

Formatting Computer Science Lectures in HTML — Beyond Microsoft® PowerPoint®

Hugh McGuire
Department of Computer Science
University of California – Santa Barbara
Santa Barbara, CA 93106-5110
<http://www.cs.ucsb.edu/~mcguire/>

Abstract

This paper advocates using HTML for representing Computer Science lectures. This technology should be at least as good as popular presentation-packages such as Microsoft PowerPoint, being able to handle multimedia. Beyond that, HTML can conveniently handle material in chunks larger than a screenful, it can be used in more computer-systems, and it is more extensible.

This paper further presents a package which formats lectures in HTML. Features of this package particularly suited for lectures include: (i) enabling private lecture-notes to be interspersed with the lecture-materials that will be publicly displayed, and (ii) arranging for some texts to be filled in during presentations of lectures, which induces students to be somewhat involved.

Keywords

Electronic presentation, hypermedia markup languages, HTML, teaching, Computer Science education.

1 Advantages of Formatting C.S. Lectures in HTML

One important aspect of teaching is the choice of media used for lectures. And for more than two decades, a trend has been for more and more classrooms to get equipment enabling lecture-materials to be projected from computers [13, 15]. With such equipment, lecturers should be able to more easily and fluidly use wider ranges of different fonts, colors, images, animations, sounds, etc. — general multimedia.

\LaTeX [6] has been a popular package for producing slides for presentations, but it does not support dy-

amic media. Microsoft® PowerPoint® does support advanced media [9], and it is bundled with standard software, so it is now most commonly used for electronic presentations. PowerPoint is designed specifically for presentations, e.g. with large font-sizes and automatic facilities for outlining. It also has Microsoft's standard GUI for editing documents. But how about using HTML Web-pages for presentations, as with [4, 14, 16]?

HTML is equal to PowerPoint in supporting advanced media [17], and it can certainly be displayed in a large font-size. For composing HTML documents, various utilities with nice interfaces are generally available: it's standard for each Web-browser to be bundled with an associated composer for HTML documents, e.g. Microsoft® Internet Explorer® has FrontPage [10] and Netscape Navigator has Netscape Composer [11]; and there are further independent Web-page composers such as Macromedia® Dreamweaver® [8].

Then, beyond being equal to PowerPoint, HTML has advantages as follows:

- HTML can be used in more computer-systems than Microsoft PowerPoint. LINUX/UNIX® systems are fairly numerous, and they generally do not support PowerPoint, but Web-browsers are standard in all general-purpose systems. Using HTML, one can compose a presentation in any computer-system and then present the material in that or any other such system.
- Like the slides-facilities of \LaTeX , PowerPoint generally requires 'chunking' material into single-screen slides [2]. But a computer-program to be presented in a lecture doesn't necessarily fit well into limited slides. By contrast, if a lecture is prepared as a Web-page, then it can contain programs of arbitrary lengths without artificial breaks; as desired, the lecturer can scroll the page to display different sections while retaining the material's coherence. (Some people prepare presentations in HTML but still generally chunk material into single-screen 'slides'. Occasionally, some do take advantage of the possibility for greater lengths, e.g. to present all together an algo-

rithm for making a peanut-butter and jelly sandwich. [4, 14, 16])

- Whereas PowerPoint monopolizes the screen [3], it's easier to integrate separate things with a Web-page. In fact, a hallmark of HTML is the flexibility of its display. For example, the window for a Web-browser displaying lecture-material can be reduced to a sliver, enabling a lecturer to do other things with screen-space such as demonstrate programs, while the lecture-material remains visible and scrollable.
- While any specific circumscribed package for preparing presentations may have many powerful features, one will always be limited to just those features and parameters that have been implemented. But the Web-technology of HTML/XML is already arbitrarily extensible. With HTML, one can use Java Applet programs, which do anything a computer can do. For example, a lecture that is an HTML Web-page could contain an Applet that is a drawing-program, to do arbitrary drawings during the presentation — with some material already drawn, if desired.

The package presented in this paper, `fmt_html`, may be considered as a primitive facility for composing HTML documents; but it has some distinguishing features which make it particularly suited for preparing Computer Science lectures, as follows:

- (i) In addition to materials that are publicly displayed, it is standard for lectures to have associated with them some private notes, which are read aloud by the lecturer but not displayed. With \LaTeX slides, one could have separate notes, and one could also write private lecture-notes as comments in the source materials (from “%” to the end of a line, or via commenting facilities added to \LaTeX). PowerPoint has provided only the capability for notes separate from the content. HTML itself has comments beginning with “`<!--`” and ending with “`-->`”; but working directly with HTML is tedious. The package presented in this paper, `fmt_html`, enables private notes that won't be displayed to be mixed arbitrarily with materials that will be publicly displayed for lectures.
- (ii) It's good for students to be somewhat involved in lectures [1], even if there are also teaching-assistant-led discussion-sessions. If all of the materials of lectures are displayed via old-fashioned writing during the presentations of the lectures (via chalk on chalkboards, markers on whiteboards, markers on transparencies that are projected, etc.), then students generally need to at least copy what the lecturer writes, which is some involvement. With a presentation system, a lecturer can pre-format all the materials. But then during the presentations, students may become

passive, simply watching the shows — even if the lectures contain relatively exciting elements such as animations and sounds.

With HTML, one can create forms with input-fields and text-boxes. Using these, a lecturer can leave some parts of lecture-materials blank, filling them in during presentations of the lectures. This again involves students in copying some materials. Thus, advantages of having materials generally pre-formatted can be retained, while some involvement by students can still occur. A pedagogical consideration is that choosing materials to be filled in during presentations of lectures should highlight them.

Again, working directly with HTML is tedious. But with `fmt_html`, a lecturer can prepare lectures simply indicating what materials will be filled in during the presentations, and `fmt_html` generates HTML with appropriate input-fields and text-boxes.

2 Operation of `fmt_html`

As indicated above, `fmt_html` is basically a program for formatting material in HTML. Figure 1 shows an example of its output, displayed in a Web-browser, with some input-fields filled in. When displaying this material for a lecture, one would set the Web-browser's font-size large. As mentioned above, note how HTML is inherently re-configurable to be presented different ways, e.g. in a thin column for this paper.

For input to `fmt_html`, one types plain text with a few conventions and ‘control’-characters guiding the formatting. The basic features are as follows:

- Running in a command-line environment, `fmt_html` accepts parameters specifying details such as the course-id, e.g. “CMPSC 10” for Figure 1, and the lecture-id, e.g. 17 for Figure 1. Header-material such as the title is derived from these parameters and from the first few lines of the input.
- `fmt_html` does basic WYSIWYG* formatting of whitespace such as line-breaks, indentation of lines, and empty lines. With such material, `fmt_html` generates HTML yielding similar formatting/whitespace in the output. For example, “`
`” is used for a line-break in HTML. But then, one can have a line-break in the input not yield a line-break in the output by having a tilde, “~”, preceding the line-break in the input.
- Outside the preceding situation, a tilde “~” in the input yields a non-breaking space, “` `”; in the output.

*WYSIWYG = “what you see is what you get”.

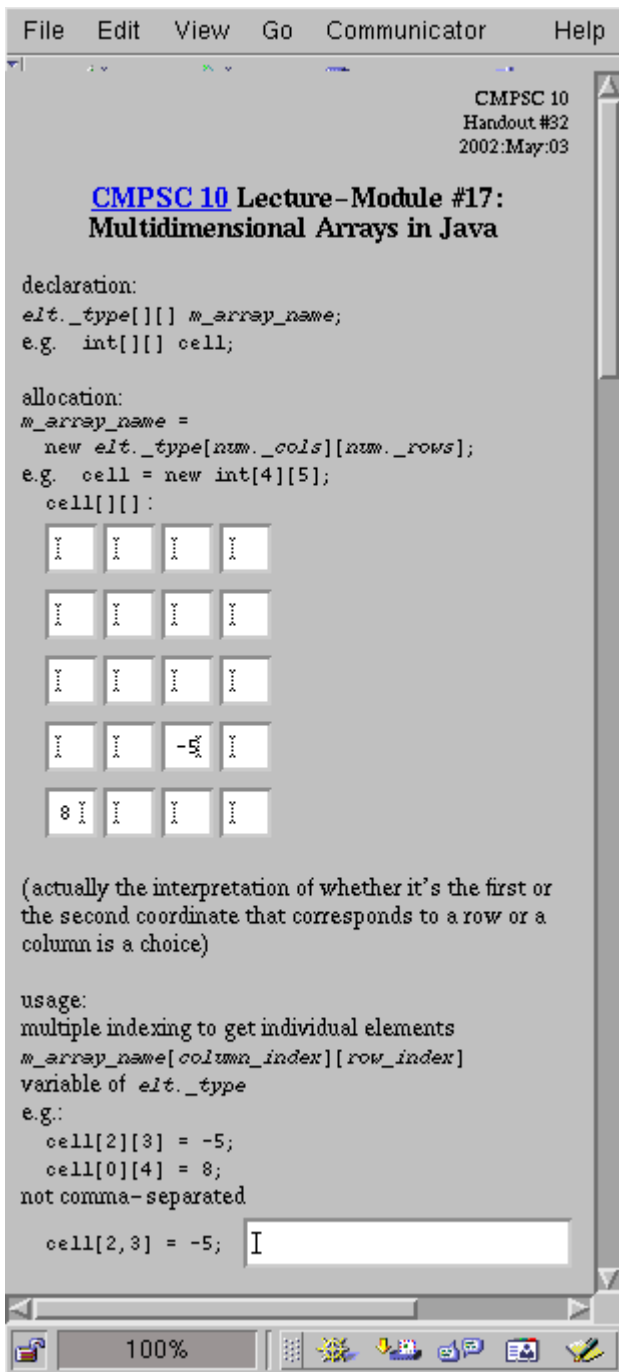


Figure 1: Sample output from `fmt_html`

- If some of the input-material is delimited by bars, “| ... |”, then `fmt_html` generates HTML formatting the material as code, using “`<code>...</code>`”.
- If some of the input-material is delimited by dollar signs, “\$...\$”, then `fmt_html` generates HTML formatting the material in slanted type, using “`<var>...</var>`”.
- If some of the input-material is delimited by carets, “^...^”, then `fmt_html` does not include the material in the output but makes an input-field or text-box sufficiently large for the material. Thus, `fmt_html` makes a blank for such demarcated material to be filled in when the lecture is presented.
- To have one of the above ‘control’-characters appear in the output, the desired character can be preceded with a backslash, “\”.
- If some of the input-material comprises an HTML comment, “`<!-- ... -->`”, then `fmt_html` simply does not output it. With this feature, a lecture can comprise private lecture-notes placed precisely with the publicly displayed materials to which they are relevant. Then, it's convenient to use the document prepared as input to `fmt_html` for lecture-notes.

Here is input-material yielding Figure 1:

```
Multidimensional Arrays in Java
Handout #32
2002:May:03

<!--
PREPARATION BEFORE STARTING LECTURE:
LOAD IDE WITH |life.java|, |life.html|
-->

<!-- SAY:  multidimensional array

           for matrices in Mathematics,
           simple two-dimensional graphics,
           etc.
-->
declaration:
|${elt._type}$[][] $m_array_name$;|
  <!--( this is for two dimensions;
           for more dimensions,
           more sets of brackets
  )-->
e.g. ~ |int[][] cell;|
<!-- as with one-dimensional arrays,
      can be done with brackets following;
      but it's clearer this way
-->
```

allocation:

```
|$m_array_name$ = |
  |new $elt._type$[$num._cols$] [$num._rows$];|
e.g. ~ |cell = new int[4][5];|
|cell[][]| :
^00^ ^00^ ^00^ ^00^
^00^ ^00^ ^00^ ^00^
^00^ ^00^ ^00^ ^00^
^00^ ^00^ ^00^ ^00^
^00^ ^00^ ^00^ ^00^
```

(actually the interpretation of whether ~ it's the first or the second coordinate ~ that corresponds to a row or a column is ~ a <!-- human --> choice)
<!-- in some languages (FORTRAN column-order, C row-order) there may be considerations of efficiency or maybe what graphics packages expect -->

usage:

```
multiple indexing to get individual elements
|$m_array_name$[$column_index$] [$row_index$] |
variable of |$elt._type$|
e.g. :
|cell[2][3] = -5;| <!-- PUT IT THERE -->
|cell[0][4] = 8;| <!-- PUT IT THERE -->
not comma-separated
|cell[2,3] = -5;| ~ ^// compilation fails^
```

Some further features of `fmt_html` are as follows:

- With code delimited by “<pre>...</pre>” or by bars, “|...|”, `fmt_html` ensures that the characters of the code will be displayed properly in HTML by changing “<” to “<”, “&” to “&”, etc. Thus, a lecture can contain real code such as “if (x < y && value ^ ~mask != 0) ...” and it will be displayed as desired.
- A lecture can use input-fields and slanted type in material delimited by “<pre>...</pre>” or bars “|...|” by using control-characters such as ‘Ctrl-^’ instead of “^” etc. Or if desired, one can arrange for the characters “^” and “\$” to have their special interpretations within such material.
- If desired, a lecture can use some raw HTML: `fmt_html` passes through character references and tags such as “½”, “<hr>”, and “”, which yield “1/2”, a horizontal rule, and an unordered list, respectively. Thus, beyond `fmt_html`'s processing, a lecture can take advantage of the full power of HTML/XML.

While everything could be done directly in HTML, it is

tedious [14]. For example, here is some of the HTML from the example above:

```
allocation:
<br>
<code><var>m_array_name</var> = </code>
<br>
&nbsp; &nbsp; &nbsp; <code>new <var>elt._type</var>
      [<var>num._cols</var>]
      [<var>num._rows</var>];</code>
<br>
e.g. &nbsp; &nbsp; <code>cell = new int[4][5];</code>
<br>
&nbsp; &nbsp; &nbsp; <code>cell[][]</code> :
<br>
&nbsp; &nbsp; &nbsp; <input size=2> <input size=2>
      <input size=2> <input size=2>
<br>
&nbsp; &nbsp; &nbsp; <input size=2> <input size=2>
      <input size=2> <input size=2>
<br>
```

It is tedious to type for example “ ” many times. XSL may address this issue somewhat, enabling less typing via one's own definitions. But there's a remaining issue: error-checking. With HTML (or WYSIWYG systems), the standard way to check for errors such as omitting the “/” in “<code>...</code>” is as follows: you view the result, you check if it appears OK, and if it's not then you must locate the error in your source material. It's not clear that people use validation services such as [7, 12] much. By contrast, `fmt_html` notes significant errors and reports their locations in the input.

Another point is that `fmt_html` is a small program written in C. Then since C works on generally all computers, `fmt_html` can be used universally. And then further, since C is a basic standard language, if desired a Computer Scientist can change `fmt_html`, e.g. to use different control-characters.

There are further features of `fmt_html` beyond the ones presented in this paper. For more information, contact the author.

3 Feedback

Formal evaluation of this scheme remains to be done (see Future Work below). But students have submitted comments about it, and this feedback is informative, as follows:

- When asked, some students do say that they dislike PowerPoint. An advantage of HTML mentioned by them which I hadn't thought of is that lectures represented in HTML can be searched for presentation of specific desired material, e.g. “new”.

- A lecturer should avoid overindulgence with features of this scheme. With no restriction to the limitations of slides, a lecture may contain a sea of too much material. If this occurs, a key aspect of lectures — highlighting selected material — is undermined. Additionally, if a lecture has too much material that gets filled in during the presentation of it, then the busy-work of copying what is filled in detracts from students' abilities to really absorb the lecture.
- This scheme makes it very easy to change the contents of lectures. But if one has provided to students a printout of a lecture, then it's best to present the version that was printed rather than a version that one might have made after doing the printout. Students get confused and/or annoyed if a lecture presents material significantly different from a printout of it which they have.

4 Future Work

Thorough evaluation of using HTML and/or `fmt_html` for lectures would involve a formal study teaching different students the same material but different ways (employing `fmt_html`, PowerPoint, etc.), and then measuring how well the students learned with the different systems. Evaluation also needs to come from lecturers comparing different utilities' suitability for composing lectures.

Then, `fmt_html` is still 'beta' or 'alpha'. For one thing, it needs to be tried on more platforms. And some desirable features could be added, e.g. inclusion of a Java Applet program tailored specifically for doing drawings during lectures.

5 Acknowledgements

The utilization of the 'control'-characters “~”, “\$”, and “|” is derived from \LaTeX [6] and \TeX [5]. The utilization of “\” is common.

References

- [1] Boyle, J., Nicol, D., Hamilton, B., and Dempster, B. Experience with classroom feedback systems to enable socratic dialogue in large engineering classes. In *IEE 2nd Annual Symposium on Engineering Education* (London, UK, Jan. 2002).
- [2] Held, J. Powerpoint goes interactive. *Macworld* 15, 9 (Sept. 1998), 107–109.
- [3] Hlynka, D., and Mason, R. “PowerPoint” in the classroom: what is the point? *Educational Technology* 38, 5 (Sept.–Oct. 1998), 45–48.
- [4] Holland, D. A., Lim, A. T., and Seltzer, M. I. A new instructional operating system (talk), 2002. Available [August 28, 2002] WWW: <http://www.eecs.harvard.edu/~dholland/sigcsetalktmp/>.
- [5] Knuth, D. E. *The \TeX book*. Addison-Wesley, 1986.
- [6] Lamport, L. *\LaTeX : A Document Preparation System*, second ed. Addison-Wesley, 1994.
- [7] Lemay, L. Validating html code. *Web Techniques* 2, 4 (April 1997), 16–18.
- [8] Macromedia Inc. Macromedia Dreamweaver MX, 2002. Available [August 28, 2002] WWW: <http://www.macromedia.com/software/dreamweaver/>.
- [9] Microsoft Corp. Microsoft PowerPoint 2002, 2002. Available [August 28, 2002] WWW: <http://www.microsoft.com/catalog/display.asp?subid=22&site=10877&x=22&y=8>.
- [10] Microsoft Corp. Microsoft® FrontPage 2002, 2002. Available [August 28, 2002] WWW: <http://www.microsoft.com/catalog/display.asp?subid=22&site=10875&x=47&y=10>.
- [11] Netscape. Netscape 6.2, 2002. Available WWW [August 28, 2002] <http://wp.netscape.com/browsers/6/index.html?cp=dowpod6>.
- [12] Oskoboiny, G. W3C® HTML validation service, 2001. Available [August 28, 2002] WWW: <http://validator.w3.org/>.
- [13] Presnell, L. K., Ramesh, V., and Browne, G. J. Using information technology to improve learning in higher education: an investigation of multimedia presentations and group support systems. In *Proceedings of the Fifth Americas Conference on Information Systems (AMCIS 1999)* (1999), Assoc. Info. Syst., pp. 37–39.
- [14] Rodger, S. H. Recent talks, 2002. Available [August 28, 2002] WWW: <http://www.cs.duke.edu/~rodger/talks.html>.
- [15] Rollins, S., and Almeroth, K. Deploying an infrastructure for technologically enhanced learning. In *ED MEDIA Conference* (Denver, Colorado, U.S.A., June 2002).
- [16] Ross, R. J. Presentations, 2000. Available WWW [August 28, 2002] <http://www.cs.montana.edu/~ross/personal/presentations.html>.
- [17] World Wide Web Consortium (W3C®). Html 4.01 specification, 1999. Available [August 28, 2002] WWW: <http://www.w3.org/TR/html401/>.