**COA122 - MiTunes Coursework**

On completion of my script I can say that I was able to complete all tasks assigned. After getting half way through writing the script I thought of a much more efficient way of storing the data from the XML. Originally I was constantly referring to the xmldoc and searching through it in many functions. However my new idea was to put all the XML data into records on load, and then just access these records whenever needed, which I believe was a significant improvement to my script, making it a lot easier to read and write further functions. The script begins by loading in the XML and creating a global variable XMLdoc which is the xml document. I then created four further global variables, allalbums and allsongs which are arrays which a function will later put data into. Also searchlist, which is an empty string which will later be accessed by the search functions and finally albumnumber, which stores the albumnumber currently selected.

**Good Points**

- It only performs the search once. If you click to view an album and then go back it uses the results stored, rather than performing the search again.

- It stores everything in records and so it can very easily access all the information.

-It searches the album/artist first, and if this comes up, it doesn’t search the tracks in the corresponding album as it will be displayed anyway, saving time.

**Bad Points**

- Even though all the information is put into records, saving time, it takes up memory to do this, which is a slight flaw.

- The search is not case sensitive.

- You only can search one string, it is not a multiword search.

**Mitunes.htm Script**

Function makerecords() - This function takes the XMLdoc and searches through it finding album names, artist names and artwork. For every album it makes an array or records and puts this array into the global array allalbums. Eg for album 1, it puts .artistname, .albumtitle and .artwork into an array and puts this is position 0 of the allalbums array and so on. The function does a similar thing for the tracks. For each track it gets the mp3 and title information, puts this in a record, and then into the array tracklisting. It does this for every track in the album. It then puts this array in the global array allsongs. After this function has run I have all the information for everything in global arrays, both of which can be accessed by the same position number to find information on an album.

Function makelist(a,b,c) – This is a function that just takes a name for list (a), an array (b) and a size for a list (c), and produces a html select list from this information. When a list is made it makes, click(a) is called on change and sends the selected index as the variable.

Function writeit(a,b) – I wrote this function to tidy up my script, it simply takes the name of the iframe to be written to (a) and write the content given (b). It also opens and closes the html, head and body tags.

Function makealbumlist() – This functions starts and array by adding All and Search to it. It then finds all the artist names and album names, and for each album adds ‘artist/album’ into the list. It then calls function makelist to turn this into a html list and then write this to the first iframe.

Function clickalbum(b) – This function is called when you click on something on the first list. It takes b, the index of the selected option on the list. If this is 0 it means that All is selected and it calls function makeall. If this is 1 it means search is selected. The function then accesses the global searchlist variable. If this is empty it tells the user they can use the search box to search, if it contains html written by the keyboards search function it simply writes this to the second iframe. If b is above 1 this means an album is selected. The function takes 2 off b, to account for All and search and outputs this as the albumnumber. It then makes an array of all the tracks on this album, using the records made earlier and calls makelist to produce the html and then adds information on the album. This is then displayed all this in the second iframe.

Function makeall() – This is called if All is selected on the first iframe. It creates a html list of all the tracks in all of the albums. For every album it gives it an id of “Albumnumber,Songnumber” and onchange clickall is called with the option id. It then writes this html to the second iframe.

Function clicksong(b) – this is called when a song in an album is clicked. It takes the selected index of the option(b), which is the song number and it accesses the global album number. It uses this information to create html for the third iframe. It displays the song name, album name, artist name, artwork and it lays the song.

Function clickall() – this is called when a song is clicked in the All list. It finds the first and second numbers; it does this by searching by the comma rather than using index of as this can cause errors if two digit numbers are found. It outputs the first number into the albumnumber global variable, and uses the second number as the variable in clicksong, as this is the song number. By doing this the correct information on the song is outputted to the third iframe.

**Keyboard.htm Script**

Firstly I created four global variables. Each one is an array which corresponds to a keyboard layout. There are arrays of letters inside the arrays. Each array inside the main array corresponds to a new line for the keyboard.

Function makekeyboard(a) – This function takes string a, which is the value of the button pressed, and makes the keyboard that corresponds to that value. It does this by using if statements to find which button was pressed and then assigns keytype to the global layout array the button is assigned to. There is then a for loop which writes html for a keyboard. It makes a button for every letter in the array and creates a new line when it reaches the end of each array in the main array. Each button has the value of the letter and has an onlclick which calls clickkey along with variable of its value.

Function clickkey(x) – this is called when a button is pressed. It takes the value of the textbox, and adds x (the value of the button) to it and then outputs this new value back into the textbox.

Function backspace() – this takes the value of the textbox, takes the last letter off it and then outputs this new value back into the text box.

Function wipe() – this outputs a value of nothing into the text box and then calls search() which will in turn make the original album list and there is nothing in the textbox.

Function search() – Firstly the function takes the value of the textbox, as this is the search criteria. It then comes to the first if statement, to find whether the search is empty or not. If it is empty it clears the third iframe and in the second iframe it outputs some html telling the user that they can use the searchbox. It also clears the parent global variable searchlist.

If there is something in the textbox, it starts searching. Firstly it automatically selects the search option on the first list. It then creates two empty arrays and sets two counters to zero. A for loop then initiates. For each album it searches the artist name and album name for the search criteria, if it finds it enters the album number into the list array and moves onto the second album. If it doesn’t find it, this if preventing doubling up songs, it will then search through all the tracks on that album. If the search criteria is found on any track the album number and song number are stored in the list 2 array.

After this for loop it is left with two arrays. List, which is an array of all album numbers which need to be displayed, and list2 which is an array of album numbers and song numbers that need to be displayed. It then creates a html list of all the songs on the albums in list, and all the songs in list2. This is then written to the searchlist parent global variable, and written to the second iframe. In the list the onchange function is clickall with it giving the id as the variable and it works in the same way as the makeall function.