# Banal: Because Format Checking Is So Trite

Geoffrey M. Voelker
*University of California, San Diego*

**ba·nal** (adj.) lacking originality, freshness, or novelty

## 1 Introduction

This paper is not very interesting. It briefly describes the motivation and implementation of banal,[1] a format checker for PDF documents. Banal deduces the format specification of a document (e.g., font size, margins, etc.) and, optionally, compares the document formatting against a set of rules and reports any violations. Banal is most useful as part of a conference management system for conferences and workshops. To this end, Eddie Kohler has integrated banal into the HotCRP system [3], thereby making banal's services potentially widely available.

In the grand scheme of important issues facing conference organization in our research community, format checking decidedly ranks well close to the bottom. But since banal is becoming more widely used, a brief note on its genesis and implementation seems timely.

## 2 But...Why?

If there actually is an interesting question that banal touches upon, perhaps it is what our community goals are for having formatting requirements in the first place. One perspective is that page limits, much less trifling formatting rules like font sizes and leadings, have a negative overall impact on the ability of researchers to effectively communicate their ideas and results. Towards the other end of the spectrum, another perspective is that page limits and formatting requirements reflect the practicalities of the cost of publishing and the time demands on the community. I'm in the practical camp.

I wrote banal because time is precious. The motivation for banal was born out of the reviewing workload for OSDI 2004. Submissions were 14 pages, and occasionally authors would use a 9-point font to maximize the material included. Review workloads for conferences and workshops in the systems and networking community are substantial; papers with effectively multiple additional pages multiplied across an entire review workload adds up to a noticeable additional time burden.

Another reason could have been fairness. Papers that violate formatting to include additional material could have an advantage over papers that respect the requirements. I have never seen a paper whose outcome depended upon violating formatting, though; papers in violation could have easily fit within the constraints with the same outcome from program committees.

Even though the formatting requirements for submissions are typically quite clear and adamant, program committees are understandably reluctant to penalize even egregious papers. Having a tool that makes it clear to both the authors and the committee how a paper is formatted removes the need to make ad-hoc judgement calls during the paper review process.[2]

A tool like banal can be useful for more than just checking formatting, though. It can aid in other program committee tasks:

*Paper assignments.* Assigning papers to program committee members can be a time-consuming task. Banal could extract author names from the citations in bibliographies and check for overlap with PC members as a basis for initial review assignments. (Suggested by Brian Bershad after OSDI 2004.)

*Preserving reviewer anonymity.* With Acrobat 6, Adobe added JavaScript support for PDF documents. As a result, scripts can be written, for example, to track the IP addresses of machines that open submitted PDF files [1]. Banal could prevent the use of JavaScript in submitted PDF documents. (Kavé Salamatian once encountered a submitted paper that used JavaScript for this purpose.)

*Anonymous submissions.* Conferences with double-blind review require authors to anonymize their submissions. After the submission deadline, PC chairs look through submitted papers to ensure that papers do not have author lists in the title block. Banal could automatically check for the presence of author lists and warn authors during the submission process. (Suggested by Stefan Savage after the SIGCOMM 2008 deadline.)

With time, other uses for a tool that processes submit-

[1] Not entirely coincidentally, another way to pronounce the name of the tool is "be-anal".

[2] At this point in the story, a frequent comment is that it is *just formatting*. I can only agree. But then why do we advertise formatting requirements so uncompromisingly in our calls for papers?

ted papers will likely emerge. For its FastLane system, for example, NSF uses the Adobe LiveCycle PDF Generator Elements product to process documents. FastLane checks the paper size and the top, left, and right margins of uploaded documents, but the emphasis is on generating PDF files from a wide range of input document formats and ensuring that submitted PDF files were generated by reliable tools and are reasonably portable (e.g., have embedded fonts) [2].

## 3 Operation

Banal has three modes of operation. It can report full formatting information for every page of a document. It can print formatting statistics in columnar format on a single line, which is useful for analyzing many documents at once and gathering summary statistics (e.g., the distribution of body font sizes for a set of PDF files). And it can judge a document against a formatting specification provided as an input.

The judging operation is most useful for conference submissions. It is straightforward to convert conference formatting requirements (paper size, font size, page limits, margins, etc.) into a specification. Banal will then report whether a document meets that specification and, if not, how the document fails the specification.

Banal can be used at any stage of the submission process: offline by the program committee during the review process after papers have been submitted; as a requirement of the paper submission process; or as an informative tool that authors can use at any point during the submission process. Since banal uses heuristics and can make mistakes, I would argue that it should not be a requirement for successful submission. Use it simply as the informative tool that it is.

## 4 Implementation

Banal is a perl script that relies upon the pdftohtml tool [4] to determine the location and font size of text in the document. The pdftohtml tool marks characters and words with their position and font, and banal maps them from pdftohtml units to inches and points.

It then applies a series of heuristics. To determine the placement of text on a page, banal essentially tracks the bounding boxes around lines on a page. It then uses the bounding boxes of lines to determine a bounding box for all text on a page. For instance, the right margin of text is the maximum right position of these boxes. Since text can overflow a line, though, banal uses the maximum right position shared by multiple lines — most lines in justified text will have the same right position, and lines with overflowing text will be outliers.

Banal performs similar calculations to determine the left, top, and bottom values for the text region bounding box. For headers, footers, and page numbers, banal uses the width and number of lines at the top or bottom as well as their font size (often headers and footers use a different font from the body font) as hints to identify them.

Once it has determined the text region bounding box, banal can then calculate the margins relative to the paper size being used (e.g., US Letter or A4). Banal uses the bounding box width of full lines to estimate the width of a column. It then uses the page width to estimate the number of columns.

Pages full of graphs and very little text (e.g., just captions) make life difficult for banal's heuristics, and banal is conservative on these pages. Banal tries to identify them according to the number of words on a page. It does a reasonable job with pages that are composed mostly or entirely of tables; e.g., it considers each column of a full-page table as a column of text.

Banal considers the font used most often by lines of text as the body font for a page, and essentially ignores the impact of text in all other fonts. It also tracks the leading between all lines, and similarly uses the most common leading as the leading used in formatting.

Banal processes each page of a document independently. It then merges the results for each page into a summary for the entire document.

## 5 Experience

Banal is starting to see increasing use by conferences in our community. As a stand-alone tool, banal was used in three conferences. Jeff Mogul and Brian Bershad used an initial version of banal for OSDI 2006. Anja Feldmann and Nina Taft used banal in the conference system Anja's group implemented for SIGCOMM 2007. At the suggestion of Stefan Savage, Eddie Kohler integrated banal with HotCRP for use with SIGCOMM 2008.

Henning Schulzrinne has also integrated banal into the EDAS conference management system [5], where it has seen much more extensive use. EDAS invokes banal automatically when people submit manuscript and camera-ready papers, and it both records and displays any problems that arise. Authors can also invoke banal on their papers directly. Henning estimates that, since 2006, over 800 events (conferences, sub-conferences, and affiliated workshops) have used banal [6].

HotCRP for SIGCOMM 2008 instructed authors about banal as follows:

> Only submissions that meet the requirements will be considered. However, since the automated format checker can make mistakes,

> checker errors do not prevent paper submission. If your paper already meets the format requirements, simply submit it as is.

Two submissions were sent to the PC chairs with concerns about the output from banal. Banal is not perfect and benefits from continued experience,[3] but appears to be sufficiently stable for general use.

The source for banal is available as part of the HotCRP project [3]. You can also test banal at the following URL:

`http://www.cs.ucsd.edu/~voelker/banal`

## References


[1] J. Brockmeier. Unexpected features in Acrobat 7. `http://lwn.net/Articles/129729/?SID=8EA81D45601439DA364B1F185795031C`, 2005.

[2] D. Hofherr and G. Strawn. Private communication, 2008.

[3] E. Kohler. HotCRP Conference Management Software. `http://www.cs.ucla.edu/~kohler/hotcrp/`.

[4] G. Ovtcharov, R. Dorsch, M. Kruk, and T. Blair. pdftohtml: A utility which converts PDF files into HTML and XML formats. `http://pdftohtml.sourceforge.net/`.

[5] H. Schulzrinne. EDAS: Editor's Assistant. `http://edas.info/`.

[6] H. Schulzrinne. Private communication, 2008.


[3] For EDAS, Henning has found that converting PDF to PNG and determining the margins in the pixel domain is more reliable than banal's current approach.