Creating a Digital Study Aid

Nicholas Larson

Abstract

In the second year of the Pharmacy program at Ferris State University, it became clear that comparing and contrasting drugs and disease states was very important. Flash cards served adequately for studying the material, but something more dynamic was necessary. A possible solution was to write a webpage using HTML, JavaScript, and CSS to serve as a study aid. The webpage allows for searching for a common property and being able to view all items with that shared property and the rest of their properties to compare and contrast. Manually updating the code to the webpage would often result in syntax errors that would break the webpage until resolved manually. As a solution to minimize syntax errors, a program written in the language AutoHotKey could be used to systematically update the webpage.

Objective

Design a method to compare and contrast drugs and disease states that is quick and easy to use and easy to update.

Languages

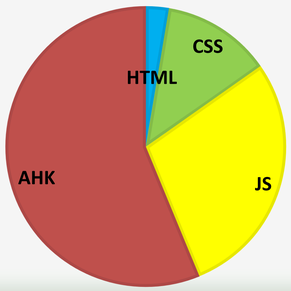
HTML, or HyperText Markup Language, is the underlying language of many websites. It supports text, links, tables, input fields, images, and video. The webpage uses HTML to display the information in tables and images.

JS, or JavaScript, is capable of making webpages dynamic and interactive. The webpage relies on JS to dynamically update what information to display.

CSS, or Cascading Styles Sheet, has its purposes in managing the visual style of a webpage including font, color, and layout. The webpage uses CSS to look as intended on any device and to use different color schemes.

AHK, or AutoHotKey, is a programming language intended for interacting with other applications and producing original applications. The webpage can be updated with an editor created in AHK.

CHARACTERS IN TEMPLATE CODE



Webpage Construction

How should the webpage look and function? Similar to flashcards.
What should the webpage load? One set of flash cards at a time.
How should the data for a flashcard be coded? In JavaScript objects, also known as an associative array.
What features should the webpage have? Displaying single or multiple flashcards simultaneously and the option to search their properties.
How should the logic be written? Relatively little of the webpage will be static. Most of it will be dynamically populated using JavaScript. Learning how to dynamically create variables and use them to insert HTML elements is a necessity.

Power of JavaScript Objects

A JavaScript object is essentially a Key:Value pair. These can be embedded inside of each other: Key:{Key:Value} where the inner {Key:Value} is a Value to the outer Key.

Example line from the source code:

```
"COX-1" : {"Selectivity":"only for Arachadonic Acid",
"Active":"constiuitively","Distinguishing
AA":"isoluceine","Role":"vascular  homeostasis,
intestinal mucosa proliferation, maintain renal/GI
blood flow, renal function, platelet function, ↑
cytoprotective prostaglandins, protect gastric mucosa,
aid platelet aggregation"},
```

In the webpage, COX-1 represents the name of the flashcard. Items in red are attributes and items in blue are values.

Every flashcard is represented by JavaScript objects like the above. The webpage can load a group of these objects, embedded in a parent object which represents the entire flashcard set. JavaScript objects are approximately 2/3rds of all the code.

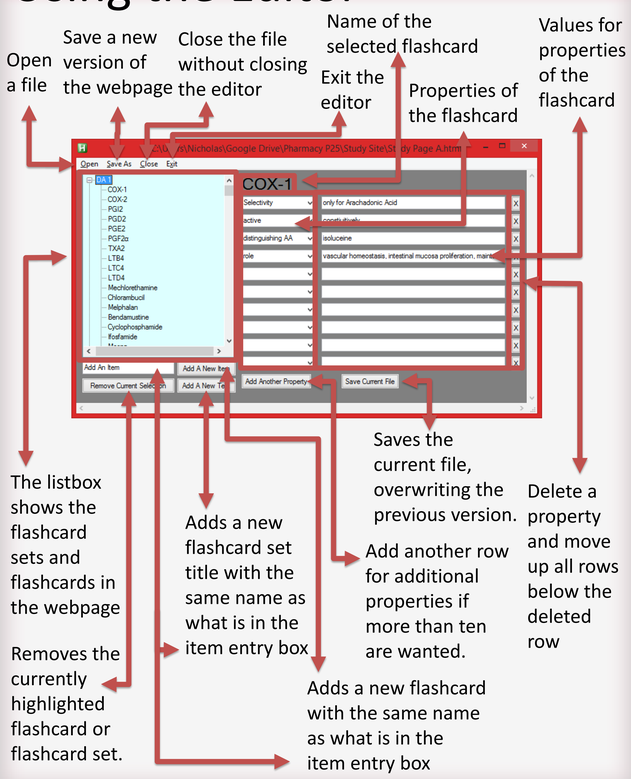
Manual Webpage Updates

Two sections of the webpage must be updated. The first is the list of flashcard sets must be updated so that when a set is selected, the newly added flashcards can be displayed. The second is the flashcards themselves which can be added, removed, or modified. The flashcards must follow a specific syntax. The rest of the webpage includes instructions for loading the webpage and does not need to be changed.

Editor Construction

The creation of an editor was inspired by many syntax errors from manual updates which left the JavsScript dysfunctional and the webpage useless. The goal would be to make a graphical user interface which can update the webpage in the two necessary sections. The editor would only provide changes to the JavaScript objects and the flashcard set list. Dynamically created variables would ensure compatibility as the number of flashcards in the webpage increased. These variables would represent the names of every flashcard set, each flashcard, each attribute, and each value to the attribute.

Using the Editor



Benefits of Separation

This entire project could have been done as either a webpage or a standalone AutoHotKey program. The benefit of a webpage is compatibility across more platforms; AutoHotKey is only for Windows. The benefit of AHK for updating the webpage is ease of use and compatibility with the current webpage code. Having a webpage update itself with the structure of the code would be more difficult because not all of the flashcards are loaded simultaneously like they are in the editor.

Educational Outcome

The primary learned concept was practicing skills for presenting a database in an intuitive user interface to allow for quickly receiving relevant facts.

The project was an exercise in logic. The webpage required learning how to access information in nested JS objects and how to dynamically add HTML elements to a webpage. Most elements required a unique ID to identify the parent element that a child element would be appended to. Those IDs would be dynamically generated as most of the elements being appended were to other dynamically generated elements. The challenge was in managing all of the dynamically created elements. There was also the need to learn how to change the class of an element so new CSS rules could take effect.

The editor required learning about managing dynamically generated variables and how to write the value of those variables to an HTML file. Additional features such as deleting a row in the properties section or deleting a flashcard required reworking code and managing how to handle a gap that was missing in the dynamic variable collection. The addition of a scrollbar to the properties section required borrowing code that was not functional on the computer being used and editing that code to be functional.

If all goes well, this project may be released for others to use for their studies. Removal of bugs and addition of other features are planned.

References

Scrollable Gui - Proof of Concept. Lexikos;; Dec 09, 2008. Retrieved Feb 28, 2015 from http://www.autohotkey.com/board/topic/26033-scrollable-gui-proof-of-concept/#entry168174
AHKScript Documentation. Chris Malett; 2015. Retrieved April 12, 2015 from http://ahkscript.org/docs/AutoHotKey.htm
HTML Element Reference. Refsnes Data; 2015. Retrieved Feb 16, 2015 from http://www.w3schools.com/tags/default.asp
CSS Reference. Refsnes Data; 2015. Retrieved Feb 16, 2015 from http://www.w3schools.com/cssref/default.asp
JavaScript and HTML DOM Reference. Refsnes Data; 2015. Retrieved Feb 16, 2015 from http://www.w3schools.com/jsref/default.asp

Using the Webpage