Web Design As a Lead-in To Computer Science For Non-majors

Christopher Vickery
Queens College

CUNY Discipline Council
November 10, 2006
Overview

- The Context and the Problem
- The CIT Minor
- The Web Programming Course
- Teaching Web Programming With “Computer Science Integrity”
The Context and the Problem

- Computer Science Department
- Liberal Arts College
  - CS stresses principles, but includes practical applications.
  - Others stress practical applications, with theoretical foundations secondary.
- Most students want jobs: *skills* not *theory*
  - Number of majors tracks perceived employability.
The CIT Minor

- “IT in a Computer Science Department”
- Courses are largely independent of the major rather than a subset.
  - There is also a CS Minor.
  - (Cross-disciplinary minors are in the works.)
Computer Science Departmental Requirements — CIT Minor and Prerequisite Structure (Curriculum 2005)

Minimum grade for a required CS course: C
Minimum grade for a required non-CS course: C-

- **CS 12 (3 cr)** Understanding & Using Personal Computers
- **Math 122 (4 cr)** Pre-Calculus
- **Math 120 (3 cr)** Discrete Math
  - or
  - **Math 141 (3 cr)** Calculus
  - or
  - **Math 151 (4 cr)** Calculus
- **CS 80 (3 cr)** Problem Solving with Computers
- **CS 111 (3 cr)** Intro to Algorithmic Problem Solving
  - Coreq/Prereq:
  - **Math 120 cr 141 or 151

Four additional CS courses numbered 81-199 chosen from:

- Database Application Programming
- HTML and WWW Programming
- Models of Computation
- Multimedia Fundamentals and Applications
- Science, Computing Tools, and Instrumentation
- Topics in Computing
The Web Programming Course

- Text
  - Holzschlag, *Spring Into HTML and CSS*

- Syllabus Topics *(if only …)*
  - HTML
  - CSS
  - Image preparation
  - JavaScript
    - DOM Manipulation
    - AJAX
  - PHP
  - Server Management
Teaching Web Programming with “Computer Science Integrity”

- Provide Accurate Models
- Use Proper Terminology
- Expose Full Scope of Development Environment
- Clarify Relevance
- Stress Development Best Practices
- Adhere to Standards
Provide Accurate Models

- File system structure and nomenclature
- Sequence of events
  - Browser Request
  - Web Server
  - Server-side scripting / database
  - Server Reply, Browser 2nd requests
  - Client-side scripting
- DOM Tree
- JavaScript Event management
Use Proper Terminology

- File system
- Web site, Configuration Files
- JavaScript
  - Statements, expressions, objects
  - Lists -> Arrays -> Collections
  - Scope of variables
- Rendering Engines
- Namespaces
Exposé Full Scope of Development Environment

- **Server Configuration**
  - DocumentRoot
  - Authentication
  - Development, Testing, Deployment servers.

- **Dreamweaver**
  - Templates
  - Rendering engine
  - Relate to client and server-side scripting
Clarify Relevance

- Why Apache
- Why Dreamweaver
- Why XHTML
- Why CSS
- Why JavaScript
  - Windows Scripting example
  - Photoshop Actions
Stress Development Best Practices

- Separation of Content, Presentation, and Behavior into separate files from the beginning
- Coding Example: Alternate background colors of table rows
  - CSS
  - JavaScript
    - Pros and Cons of alternative approaches
    - What needs to be documented
Adhere to Standards

- Downplay browser-specific issues
  - But acknowledge that even producers of non-standard browsers recognize the importance of standards support.
- Current topics: Tim Berners-Lee on the future of HTML.
- Require validation of all code from Day 1
Conclusion

- Provide a very practical, very how-to course.
- Teach programming best practices rather than cookbook techniques.
- Explicitly state CS principles being brought to bear.
  - Cite CS as the source of models and rationale for practical issues.