is supported is transforming your document into one with line numbers and a certain fixed line length. By the way escaped characters and tags don't count in the line length (this is a bit of a problem since escaped characters should count, well I'll get around to fixing it sooner or later). TAMS searches don't do anything with the line numbers: if there is a line number included in the passage that is coded, you'll see it in the results window, otherwise, if you want to see the line number you'll have to use "find record".

New feature:

> Line number and line folding

Bug fix

> Empty records of Empty searches now handled better: find record will place the user after the end of the "empty" record. Beware, there are some strange end effects in Cocoa's textview which I haven't been able to code around.

TAMS Analyzer 2.23b3 Read Me

This small upgrade increases the usability of line numbers. A switch under "Searches" in the preference panel turns on scanning for line numbers. This is necessary so that researchers who do not need line numbers do not pay (much of) a speed penalty. Here's what "Scan for line numbers" on the preference panel does.

If your text is as follows:

```
1       But {hero}d'Artagnan{/hero} had on the preceding day served his
2        apprenticeship.  {warfare}Fresh sharpened by his victory,
3       full of hopes of  future favor, he was resolved
4       not to recoil a step. {/warfare}
```

The search will return two records. if you have turned off "Scan for line numbers" The first record returned will be

```
{hero}d'Artagnan{/hero}
```

Notice the lack of line numbers. This is made even more apparent in the second record returned:

```
{warfare}Fresh sharpened by his victory,
3       full of hopes of  future favor, he was resolved
4       not to recoil a step. {/warfare}
```

Notice that the first line lacks a line number since the line number was not in the coded passage. Turning on "Scan for line numbers" returns the following instead:

```
1       {hero}d'Artagnan{/hero}
```

and

```
2       {warfare}Fresh sharpened by his victory,
3       full of hopes of  future favor, he was resolved
4       not to recoil a step. {/warfare}
```

In addition a new _line_number column is provided so that researchers can use that information. It indicates the line number of the start of the passage

Note that if you do not use line numbers turn off "Scan for line numbers" to reduce the overhead of the process.

Final note: If you want TA to number your paragraphs without wrapping the lines just use a 0 line length.

Changes in this release

1. User preference to scan for number lines
2. If "Scan for line numbers" is enabled TA adds a line number column if it finds line numbers. It also prefixes any found line number to the display and saved text and XML file.
3. Save and save as of results triggers the dirty flag of the work bench (not ideal, but better than the work bench not including the results in the project).

TA 2.3 Release Notes

TAMS Analyzer 2.3 adds a number of fairly esoteric, but important features to the program.

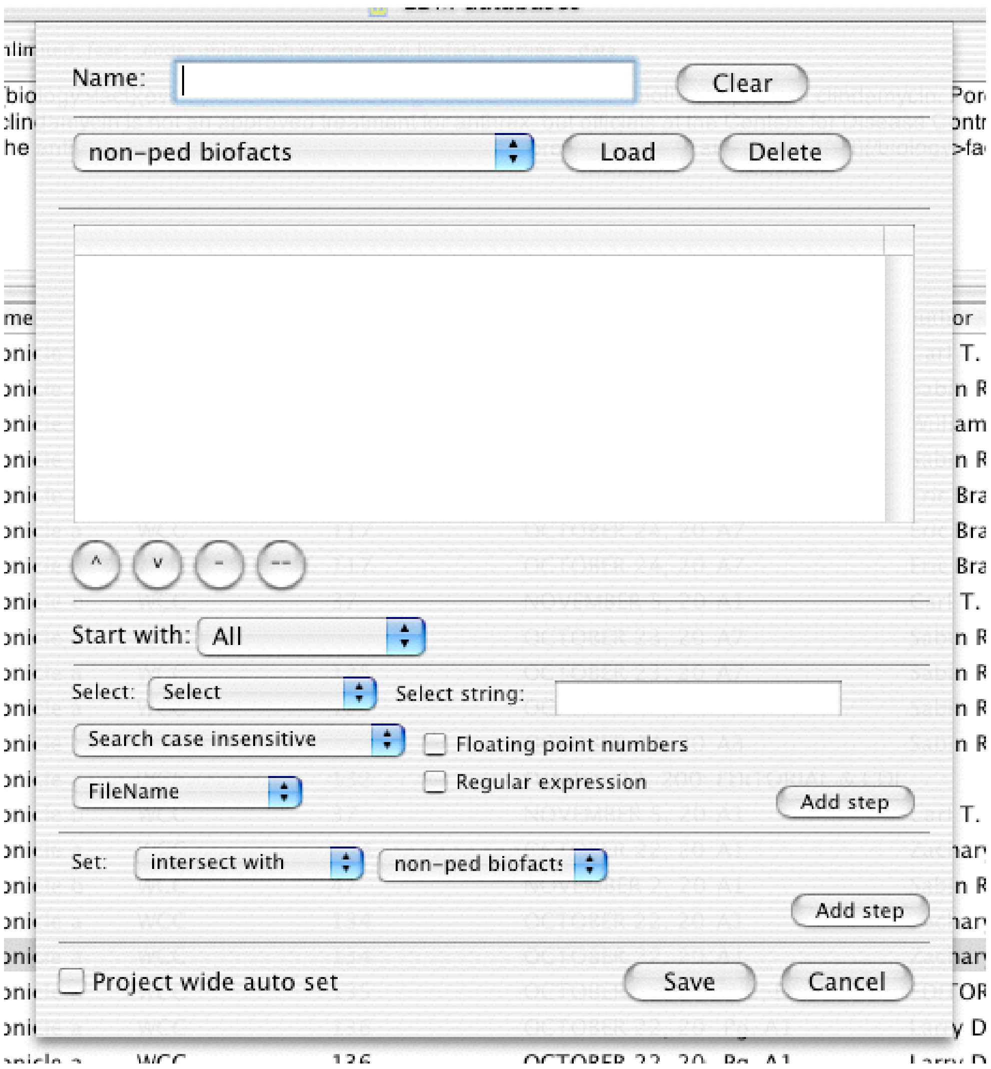
1. You can name and recall the searches you conduct on the work bench. Giving a search a name saves the type of search, the search string, the coder string and the flags (raw, exact, empty) of the search. This feature also introduces a simple way of interface for the key buttons used to manage Named searches (and named search lists as well):

    **+** which adds the current conditions, prompting for a name
    **-** which deletes the named search currently showing in the pop-up menu
    **--** which deletes all of the named searches (or search lists).

These symbols are used in both the named search, named search list and on the new auto set editor.

2. You can now create "auto sets" which function like macros, and which, in many ways they act like named sets (formerly called named selections), but instead of saving the selected passages, the program saves the rules needed to pick out a selection of records. There are two advantages and one disadvantage to auto sets over named sets. Advantages: auto sets are persistent (they aren't destroyed by a refresh) and they can be project wide. The disadvantage is that they are not as nuanced as named sets. You can remove individual records when creating named sets. Named sets do not need to follow a particular set of rules, auto sets do. To create an auto set you pick Results->Sets->Create auto set…

That displays the following editor:



Name:

non-ped biofacts

Start with: All

Select: Select  Select string:

Search case insensitive  Floating point numbers

FileName  Regular expression

Set: intersect with  non-ped biofacts

Project wide auto set

To make an auto set fill in a name (it must be unique, i.e., not a named set or auto set name; also the words All and Visible are invalid, i.e., reserved, names) and select what the program should start with: **all** of the records returned by a search, the records **visible** at the time the auto set is run (so an auto set can be a sort of flexible filter on records showing), or an already existing auto set.

Then you need to add the steps that will comprise the auto set: What does the program need to do to winnow all the records returned by a search (or showing, i.e., **visible**) to those you are interested in. There are two major types of steps: selections and set operations. Selections are of all the types already available on the results menu and are documented elsewhere. You can also use regular expression strings (regex) and determine whether the search is case sensitive, numeric, etc.

The other type of operation is a set operation that is described in earlier release documentation.

The results of one step feed into the next and are ultimately displayed.

You can delete steps and shuffle them around using the ^, v, - and – buttons. You cannot edit steps, but you can add one that you want and then use the ^ and v buttons to move the new step into place.

Finally, by clicking on the “Project wide auto set” box you can save the auto set in the project where it will be available for all result files in this project.

3. Remove from selection

This is a selection search in result windows that will kick off of the display any records meeting the criteria you specify. Previously you would have to use set operations to achieve this effect.

4. Inter rater reliability now has a tab that allows you to compare how the raters scored the different test items.

Bug fixes:

1. When remove all is issued from work bench, result windows now are properly unplugged so that they do not crash the program when closed.

2. _doc column of the results file is now searchable like the rest of results windows

3. Improved “Same folder” mode for projects

4. Shuffle up and down of results fixed

TAMS Analyzer 2.31b10 Release Notes

This version substantially improves interaction between source and result windows. Result windows are indexes to coded passages in source windows/files. Changing the source knocks the result index out of synch. TA tries to be smart about some things. In the past Recode, Add code and delete codes in source (all on your Results->Recode menu) have done the math to readjust the indexes in the results window that initiated the recode/add code/ delete code, but <u>only</u> that one result file.

This version should maintain a good index with six different operations:

From your results window:

**add code**
**recode**
**delete code in source**

(of course these were well behaved before, only now, recoding from one result window will correct the indexes in all open result windows)

From your source window:

**double click coding**
**code button coding <u>(which now has an optional comment sheet that can pop up)</u>**
**delete code pair**

This should mean that you can go a lot further without hitting the "refresh button". Furthermore, indices will be updated for all open result windows, not just the one initiating the changes.

NEVERTHELESS the tie between the two remains tentative. Any added codes will not be entered into the results, though the existing index in results will be adjusted so that records that exist are properly indexed. Similarly with deleted code pairs: the record will remain even though there is no there *there*. Of course, any direct interaction including typing, cutting and pasting will put your results file out of synch. Furthermore, if you double click code or anything else and then "Undo" the results file will not undo the index: you will be out of synch and need to use the refresh button to bring the files back into alignment. **To facilitate the user, you can lock out most of those functions by selecting Reanalysis Mode from the Project menu.** This turns off editing, undo, and other functions that are likely to throw the results file out of whack.

If you stick to the six operations you can do a lot of recoding in any order you like (some with marked records, some with using the "find record button" and selecting and double clicking, some by delete code pair and then double-clicking a new code in place) and not have to refresh.

Personally I would check the synchronization by using “find record” often and see if the program takes you to the right location.

Finally, add code and recode now have the option (see user preferences) of unmarking all the records that have been changed.

New features:

1. There is now a user preference for a sheet that asks for the comment part of a tag when the user clicks a code and then the code button.
2. The 6 operations listed above now update the indices of all open result windows.
3. Reanalysis Mode locks out options other than the 6 listed above in your document window.
4. Add code and recode now have a user preference to determine whether or not they should unmark the marked records when completed.

Bug fix

1. Turning off automatic color coding from the user preferences now works
2. Dirty flag properly set for reloaded result files when source files change
3. Improved handling of empty auto sets and text display in results

TAMS Analyzer 2.31b12 Read me

This release fixes a bug that crippled the hide code feature under the Results menu. The bug was introduced with the “Escape braces” feature a few releases back. Also, the Regex engine is updated to AGRegex .4, which provides faster response time for long documents. It is still a slow poke, and if you can avoid using it, I’d recommend it.

Changes

AGRegex upgraded to version .4; PCRE to 4.3

Bug fixes

Results->Hide tags feature resotred to working order

For previous feature additions and bug fixes see the TA2 Release Update Notices file in the Documents folder.

TA 2.32b1 Read Me

This version of TA improves recoding and add coding substantially. It fine tunes several other features. It also has one significant bug fix

1. Warning message if the program has trouble finding a close brace in a search
2. Recode and add code now have keyboard short cuts
3. Recode and "Add code" now changes the value of the _code and _data field of the recoded record and all other records affected by the change. WARNING: Add code does not add the newly created record. Use the refresh button if you need the new record in your search results.
4. Both reanalysis menu check and named set/auto set list are updated when new windows come to the front.
5. Bug fix: program would crash if a document window was closed and the user immediately switched the program OUT OF reanalysis mode.

I have also included a brief description of how I carried out a fairly complex analysis.

TAMS Analyzer 2.33b2 Release Notes

This release makes one substantial change in the operating of recode and addcode updating. It also introduces a way of creating subsets of the codes in the project so that researchers do not have to always confront the totality of their code system when a small subset will do.

1. Recode and addcode updating.

   After the last release, and after experimentation, I found that the algorithm used to update records was very much hit or miss. In fact figuring out what needs to be updated in the results window _data column involves a lot of guessing since _data contains more than just the text string marked by _begin_loc and _end_loc. I've improved the algorithm but it's still a guess and will be wrong and miss some tags. Because of this uncertainty, I've made updating results windows a user option on the preference panel. If the option is not selected (the default) only the _data from the records that are being addcoded or recoded will be affected, and only from the window initiating the change. In fact the only way to really assure that your results window reflects your sources is to refresh the results window (via the refresh button). I find that it is handy to have the program guess and try to update the _data column, even if it is wrong on occasion, but the user should beware and if need be can turn off this feature from the program preferences, if precision is needed when working through a results column. Note that if the user does want the program to update the results window, <u>this</u> version will attempt to do so for all open results windows connected with a project and not just the window initiating the change (i.e., the addcode or recode).

2. Code sets

   On large projects it is helpful often to only be dealing with one portion of your code dictionary. For instance, while doing reanalaysis I often only am concerned with one code branch (in the anthrax project, for instance, the code family around fear). Having to scroll through 150 codes to find the small number I'm interested in becomes tedious. To facilitate reanalysis, TAMS Analyzer now allows the user to create a "code set," i.e., a subset of all the codes in the project. Once a code set is selected, only these codes will be visible in the document or project code list.

   To create a code set, you need to have the <u>project window</u> in the front and select Project->Code set->Manage code sets. This will produce a sheet/dialogue window which allows the user to create and delete (and delete all in one click of the button) code sets.

   To create a code set, (1) select the codes you want to include (by clicking, shift-clicking, and apple-clicking the codes listed), (2) fill in a name at the top, and then (3) pressing the + button.

To delete a code set, (1) pick Project->Code set->Manage code sets and (2) pick the name of the code set from the pop up menu. (3) Then press the – button.

Press the “- -“ button to delete all code sets.

To modify an existing code set  (1) select it from the menu, (2) adjust the selection, and (3) press + again.

Once you have created your code set you should click the close button and save your project.

To actually use a code set simply pick the code set name from the Project->Code set menu. To see all codes pick Project->Code sets->View all codes. You can pick code sets (or view all codes) from either the project window or the document window (not Results windows),

Note that any new codes created will be added to the current code set as well as the general dictionary. Code sets are stored in the project file.

TA 2.34b5 release notes

This version has a bundle of new features.

New Features and Changes

- Select code sets by searching through codes rather than manually clicking them
- Pick the current code set with a popup on the workbench.
- if “detached result sheets” is selected in the preferences, selecting *manage code set* from a document window does not move the workbench forward
- source document windows now scroll to top of file when opened
- result windows now save the current position and the marked records
- project window now autosaves if there are no untitled source documents and the project has a file name
- {!setrepeat code=”value”} metatag to set repeat values directly (useful if {code}value{/code} seems inappropriate
- File sets to allow you to organize your files into groups on the project window
- File sets can be picked from a small menu on the workbench.
- Result files are held in their own set, listed on the File set menu (either from the project menu or the workbench “set” pop-up menu), not through the radio button on the workbench.
- Project window now returns to the last code set and file set when opened.
- Several menus are now case insensitive in their sort order

Bug fix

- manage code sets now reloads the data so that the full set of codes is displayed.

**TA 2.35b6**

1. New report: Summary report, basically incorporates the grouping, summing and counting features of dataBoiler (+ an awareness of codes) -- See **Data Summary How To** for details.

2. Code browser and result window are now resizeable to better manage long code names

3. Sorting by code level

4. Fixed document window responsiveness with large code lists

5. Jump to project/workbench button on results window.

6. Fixed handling of save and save as in both documents and results so that they better inform the project of changes

7. Reports are handled as traditional windows

TAMS Analyzer 2.35b11 Release Notes

Changes
1. Interface changes for panther
2. change the number of commands that redrew the color codes
3. source compiled on gcc 3.3

Bug fixes
1. check for null characters for matching MWFiles on opening a project

TA 2.35b13 Release Notes

This release introduces a slightly improved data summary interface and one bug fix involving marking and unmarking records.

1. Data summary. Previously the data summary sheet would clear its values each time it was loaded. This prevented developing reports iteratively by trying some things out before saving the report with a name. The new sheet keeps the former value and introduces a reset button for the user to clear the current values when s/he wants to start a new report.

2. Mark and unmark records would show the first records data rather than the currently selected record. This has been corrected!

TA 2.36b9 Update

The major new feature in this release is that both auto sets and data summaries can remember the sort order of the data. For it to remember users have to use the sort up and sort down menu options NOT the sort button on the results window.

Changes

1. Sort order can be stored in an autoset and data summary
2. Various options including select and select all now do not change the selection.
3. Bug fixes in data summary

## TA 2.37b6 Release Notes

Major improvements in speed including simple and non-simple searches. Small bug fixes as well.

Changes

- improved parsing engine; dramatically speeds up simple, non-simple, section searches, loading of documents, etc. (doesn't effect tag coloring operations—that will be worked on for next release)
- the escape character '\' now is copied into the results window.

Bug fixes

- bug in which closing a result file did not remove it from the workbench fixed
- setrepeat spelling fixed in the metatag submenu

TA 2.38b2 Release Notes

- Continuing speed increases for coloring tags, diagnostics, inter-rater reliability, and reports.

- Added workbench button (labeled "<<") to create a search criteria from everything in the hot code list. Useful for looking at all the codes in the current code set, and nothing else.

- Bug fix for code coloring when a virtual intermediate code is used (e.g., fear>through>media where there is no defined code fear>through; would default to base color rather than inherited color).

TAMS Analyzer 2.38b4

This is a maintenance and performance release. Finally regex is useable!!!

Bug Fix

- remove file from project now removes the file from all relevant lists including file sets and search sets.

Changes/Bug Fix

- Regex searching is finally fast enough to be usable. Specifically the algorithm for replace all in the find/replace dialogue has been rewritten and the regex search from the workbench has been rewritten.

TAMS Analyzer Release Notes 2.38b5

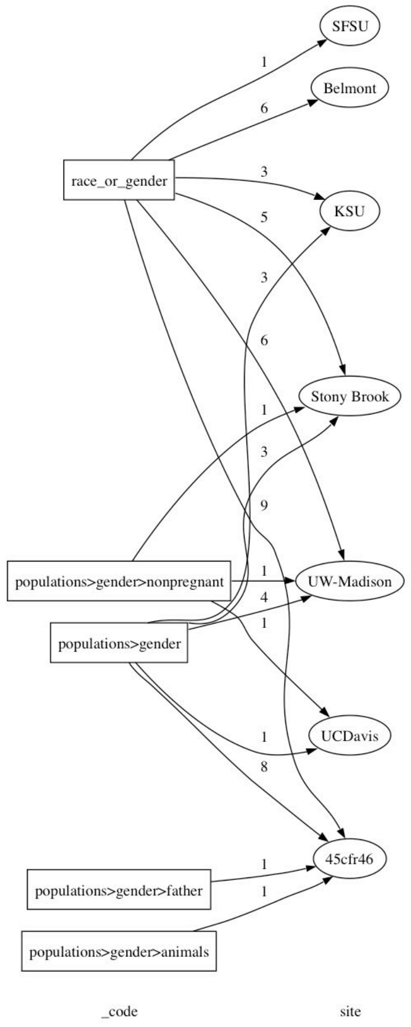
Bug fix release

- Fixes bug in which closing an untitled, unsaved file would not remove it from the project window.

TAMS Analyzer 2.39b1 Release Notes

This release contains two changes, one cosmetic and one substantive. The cosmetic change is that the report menu, previously under the Coding Menu, has been moved to the menu bar.

The second change is more important. TA can now produce files for the Graphviz program, a program for producing graphics like the one below, from result windows:



These graphics can be powerful tools for analysis and presentation. Graphviz for OS X is available for free from http://www.pixelglow.com/graphviz/download/. It is also available as an X11 application from the fink package system ("fink install graphviz"). With a result window selected pick Reports->dotGraph Output.

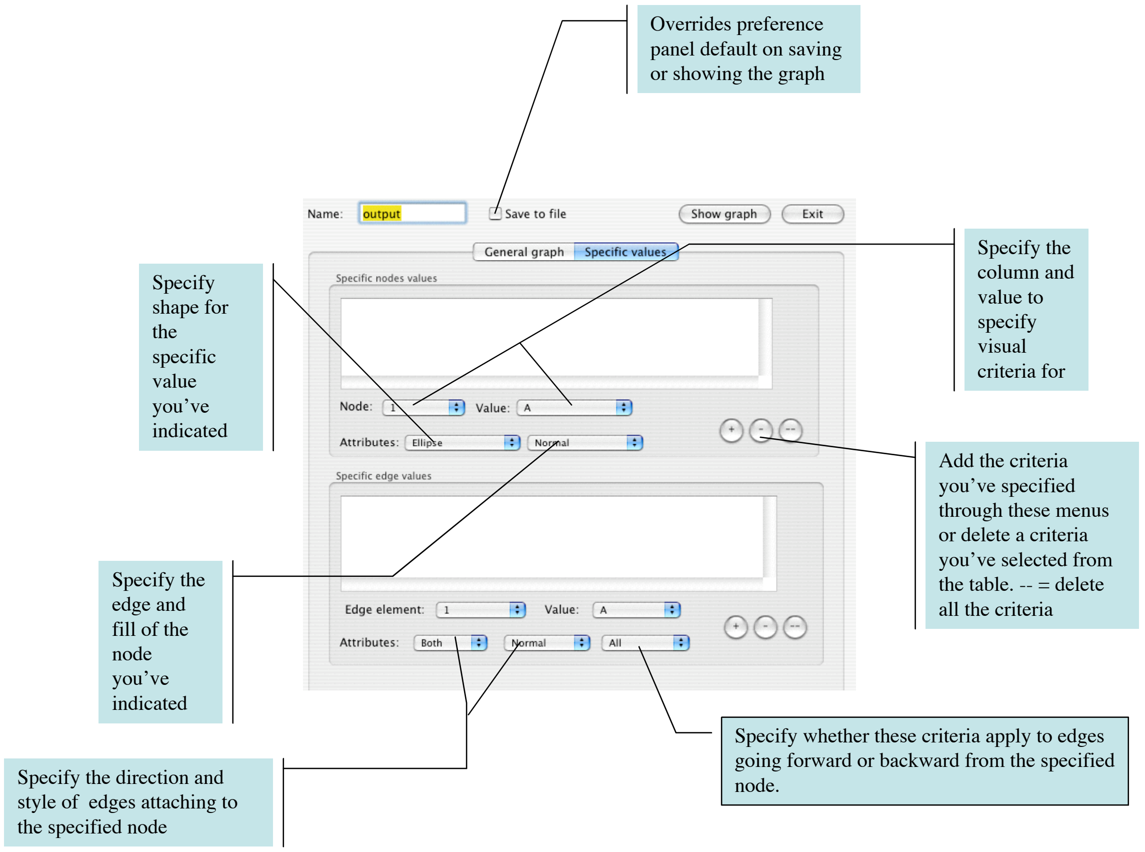
Documentation for this feature is given in a "Dot Graph How-To" file in the documentation folder.

TAMS Analyzer 2.39b8 Release Notes

This release improves on the graphviz integration with TAMS Analyzer. There is now a user preference (in the search tab) which allows users to choose whether to pop-up the Graphviz application showing the current graph or save it to disk. Furthermore there is a check box in the dotGraph pane which allows you to override your preference so that you can preview it and then your graphic to file (as a .dot file).

Why would a user want a file saved to disk? If you're using the X11 graphviz release, that will probably be your preference. Everyone else should go get the latest Aqua version of Graphviz; put it in your Application folder and turn off the "Save Graphviz files" option and just let TA open graphviz and display the image for you.

Other new features include the ability to specify characteristics for specific values in a repeat field (so just Bob in a repeat field called {name} is given a square node for instance; or only edges going forward from Bob will be dashed, but not those from Sally.). This allows a great deal of control over the appearance of the final graph! Here is a brief diagram of the new tab in the dotGraph interface which controls these criteria:


Overrides preference panel default on saving or showing the graph
Name: output
Save to file
Show graph
Exit
General graph
Specific values
Specify the column and value to specify visual criteria for
Specify shape for the specific value you've indicated
Specific nodes values
Node: 1
Value: A
Attributes: Ellipse
Normal
Add the criteria you've specified through these menus or delete a criteria you've selected from the table. -- = delete all the criteria
Specific edge values
Specify the edge and fill of the node you've indicated
Edge element: 1
Value: A
Attributes: Both
Normal
All
Specify whether these criteria apply to edges going forward or backward from the specified node.
Specify the direction and style of edges attaching to the specified node

Finally, I've added a third type of graphviz display called Tree which connects each level (repeat or universal code) only to the next most code (going left to right in your results window). See the revised version of the “dotGraph How To” file for an example of each form of graphviz output. (this is in the docs folder)

New features:

1. Preference allows user to have TAMS automatically open and display a dotGraph file or save it as a file to disk.
2. Switch to override the preference given in one in the dotGraph dialogue
3. Ability to specify characteristics for specific nodes, or edges connecting specific nodes.
4. Tree type graphviz diagram
5. Revised dot Graph (graphviz) How To

Get Graphviz at http://www.pixelglow.com/graphviz/download/
(it’s free as in speech)

TA 2.40b9 Release Notes

This is a major new release with the following added features:

- Dot graph dialogue improved so that order of fields and the code level can be set within the dialogue. Set the order of fields by dragging the names, code level has its own field.

- New dot graph report for the current code set by picking “Reports->dot Graph Output” from either the workbench or document windows.

- Forward and back browser-style buttons added to the result window (with supporting menu options on the Results menu)

- New metatag. Another attempt to save on inserting {!end} statements. {!lastrepeat X} or {!last X} in which the next repeat code after repeat code X is treated as if it had an {!endsection} before it. This is sort of the reverse of {!inner X} which acts as if there is a {!endsection} before every instance of X. !last and !inner cancel each other.

- Named code set fixed so that it properly closes and saves

- Warning added so that users cannot create empty code sets

- Bug fix for delete codes in selection

- Updated TAMS documentation

- New TAMS documentation: “Analysis for beginners”

- New packaging: graphviz is included in the TAMS release.

- Project file upgraded to Xcode Native format

TAMS Analyzer 2.41b11 (AKA TAMS AV—i.e., audio/visual) Release Notes

In some ways this is a more significant release than 2.40. In this release we add support for working with audio documents. Basically TA has added a full transcription machine for working with multimedia.

Basically, TAMS just includes a little transcription machine with a few smarts:

- Attach a quicktime friendly file (including mp3, aiff, etc.) document to a file.
- Insert time codes into the document with a key stroke (including codes)
- Control the playback of the audio from the keyboard
- Control the rate of playback
- Backstep from the current location (seconds set in preferences)
- Backstep automatically on playing
- Select a time (in seconds) and jump to that location in the media file

See the "Audio-Visual How To" for details

TAMS Analyzer 2.42b7 Read Me

- Program monitors state of play and adjusts the stop/start button appropriately
- Set colors for tags using **color well** including the preference panel default color
- Set metatag colors independently of tag colors
- Result windows displays color of tags
- Add code and recode can now fill in the new code (partially) by picking the code from the Old code menu.
- Bug fixed that didn’t copy white space over to results windows

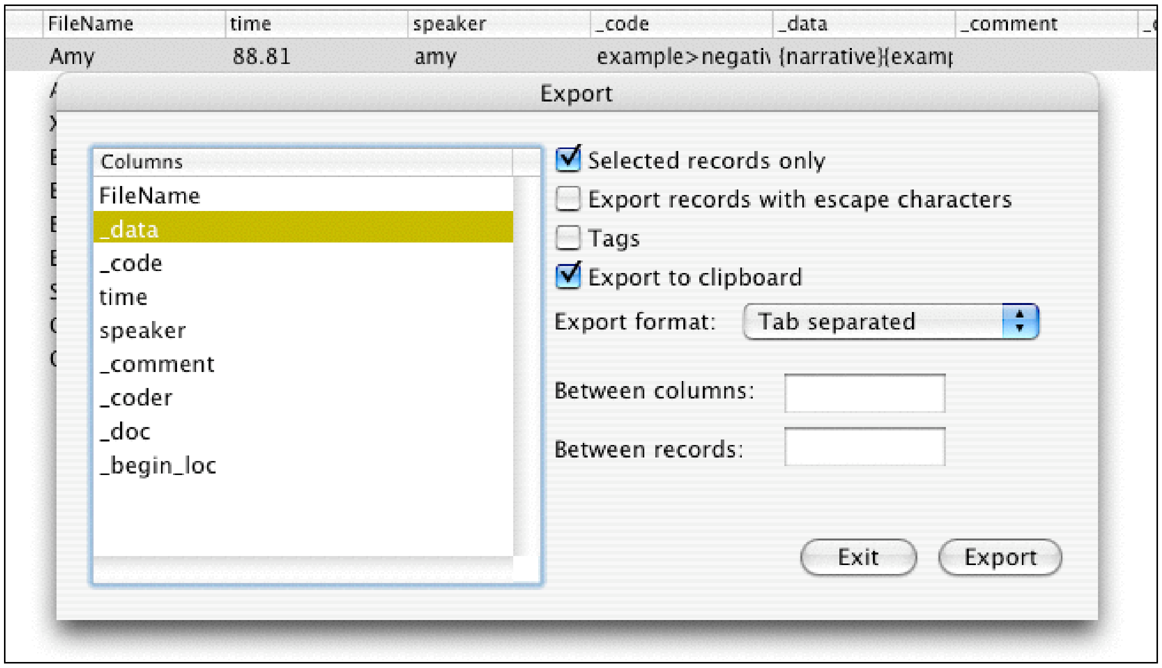
TAMS Analyzer 2.43b6 Release Notes

1. This release features keyboard access to toolbar buttons. The first 10 tool bar buttons are given command keys command-option 9-0. They also appear listed in a new Toolbar menu that also gathers together the functions that deal with the toolbar.

2. Also there is a new edit menu option of copying without tags. This allows you to copy from either the results menu or the document windows and have the program automatically remove all traces of TAMS.

3. String searches now reveal the color of tags if auto tag coloring is enabled.

4. New File->Export Data menu option/dialogue. Allows much more control of export of data to either file or clipboard. You can select the columns you want to export, you can rearrange the order of columns to be exported, you have a variety of formatting options for export. You can also create your own export format for export by picking "Other:" from the "Export format" menu and entering the strings used to delimit "fields" and "records" (columns and rows, respectively) into the "Between columns" and "Between rows" fields. These can include \n, \r, and \t to represent new line, return, and tab characters. You also can choose to include or not include tags, interpret escape characters, export only the selected (or all data) and, to reiterate, have the data saved in a file or put on the clipboard. Powerful stuff!



| FileName | time | speaker | _code | _data | _comment |
|---|---|---|---|---|---|
| Amy | 88.81 | amy | example>negati | {narrative}{examp | |

Export

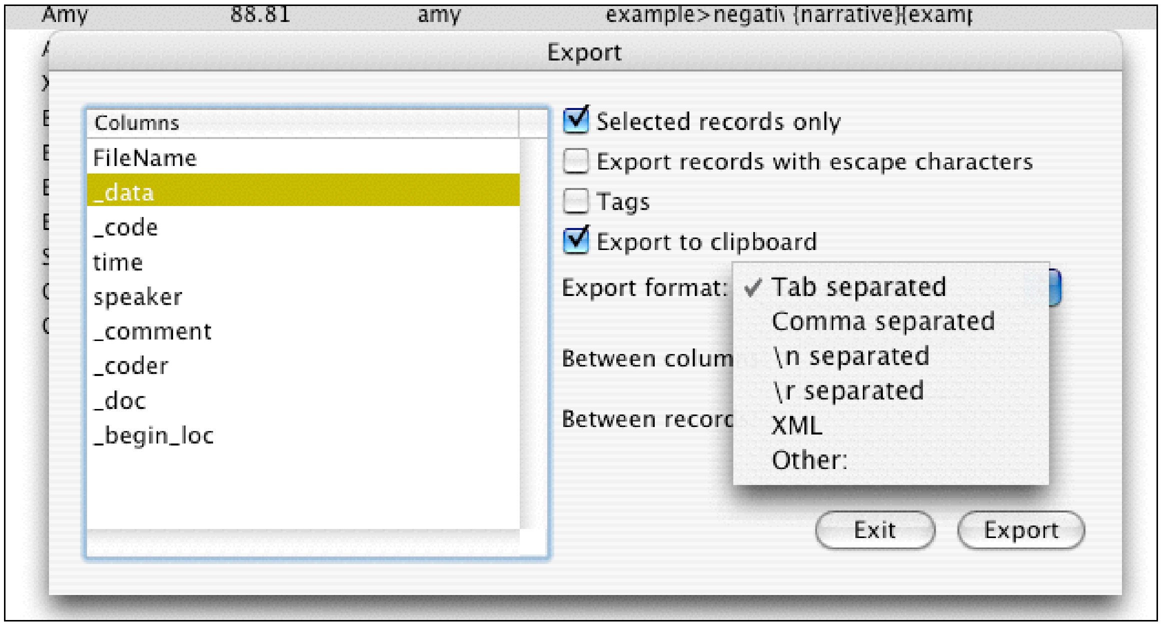
Columns
- FileName
- _data
- _code
- time
- speaker
- _comment
- _coder
- _doc
- _begin_loc

[✓] Selected records only
[ ] Export records with escape characters
[ ] Tags
[✓] Export to clipboard

Export format: Tab separated

Between columns:

Between records:



Columns: FileName, _data, _code, time, speaker, _comment, _coder, _doc, _begin_loc

[✓ Selected records only]
Export records with escape characters
Tags
[✓ Export to clipboard]
Export format: [✓ Tab separated] Comma separated, \n separated, \r separated, XML, Other:

5. Hide tags in the results window preserves whitespace.

6. Bug fix to drag and drop in dotGraph which caused crashes only the first time run.

TAMS Analyzer 2.44b6 Release Notes

TA 2.44 changes the behavior of certain operations, adds functionality and otherwise tries to smooth the product.

1. A work menu that you can add projects to. (File->Work) Add projects when the workbench is in front.

2. Autosets can now call autosets. This is a little dangerous, but seems useful. I try to check and warn if it seems like there might be an infinite loop.

3. Refresh tries to recreate the last results set. Previously you'd have to create an autoset and run it after refreshing. This version tries to recreate what you have showing. I say "tries" because the following operations will throw it off:

- select, add and subtract marked or unmarked or remove marked or unmarked or remove selected or unselected.

- Collapse in either direction.

- set math with named sets (should work fine with autosets)

4. Recode and add code now fill in the New Code blank with the first marked entry's _code value

5. Results window redesigned around toolbar

6. Default sort in the results window now registers in the sort stack so it can be used in autosets and data summaries.

7. Bug fix for a problem I've known about and have remained silent on for a while: Using the forward and back would mess up the autohistory that is used to populate autosets.

8. Bug fix in regex replace all in the document find/replace

TAMS Analyzer 2.44b10

A few toolbar additions and a bug fix. This is mostly a maintenance release.

1. Ability to add command button to document toolbar (see new *Toolbar How To* in the How To folder)
2. Many more result window toolbar buttons
3. Bug fixes for
    a. Working with tabs in documents. Program was not converting them properly.
    b. Remove from selection… now adds an item to the status panel
    c. Reordering items in the file list when there is no data sets now “sticks” (i.e., is remembered when saved)

TAMS Analyzer 2.45b9 release notes

This release includes:

- More tool bar buttons
- The ability to select a code name in the text (in either the results browser or the document window) and find the definition
- The ability to find the definition of the code in the _code column in a results window.
- The ability to remove line numbers from documents

## TAMS Analyzer 2.45b12 Release Notes

TA 2.45b12 is a bug fix release. A serious bug that would not allow you to close result windows with the close command (or apple-W) has been remedied. Another bug involving moving files to the search list has been fixed. Also, I have set back the official version of Graphviz.app since the new release produced errors in generating the scale for the dotGraph output.

Additions:

- Code browser toolbar button available in both documents and results windows. Code definition button available in document windows (redundant with def. button but has aesthetic appeal)

Bug fix:

- Result windows now close correctly.
- Add file to search list bug fixed

TAMS Analyzer 2.46b1 Read me

This version of TA includes new icons, an updated version of graphviz and two significant refinements to current features. Collapse up and collapse down (a way to reduce data in the results window) now updates the status bar and the "stack" so that users can use the back button to "uncollapse" the data. If document windows are dirty the user is now warned about the status flag before add codes, recodes, delete codes, and comments. Finally, the summary report has a new feature that allows users to establish the sort order based on the groups that the summary is using. This is worth explaining a little more:

Previously, the summary report would simply start at the top of the visible data and move to the bottom watching for breaks (changes) in the various fields that the user indicated they wanted grouped and keeping track of the count of the records in that field. This meant that if you had grouped the _code column and hadn't sorted it and a code (say a>b) appeared in two places in the results not next to each other, a>b would be listed twice. The implication of this is that it was important for users to sort their data first to keep values they wanted counted together. Then they would have to create groups that were almost inevitably the same as the fields they had been sorting! This always felt redundant to me.

In this version there is an option (selected through a pop up menu) that allows the user to use the groups to presort the data, i.e., you don't have to sort the data first If the user groups _code as a code field and Name as an alpha for instance, selecting this new option ("Use current groups to sort before doing report") then the data will be sorted first with code and then name as an alpha using the sort within.

Note that this is a lot less flexible an option for the user than the old way. Specifically the program (using this option) will only sort up, will only do non-case sensitive sorts, will force the code level to the one used for the group (which I find is often not what I want: I'll sort the data with one code level and group _code at another), and will use the same date format (the one currently selected in the results) for every date sort. If you can live with these assumptions and restrictions this new item is a blessing. Otherwise do it as earlier described: sort your data first and then create your groups. Pick the second item from the same pop up menu ("Use current sort before doing report") and your work sorting the data will be attached to the summary report if you wish to run it again.

In short:

- New icons
- Improved handling of Collapse up and collapse down in the results window
- Data summary report can now use the groups to sort the data

Release notes for TAMS Analyzer 2.47b1

This release adds both support for beginning projects as well as advanced projects. For beginners, the program now defaults to an “unstructured” mode in which the close tag of every coded passage is now taken as having an !endsection immediately afterwards. For most projects this means you can use repeat values without having to use !end or !endsection anywhere in your documents. **However**, it also changes the default behavior of the program. To force traditional behavior researchers who have used !end, !endsection, !last or !inner should add a {!struct} (for a structured document) in their init file or at the top of the first document. This will prompt TAMS to act as it has before: waiting for !end, !endsection (or !endsections implied by !last and !inner) before matching repeat values with data values. To understand this new behavior read section IV of the user guide. (There is also an {!unstruct} metatag which forces the program to !endsection after each close tag, if you have turned off this behavior). Note in its default, unstructured mode section searches and empty searches will be meaningless!

For advanced users the new program offers a way of combining heterogeneous types of data (interviews, memos and fieldnotes for instance). The new version allows you to “map” one repeat code onto another. So if you have a repeat code column called “speaker” from interviews early in your search list,  but your newspapers are coded with author, you can add {!map author->speaker} at the top of your newspaper file and occurrences of the repeat code author will appear in your speaker repeat code column. Repeat codes must map to repeat codes, only repeat codes are supported—not universals or data codes. !map can take multiple mappings: {!map author->speaker, x->y, z->aaa}. This means put any instances of author in the speaker column, of x in the y column and of z in the aaa column. To zero maps you’ve already created there is a {!zapmap} metatag.

New features.

1. Count of codes now appears on the workbench, shows how many codes are in the current code set.
2. Default behavior assumes {!endsections} appear after every close tag. For most purposes this means you don’t need {!end}, {!endsection}, {!last} or {!inner} (any of these will turn off the default behavior). To force this behavior off use the metatag {!struct} (for structured document) in you rinit file or at the top of the first document in your search list. It can be turned on through the {!unstruct} metatag.
3. {!map X->Y} takes repeat code X and puts any instances of it into a Y column if it exists. It basically reassigns any instance of X to be a Y. The map created by !map can be emptied through {!zapmap} metatag.
4. Data summary can now convert sorts into groups (a recent release did the opposite: it would sort based on the groups created).
5. Large app redesign (a little less dramatic, but looks better on a small dock)

TA 2.48b3 release notes

1. This release refines the behavior of the !inner metatag with four additional metatags to enable or disable different behaviors of !inner. Traditionally {!inner X} would not interpret the first occurance of X (i.e., put a virtual !endsection before the first occurance of X). While this is still the default behavior, it can be turned off by putting a {!noskipinnertopofdoc} (of course available from the metatag menu) before the first occurance of X. If you want this behavior reenabled put {!skipinnertopofdoc} at the top of the document.

Similarly, in prior versions, inserting a {!end} in your documents explicitly would cause the next occurrence of X not to be interpreted. Strangely (and I have no idea of my thinking at the time) {!endsection} had no such effect. Now both, by default, do not interpret the subsequent occurence of X, unless you put {!noskipinneratend} in your document . To re-enable the default behavior put {!skipinneratend} in your document as needed.

2. There is a recolor tags menu item (and toolbar item) for document windows. This turns the text black and then colors the tags. The previous separate commands for turning text black and coloring tags are now in the color tags submenu.

3. Counts of files and searchlists are now located in the lower left corner of each list on the workbench, making them equivalent to the code list format.

4. Various wordings of menu items have been adjusted to clarify their function.

5. More support for the use of the comment field. You can now append a comment to codes in an arbitrary portion of a document through the {!setcomment X} and {!appendcomment X} metatags. !setcomment  X clears any existing comments and sets the current comment to X; !appendcomment X adds the given comment to previous comments set by setcomment and !appendcomment without clearing the previously set comment. Comments can be selectively removed with the {!endcomment  X} where X has to be verbatim what was set in a previous !appendcomment or !setcomment. The last comment set by !setcomment or !appendcomment can be removed with {!endlastcomment}. To cancel the existing comments use {!clearcomments} (or use {!setcomment} or {!endcomment}  with no argument provided). Comments set by !setcomment and !appendcomment are added to any comments provided in the close tag.

6. Equivalent metatags have been established. These are aliases or synonyms to the tags that have been there all along. It's just that as the program has grown, the original meaning has changed, so that I thought it would help to have alternatives that might be more descriptive to the reader.

| Original metatag | Synonymous metatags |
|---|---|
| Repeat | context<br>contextcode |
| Inner | first<br>firstrepeat<br>innerrepeat |

| | |
|---|---|
| Last | lastrepeat |
| Dummy | comment |

**TAMS Analyzer 2.49b2 Release**

May 20, 2004

This release adds to TA's support for multimedia documents in three critical ways.

1. Result windows can now load multimedia files and play them from the time code associated with the selected row. The "Code AV time units when inserting" preference must be checked and the time code must be the one listed after that option in the program preferences (defaults to _*time*). This time code (e.g., _time) must be declared as a repeat value.

Given those conditions, researchers can select a row from a results window and pick Results->Play media. A sheet will appear (detached, so you can move it relative to the results window) with a Quicktime player positioned to the time code value of that row. In addition to the standard Quicktime controls, researchers can move up and down through their results window listening to the attached media for each row at the appropriate time indices with a previous and next button. Furthermore, with the media player showing, researchers can continue to select rows in the results window and pick Results->Play media and have the player jump to that media file cum time code value. The play media function is also available as an optional toolbar button.

2. The media player can now be used to jump to locations in transcripts. The reverse has been true for a while, i.e., researchers have been able to select a time code and pick Coding->Audio/Visual->Jump to force the Quicktime media player to index that location. Now the reverse operation can occur. Researchers can position the "scrubber" (i.e., the slide bar in the media player) and jump to the nearest inserted time code in the transcript. This is done through Coding->Audio/Visual->Find nearest time code

3. Media time can either be formatted in seconds or in traditional hh:mm:ss format. User preference panel determines which format is seen. Formats can be converted (if the time units are coded) in either direction through new menu options on the Audio/Visual menu. This conversion can happen either to the open file or, if picked with the workbench in front, to the entire search list.

TAMS Analyzer 2.39b4 Release Notes

May 26, 2004

Bug fix release

1. Improved handling of items in the Work menu. Users can now delete non-locatable items.
2. Users can now delete all (clear all) from the Work menu
3. Bug fix to results media player
4. CFBundleIdentifier established so that program can be used on managed accounts on laboratory machines
5. Definitions no longer have as default time-date stamp

## TAMS Analyzer 2.49b7 Release Notes

May 28, 2004

This is an important bug fix release.

1. Several fixes to “regular expression” find and replace.
    a. REGEXCOUNT variable is restored so you can enumerate your regular expression find/replace in documents.
    b. Multi-line regular expressions now work. (this is not the multiline pcre option, but just that regular expressions with \n’s are now handled correctly)
    c. Memory management improvements to regex searches from the work bench. These searches should now be slow but not deadly.
    d. Regex searches from the workbench allow the researcher to look for substrings of regular expressions. The number of the substring is indicated in the Coder ID: field (though with this version changes the prompt for regex searches from “Coder ID:” to “Substring:”). Substring 0 refers to the entire regular expression.
    e. Workbench empty switch enables the multiline feature of regular expressions (Reminder: exact switch enables case sensitive regular expressions).
    f. PCRE library updated to version 4.5

2. Fixes made to parsing engine
    a. There were problems with empty searches. In some cases the search would be broken. This was most evident in empty searches. In addition, the last record would be wrongly matched with its file if end of files were treated as !ends. These have all been fixed.

## Why Regular Expressions are important

Some qualitative research packages have “autocoding.” Granted, I’m not sure I know what that means. But if it means code every sentence with the word Xanadu as code Y, then yes TAMS Analyzer can do it. You’ll need to use regex searches from the workbench. There will be three steps to this: first we find all the sentences; second we mark all (perhaps sifting through these sentences to make sure they are appropriate), third, and finally, we “add code” to assign the sentences a new code.

Step 1: finding Xanadu. On the workbench pick regex as your type of search. Then type in “[^\.\n]*Xanadu[^\n\.]*[\.\n:]” (without the quotes. Let me translate. This means in regular expression speak “find every passage of 0 or more characters that don’t have a period or a return followed by the word Xanadu followed by 0 or more of any character ending in a period. Definitely search the web for tutorials on regular expressions. Now press search, and wait, and wait. It can take many minutes to find everything.

Step 2: Review the results and if you need to be selective, you might mark the ones you want and pick Select marked.

Step 3: Pick Recode->Add code.  If it's a new code use the bottom half of the dialogue, or use the top half if it's an existing code you want to apply to these sentences.

Voila

Hint #2. To code whole paragraphs you might try this regular expression:
"([^\n]*Xanadu[^\n]*)\n" and enter 1 as the substring. This will put the code before the return at the end of each paragraph. Press the search button and pick up with step 2 above.