PhUSE 2006

**Paper PD02**

# MicroContent with TiddlyWiki

David J Garbutt, BSI AG, Baden, Switzerland

## ABSTRACT

The last year has seen a revolutionary tool emerge in the space of personal information management and information dissemination. Developed out of the WikiWiki it offers a must-have tool for all of us, right now. Who knows what it can become? I will demonstrate TiddlyWiki and some things in your professional life it can help you with. You can use it for building rich hyperlinked documents, for managing simple lists, technical documentation, personal blogs and up to a full PIM system. It can replace Word and Excel for many jobs. And it is extensible, and did I mention it – free!

If you cannot attend this introduction then visit ***http://www.tiddlywiki.com/*** and other links in the REFERENCES section and scattered through the text to discover more.

## INTRODUCTION

This paper aims to bring TiddlyWiki to your attention if you have not already heard of it. I am going to explain what a TiddlyWiki is, give examples of what it can be used for and finally, recommend some sources for further information. We deal a lot in our professional lives with the so-called paperless office. This means reading large manuals, documenting our programs, making and signing validation records, scanning SAS logs for errors, planning projects, scheduling our daily tasks, searching for information, and communicating some of this work to others.

The computer tools we use for this have not changed much in 15 years. We now have Google and the web but we don't write much HTML and many documents are still written without hyperlinks or any internal navigation at all. We use mailing lists for asking our peers about technical issues, we may have internal bulletin boards and the intranet, we have e-mail, and we have text files holding our technical secrets.

A new development in the content management stakes has been the WikiWikiWeb; these are easy to edit web sites that can be built from the contributions of individuals, gradually. The Wikipedia is an example and has long ago progressed beyond being merely interesting. Wikis have marked advantages over other ways of information sharing: they are more permanent and structured than e-mail threads or bulletin boards but they do require a web server (even the personal ones). They do not all support categorization of pages and generally the category structure is somewhat rigid and 'top-down'. Many implementations have a rather poor interface and appearance. All the ones I have seen navigate a page at a time, regardless of how much information is on the page.

TiddlyWiki is a new implementation of a WikiWiki as a local, single HTML file with an interface run by client-side JavaScript. It is a non-linear, personal, web notebook filled with micro-content and it has created a new niche in the web and in the paperless office.

## MICROCONTENT? A NON-LINEAR PERSONAL WEB NOTEBOOK?

A TiddlyWiki keeps 'pages' and visible until you exit or close one. Each 'page' can be opened independently. This piece of genius – to change what seems 'normal' and 'the obvious way to do things', A web page should be a web *page*, after all – is what really sets TiddlyWiki apart from other text and hypertext systems. It may seem a simple change to the interface but a document which is a ***single page application (http://en.wikipedia.org/wiki/Single_Page_Application)*** with its own GUI is something new.

### WHAT IS MICROCONTENT? :

The idea of ***micro-content (http://www.tiddlywiki.com/#MicroContent)*** is straightforward and is an answer to the question – what is the important unit of information for an application? In good writing the unit is a paragraph:

> *The paragraph is essentially a unit of thought, not of length: it must be homogenous in subject matter and sequential in treatment,*
> Dictionary of Modern English Usage –(Fowler 1926*)*

The purpose of building systems around micro content is to free paragraphs from their prisons; enable easier linking of information, and provide an better environment for storing and retrieving information. Another more practical reason is to reduce the marginal cost of adding information and preparing it for publication. The unit of micro-content is called a tiddler[1].

The tiddler is a unit because it is the smallest piece of text: with its own name, that can be tagged and that can form the destination and source of links.

### MACRO-CONTENT DOCUMENTS CONSIDERED HARMFUL

Most of the tools we use to deliver information are delivering macro-content. This Word document, for example, acts as a unit in many contexts. It is implemented as a single file and has to be treated as such when read, downloaded or opened. But all the contents are not uniquely tied together. A good example in this document is the abstract. It is extractable by copy and paste and can then be published separately. But that separate thing is really a different copy in a different macro-content document. They can easily get out of step. And we cannot call or open the macro-document with an argument that says, “just show me the abstract”. The common ways to make (even electronic) documents are navigation tools that have not changed since the invention of the index and the table of contents.

A subtle problem with macro-content is that putting everything in one document means the document must have a ‘reader level’. Because a unitary document cannot fulfill more than one purpose because *all its content is visible*.

To take an example, if we include technical reference material in a Quick start user guide then beginning user cannot turn that content off and has to bear the cost of looking at those sections and disregarding them. Therefore, the existence of a macro document entails an audience and implies that we must design (which means, restrict) the content, and so we have to make several documents: a quick start guide, a users’ guide, a reference manual, a technical reference…, and so on. None of these documents can easily share content – they must copy it. And someone needs to maintain these multiple destinations and this does not always happen. Furthermore, the division is done by the authors and how can they know exactly how to classify this information? If a piece of information is called technical, it may be relegated to the detailed manual; but what if the user is prepared to learn enough about that piece because the problem it solves is worth it to them? These problems with the single document model are issues we do not even see because we accept implicitly that that is what a document *is*.

But a micro-content based system breaks this mould forever because it can *both* hide irrelevant material *and* make it all available: in one document.

### A PERSONAL, NON-LINEAR, EXTENSIBLE WEB NOTE BOOK?

*Personal* – A Tiddlywiki is a single HTML file. You can transport it with you via USB stick, keep it on your web site for download or browsing anywhere, and edit it locally using a file:// URL. It is not a server side tool[2], it needs no IT department permission or involvement to set one up, and you need no additional tool except a modern web browser (Firefox, Internet Explorer 6, or Safari). TiddlyWiki is free and open source; there is no license to expire; your data will stay yours.

*Web Notebook* - A TiddlyWiki is a single HTML file containing about 300k of JavaScript which manages the display and addition of tiddlers. Not all tiddlers are displayed initially[3]. Links and searching facilities are included. The formatting is controlled by an included CSS stylesheet and is freely editable. It is easy to copy and clip notes from other sources and links.

*Non-Linear* – there is no fixed order to the tiddlers in a TiddlyWiki. They are displayed in the order you, the reader, chooses. From a table of contents you can skip to the abstract, then open the references, or just read the section on personal diaries…. it is all up to you. Structure is given in several ways:

1. *links* between tiddlers,
   Linking is very easy, so easy you can do it while typing, safe in the knowledge you can find missing tiddlers later and fill them in. You can easily get a list of all the tiddlers linking to a tiddler.
2. *tags* you give each tiddler.
   Tags are words, or phrases, related to the tiddlers content. Tags can be explored themselves so they also can form paths through the tiddlers. Tags can also be used to generate clickable, dynamic, lists showing what tiddlers match.
3. *by constructing tiddlers for summaries, tables of content or topic lists*
   You can build a tiddler with content from other tiddlers without copying it. To make a tiddler that appears as a single long text. There are various ways to do this (see Aggregating Tiddlers below). You can also implement such a page as a nested disclosable list making it look like a document outline. As of the time of writing TiddlyWiki does not have a way to impose an outline order on your tiddlers[4].

[1] A *tiddler* is British word for a small fish, the kind so small that even small boys will throw back if they ever do catch one.

[2] There are several server-side implementations available and for situations where more than a few people will be contributing they have definite advantages. There is also the new ***GroupEditplugin*** and the hosted ***TiddlySpot*** service

[3] DefaultTiddlers holds a list of what tiddlers should be displayed at start up.

[4] Discovering this was a moment like the one when I realized that the ***Korn shell has no string functions***. I thought ‘then, programming is not possible at all!’ But as I used it I realized the methods it

4. *opening defined groups of tiddlers as a unit*
   this is a new method made possible by the TiddlerGroups (***http://www.tiddlytools.com/***) plugin. A drop down box lets the user access named groups. The lists are held in a special tiddler.

*Extensible and customizable*. There is a growing community of developers working with TiddlyWiki and they are exploiting its plugin architecture with a huge array of modifications and extensions. Adding a plugin to a tiddlywiki is so simple it is almost cruel (for us software developers). Extensions include the ***taggly-tagging*** (at http://mptw.tiddlyspot.com/) variant, and many presentation and content developing tools, fun plugins such as a ***Sudoku assistant*** (http://danielbaird.com/tiddlywinks/sudoku09.html), image plugins, slideshows, encryption, data storage the list is almost endless. You can write your own macros, for example, I made this one for my ***Unix tutorial*** (http://daveg.tiddlyspot.com/#PhUSE-Talks2006%20%5B%5BGnit%20your%20own%20-%20creative%20Unix%20solutions%20for%20SAS%20programmers%5D%5D) and Tutorial stream TU 04) to create links to Unix man pages on the web by altering the Wikipedia link generator macro

```
config.macros.man = {};
config.macros.man.handler= function(place,macroName,params) {
   var key=params[0];
   wikify("[["+key+"|http://unixhelp.ed.ac.uk/CGI/man-cgi?"+key+"]]",place)
}
```

To install and use it (or any plugin) place it in a new tiddler, tag the tiddler systemConfig, save the TiddlyWiki, reload into the browser. Add this to your tiddler about the awk command: <<man awk>> and this link is generated: ***awk***. (http://unixhelp.ed.ac.uk/CGI/man-cgi?awk)

## USING TIDDLYWIKI

### ADDING INFORMATION

To add new information, click the New tiddler button (generally on the right-hand-side side bar). An edit box opens with three fields: the title, tags and content.

### FORMATTING YOUR INFORMATION

Tiddlers generally contain just text, not HTML. They are edited via a normal HTML text box. The sidebar (on the right) displays tabs that list all tiddlers alphabetically, by date, and also all the tags. Open the tiddler, tag it. Done.

TiddlyWiki is dynamic and lists etc. are updated automatically. Pretty well the entire content is held in editable tiddlers. This includes plugins (in JavaScript) and dynamically loadable CSS stylesheets.

### TABLES, BLOCK QUOTES, PREFORMATTED , HEADERS, ...

There is a special syntax of text indicators for ***formatting***. Some formatting codes must be at the start of a line, for example

! is a header1 line,

!! is a level 2 header

# is a numbered list

## is a numbered sub list (of numbers or bullets)

* is a bullet point

** is a bulleted sub point (of a numbered or bulleted list)

> is a quote (indented, with a left bar)

These start of line codes apply to blocks of lines:

| begins a | table|

| first col| line2|

<<< to <<< marks a block quote

{{{ to }}} marks a block of fixed pitch text

Other formatting codes can be in-line.

text enclosed in {{{`text`}}} is rendered in a fixed pitch font.

@@ encloses highlighted@@ text

The sparkline macro draws a word-sized in-line graphic.

All the formatting is controlled by CSS style sheets they are dynamically loaded so changes take effect immediately. There are many different style sheets available and you can customize your own. Any changes are shown immediately. You don't need to be a CSS guru to guess that `font-family: Arial, Verdana;` can be changed to `font-family: “Trebuchet MS”;` to change the font.

### STRUCTURING YOUR INFORMATION

used worked were much simpler to use, more flexible and more powerful than the string functions a typical 3GL uses.

### *AUTO LINKING, EXTERNAL LINKS*

You make links in two ways: by enclosing the tiddler's name in double square brackets [[…]] or by using CamelCaseWords. This convention feels strange at first, but then becomes very liberating, because it is easy to type and create future links to be filled later in as you go. For ease of use there is no comparison to Word and other programs where you must open a dialogue box, highlight the text, copy the link and the text, and so on…, this a major distraction from writing, whereas I find TypingAPhraseWithExtraCapitals is not. Even [[typing a phrase like this]] is relatively easy.

Easily added links might seem like a minor advantage, but it is not. It is a common story in computing and interface design: what matters to the user is the marginal cost of adding value. If it is so easy that it doesn't make typing the content 'harder', then it will get done. If it is not 'easy' then it will not be done.

Links can be prettied up by adding a label and | as in [[how? |How do I hide the name of a long tiddler from my text?]] will appear as just ***how?***.

### *TAGGING*

Informal tagging is emerging as an alternative to pre-determined hierarchical organization. Its great advantage is that you can let the organization emerge out of the sea of tiddlers, and so you don't have to commit to a structure before you begin. This works very well for writing, technical writing and personal information. First because there are many ways people think and talk about the same thing. T ambiguity is a feature of human languages in general, and our brains can handle it from an early age. Children make puns almost as soon as they learn to speak. Second because you spend a long time gathering and sifting information before 'writing'. With a TiddlyWiki you can gather and sift, and then later structure and publish, in the sense of determining if the reader will see that topic. It can remain in the document.

Tagging can create structure content 'free'.
For example consider tiddlers with Unix programming tips. Tag each tip as you create it with the Unix commands used in it.
Now you can look for examples of awk by simply opening the awk tag and checking for where it is used. It could be said that this is covered by a search facility, and it is once you form the idea to look for tiddlers containing the string awk. And you do not know to look for that unless you already know roughly what awk is. Before you know that though you *could* look for tiddlers tagged 'Unix commands' or 'processing text' and then find the names of commands you did not know about. And all possible because the author spent 10 extra seconds adding the tag 'Unix commands' to tiddler awk. I suppose this is not *completely* free. But is there anything nearer?

It is possible to use tagging to implement hierarchical content organization, but this is not the strength of TiddlyWiki. I believe that once we get used to navigating a few TiddlyWiki documents we will see more clearly the limitations that hierarchically organized documents have and know to ask for more.

There are various ways to display tags in TiddlyWiki the sidebar list is always there, and for including in a tiddler the built in macros <<tag mytag>> (dropdown box) and <<tagger mytag>> (rendered as a list) are the primary ways, other ways using plugins with TagglyTagging, the tagAdder macro, the SiteMap macro and the TagCloud deserve special mention.

### *STRUCTURING YOUR DOCUMENT*

TiddlyWikis are structured with tags, links, and by adding tiddlers that serve as starting points for exploration. There is always an alphabetical index available, a list of all tags, a timeline, and a whole TiddlyWiki search facility.

There are also tabs showing all missing tiddlers (referred to with a link but not yet created), orphaned (no tiddlers refer to them) and shadowed tiddlers. Shadowed tiddlers have default content and are critical to the TiddlyWiki in some way. The lists of Missing and Orphaned tiddlers locate the places where there is editing work to do.

*Aggregating tiddlers*
When entering information into a TiddlyWiki from scratch it is easy to build small tiddlers as you go, consequently the most common need is to aggregate tiddlers. The built in <<tiddler>> and the <<slider>> macros and the widely used plugin containing the <<nestedSlider>> macros are simple way to concatenate tiddler content. Another approach is to create a list of tiddlers to act as clickable entries in a content page. Other list methods are the TagCloud and the SiteMap plugins already mentioned. There is also a plugin that gives a way to render content (or a named tiddler) as a pop-up box.

*Disaggregation*
The need for disaggregating comes when moving already written tiddlers into a TiddlyWiki splitting the content can be make it all accessible to re-use. The TiddlerSplitter macro is one of the few tools for moving information in this direction. It allows large tiddlers to be separated into smaller tiddlers using a variety of cues. The partTiddler plugin allows the labeling of parts of a tiddler so they can be linked to separately, and aggregated elsewhere if needed.

*Sequences*
It is perhaps true that the lack of a simple way to make a sequence of tiddlers go in a specific order makes TiddlyWiki better for reference documents, but I think this is also a question of planning and mapping out paths for the reader.

*Adding tags*
This is done in edit mode into the tag box. A drop down list is provided but for large documents with

many different tags the Intellitagger and tagadder plugins are recommended.

*Authoring tools*

Lists showing missing tiddlers, untagged tiddlers and orphaned tiddlers (tiddlers with no links to them) are always available. I find that tagging a partially completed tiddler with (say) ?? or adding link to the ?? tiddler (by adding [[??]] to your text) works well for labeling items to be completed. This flexibility means you can add things when the ideas strike you and help you check your document for completeness in those moments when inspiration has tiddled off.

## SHARING YOUR TIDDLYWIKI

You can install a TiddlyWiki on a web server, or on a disk share from whence it can be opened by your readers.

If that sharing results in the same URL then you will find the permalink tool very useful. It records a URL with all the open tiddlers listed, perfect for sending a custom document to someone.

Currently sharing authoring rights is not possible although a new plugin address that issue and it is a growth area for future TiddlyWiki versions.

## USES FOR TIDDLYWIKI

I cannot cover all uses in this short time but here is a sampler. Enjoy.

### JOURNAL, DIARY AND DAY PLANNER

The standard TW includes an New Journal button but this concept has been extended and further developed. There are now two popular implementations of David Allen's 'Getting Things Done' methodology (***dcubed*** (***http://www.dcubed.ca/***) and ***MonkeyGTD*** (***http://monkeygtd.tiddlyspot.com/***) ) and the for fans of Joel Spolski's ***Painless Software Scheduling*** (***http://www.joelonsoftware.com/printerFriendly/articles/fog0000000245.html***) there is the TaskTimer macro from ***Luke Blanshard.*** (http://labwiki.sourceforge.net)

### TECHNICAL NOTES

Copy & paste, link…

```
data one;
set two;
if mix(var1, var2) > 0 then do;
```

Then as you add information add tags for more general topics then tie those tags together etc. Add a TagCloud and SiteMap to easily find things.

### PRESENTATIONS

The ***SlideShow*** (http://www.math.ist.utl.pt/~psoares/addons.html ) plugin allows a long tiddler to be viewed one piece at a time. The slides are defined by the header level 1 formatting. Ordinary Tiddler formatting can be used within them and all content after a horizontal bar will be hidden from the slide.

### GENERATING VALIDATION REPORTS

Many of us have to prepare validation reports for programs and outputs with lists containing details of the files, comments etc. What if we could make a table based on the actual files present in a directory? With TiddyWiki this is easy because Unix Text Tools can easily be used to generate the markup to make a table in the tiddler text format. Tiddler tables use vertical bars to separate the columns, every line starts and ends with a vertical bar. 'h' at the end of a line formats it as a header line .

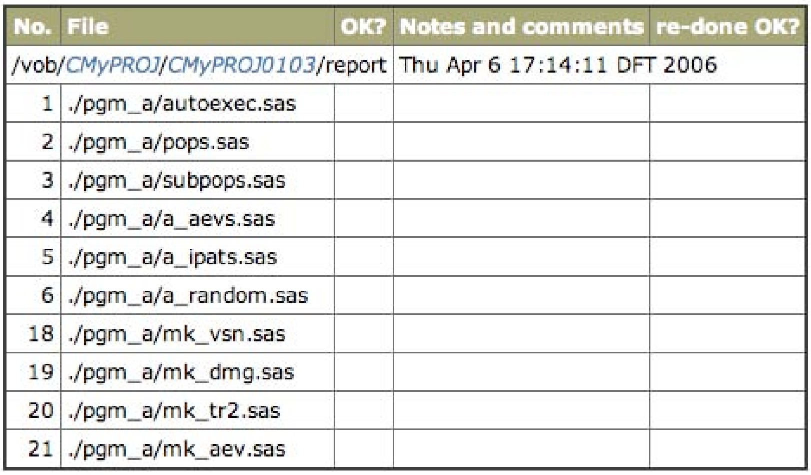
This Unix one-liner delivers a list of files with the parent directory date, and added columns for comments and recording a re-run:

```
    find $PWD -type f  -print | sort |
        awk -v pwd=$PWD -v date="`date `" '
        {if (NR == 1 ) print "|No.|File | OK? | Notes and comments | re-done OK?
 |h\n|>|>|All files in:|" pwd"|"date"|" }
         {sub(pwd,".",$1); print "| "NR"|"$1" | | | |"}'
```

prints plain text on the terminal like this:

```
|No.|File | OK? | Notes and comments | re-done OK? |h
|>|>|/vob/CMyPROJ/CMyPROJ0103/report|>|Thu Apr  6 17:14:11 DFT 2006|
| 1|./pgm_a/autoexec.sas | | | |
| 2|./pgm_a/pops.sas | | | |
| 3|./pgm_a/subpops.sas | | | |
| 4|./pgm_a/a_aevs.sas | | | |
| 5|./pgm_a/a_ipats.sas | | | |
| 6|./pgm_a/a_random.sas | | | |
| 18|./pgm_a/mk_vsn.sas | | | |
| 19|./pgm_a/mk_dmg.sas | | | |
| 20|./pgm_a/mk_tr2.sas | | | |
| 21|./pgm_a/mk_aev.sas | | | |
```

Copy and paste this text into a tiddler and close, to get a rendered table:



| No. | File | OK? | Notes and comments | re-done OK? |
|---|---|---|---|---|
| /vob/CMyPROJ/CMyPROJ0103/report | | | Thu Apr 6 17:14:11 DFT 2006 | |
| 1 | ./pgm_a/autoexec.sas | | | |
| 2 | ./pgm_a/pops.sas | | | |
| 3 | ./pgm_a/subpops.sas | | | |
| 4 | ./pgm_a/a_aevs.sas | | | |
| 5 | ./pgm_a/a_ipats.sas | | | |
| 6 | ./pgm_a/a_random.sas | | | |
| 18 | ./pgm_a/mk_vsn.sas | | | |
| 19 | ./pgm_a/mk_dmg.sas | | | |
| 20 | ./pgm_a/mk_tr2.sas | | | |
| 21 | ./pgm_a/mk_aev.sas | | | |

**Figure 1 Screen shot of a rendered table for validation notes**

### CONSTRUCTING CUSTOM TIDDLYWIKIS FROM TEXT FILES

Some open source Unix scripts are available to do this. They are being used for presenting performance information etc. by scheduling the build of new TiddlyWiki from text files updated by other programs periodically.

### DELIVERING DOCUMENTATION

See my Unix tutorial for some examples of what can be done.

## CONCERNS

*Scalability:*

I certainly thought about 5 years ago that JavaScript was a great idea but hopeless for real work, it was too slow and many of the implementations were buggy and crashed by browser strangeness. But now we have personal computers with 4-5 times the power and memory and more importantly browsers have matured and the implementation differences are not so great anymore. In short, it is now a practicable platform for implementing a document based GUI[5]. Byte for byte a 1.5 MB TiddlyWiki out performs a same sized Word document.

*Privacy*: Unless you install the encryption plugin or keep your TiddlyWiki where no one else can open it, it is not secure.

*Safety of typing investment:*
All your information can be extracted from a TiddlyWiki using a text editor, and it is in any case an open system where you can modify the source code. But the real issue is that the text information is in a unique format so it would require re-formatting. This is indeed the case, but other WikiWikis use very similar commands and could be programmed to support the TiddlyWiki format.

There are backup and autosave options on the TiddlyWiki side bar. Enable them.

The import plugin allows you to easily copy tiddlers from other TiddlyWikis either on the web or as local files. Highly recommended.

*Conversion to other formats* (eg Word)
PDF files can be generated from TiddlyWiki with ease (using Acrobat PDF writer and similar tools) however links are not created. Copy and paste of rendered output looks good and works well into e-mails and Word.

## CONCLUSION

In my opinion for many of the above sample uses it is simply the best tool available now. Try it yourself.

## REFERENCES

***http://www.tiddlywiki.com/*** For other examples see my presentations at PhUSE in TiddlyWiki form at ***http://DaveG.tiddlyspot.com/*** (http://daveg.tiddlyspot.com/#PhUSE-

[5] There some strange similarities between what you can do with TiddlyWiki and what was envisioned for *OpenDoc*.

Talks2006%20%5B%5BGnit%20your%20own%20-%20creative%20Unix%20solutions%20for%20SAS%20programmers%5D%5D)

## ACKNOWLEDGMENTS

I cannot claim any responsibility for TiddlyWiki. Thanks to Jeremy Ruston for creating a ground-breaking tool, and thanks to the community of Tiddlywiki developers for putting in so much creativity and energy in the last year and a half to bring it to where it is now.

## RECOMMENDED READING

First check the links already mentioned, second for training and an introduction try ***here***. Do not miss the google ***group*** and the Plugin repositories, ***here***, and ***here*** among others.

For more recent reading on MicroContent and TiddlyWiki developments see Jeremy Ruston's ***Blog*** at www.osmosoft.com.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

David J Garbutt
BSI AG
Täfernstrasse 16A
Baden-Dättwil / CH – 5405
Work phone +41 56 484 1920
Personal Phone: +41 79 32 68970
Fax: +41 56 484 1930
Email: ***D.Garbutt@bsiag.com***
Web: ***DaveG*** , ***BSI AG***