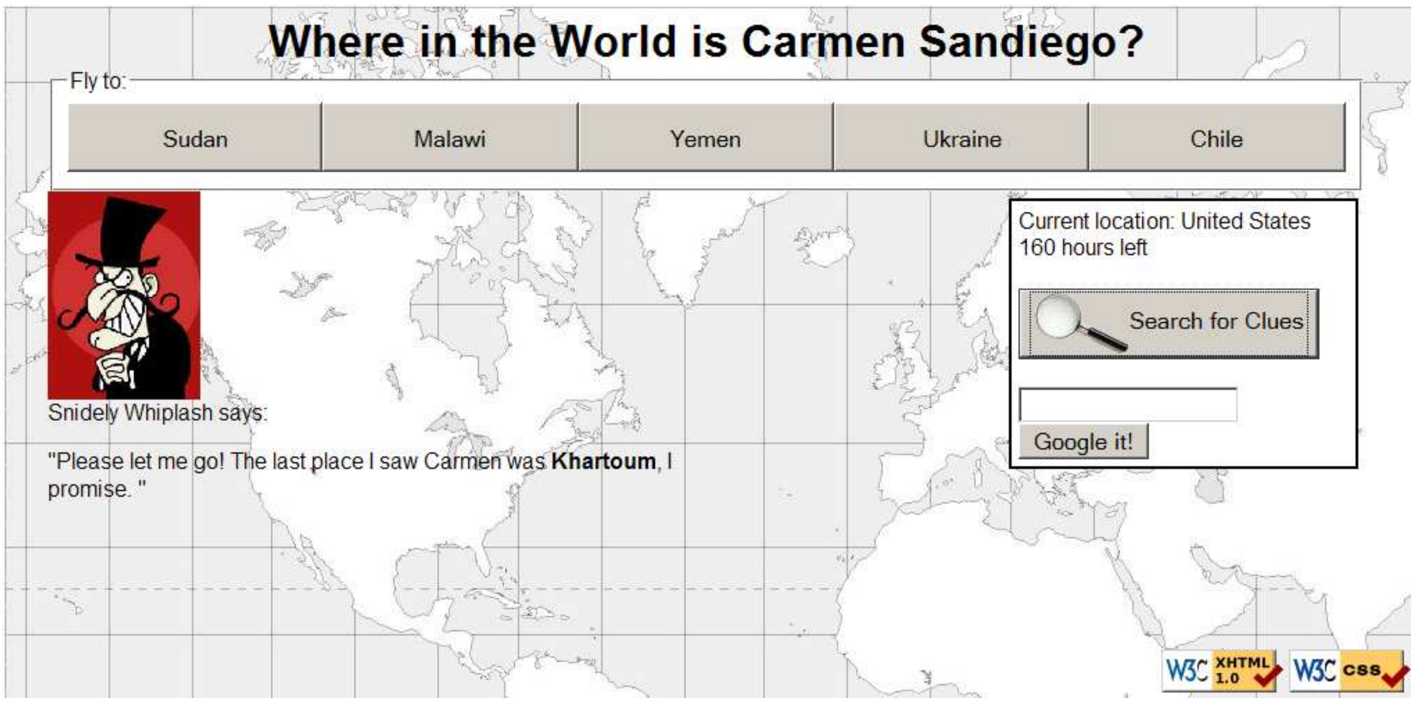
University of Washington, CSE 190 M, Spring 2007
Homework Assignment 6: Where in the World is Carmen Sandiego?

due Tuesday, May 15, 2007, 11:59pm electronically

This assignment asks you to use the Google Ajax Search API and embed multimedia files into a web page. You must match in appearance and behavior the following web page:



Current location: United States
160 hours left

Snidely Whiplash says:

"Please let me go! The last place I saw Carmen was **Khartoum**, I promise. "

Where in the World is Carmen Sandiego? was the title of a successful educational computer game released in 1985. The game challenged the player to control a detective looking for clues to stop the criminal mastermind Carmen Sandiego and her associated goons. Later the brand became a TV game show on PBS.

Your task for this assignment is to write the Javascript code to handle user events for playing a web version of the Carmen Sandiego computer game. You will be provided with a Javascript object that implements most of the internal game logic, but you will add code to present the game's state to the user on the screen, as well as code to let the player search for clues using Google's Ajax Search API.

You do not need to submit any `.html` or `.css` file. We will provide you with HTML (`carmen.html`), CSS (`carmen_style.css`), and Javascript support code (`CarmenGame.js`) code to use. Turn in the following file:

- `carmen_script.js`, the Javascript code for your web page

Part of your grade will come from successfully posting your program solution to your UW web space. To receive this point, we must be able to reach your password-protected page at *exactly* the following URL (case-sensitive):

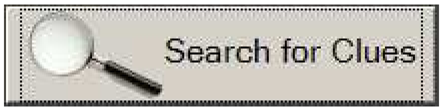
`https://students.washington.edu/`***`your_UWnetID`***`/cse190m/homework/6/carmen.html`

You'll need to upload provided HTML file to the web space with your Javascript file. (You don't need to upload the provided CSS or JS files, because the provided HTML file links to the ones on our server.)

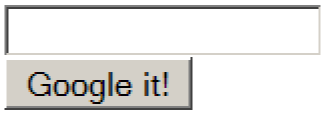
Game Play:

When the page initially loads, the game starts. An audio clip stored in the file `carmen_intro.mp3` plays.

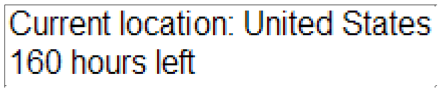
At each stage of the game, 5 buttons containing country names are visible at the top of the window (they are children of the element with ID `flyarea`). The player can click any of these buttons to fly to its respective nation. The goal is to fly to the nations Carmen Sandiego has visited on her crime spree and to eventually catch her.



The player figures out where Carmen has gone by interrogating local villains to get clues. This is done by clicking the **Search for Clues** button (which has ID `cluebutton`). Clicking this button causes a clue to appear in the center of the screen in an area with ID `cluearea`. Each clue gives a hint about Carmen's whereabouts, such as the population of the nation she is in, the national flag, or the currency used in that nation. If the player chooses an incorrect city to visit, the local villains won't know anything about where Carmen is. In such a case, it's important to remember where you came from so that you can retrace your steps and try again to find Carmen.



In the original computer game, the player was expected to have an encyclopedia handy to look up facts about each nation to guide the search. In our version of the game, the player will perform Google searches instead using Google's Ajax Search API. When the user clicks the **Google it!** button (which has ID `googlebutton`), the search results are fetched from the input text box with ID `searchbox` and displayed in the area with ID `searchresults`. The exact format in which these results are displayed is up to you, but you should at least show each result's content in its own paragraph.



Current location: United States
160 hours left

The game has a "time" limit where the player starts with 168 hours (7 days) to catch Carmen. Each game action takes up some time, such as flying to a new nation (8 hours), searching for clues (2 hours), or performing a Google search (1 hour). The time remaining shows on the right side of the page in an area with ID `timeleft`, containing text such as "168 hours left". You must write code to update the text in this area every time the game's state changes. By default the game ends when the player runs out of time or catches Carmen Sandiego. You do not need to support the ability to play multiple games; the user can achieve this by refreshing the page.

The player catches Carmen after correctly following her on 10 hops through the world. When Carmen is caught, you should display a message saying so, as well as some sort of image and multimedia content. For example, an acceptable image would be `carmen_caught.gif`, and an acceptable use of multimedia content would be to play `carmen_theme.mp3` or this video. If you prefer, you can substitute your own multimedia content. If you choose to play a video, you can omit the image. See the `12-multimedia` slides for directions on playing multimedia content through the DOM.

If the player loses the game, you should display at least a simple text message saying this on the screen. If you like, you can display a more elaborate "you lose" message, though this is not required.

The screenshots in this document were taken on Windows XP in Firefox 2.0, which may differ from your system.

Implementation:

Your code will interact with an object of the instructor-provided `CarmenGame` class that models the state of the current game. This object manages much of the game's state and flow, but it does not directly implement the web page at all using the DOM. Your code must connect the CarmenGame to the page appearance on the screen. A `CarmenGame` object has the following constructor, properties, and methods.

Constructor:

`var` *`game`* `= new CarmenGame();`
Constructs a new object representing the current game. When the `CarmenGame` is loaded, it fetches data from some files stored on `https://students.washington.edu/`, so your code must reside on that server to test it.

Properties:

`game``.currentLocation`
A string representing the nation the player currently occupies. This property's value changes each time the player flies to a new city, so you should update the text in the area with ID `currentnation` to match.

`game``.nations[]`
An array of 5 strings representing the possible nations that you can choose to visit next. These nations' names should be placed onscreen as the text of buttons in the area with ID `flyarea`. The user can click these buttons to fly to each corresponding nation. The elements of this array change each time the player flies to a new city, so you should update the onscreen buttons' text to match.

`game``.time`
An integer representing the number of hours remaining to catch Carmen. This property's value changes every time the player flies to a city, requests a clue, or reports a Google search, so you should update the text in the area with ID `timeleft` to match.

Methods:

`game``.fly(`*`nation`*`)`
Instructs the game that the player would like to fly to the nation represented by the given string. Calling this method reduces the time left by 8 hours. If the nation is invalid, the game throws an exception.

`game``.getClue()`
Creates and returns a DOM object representing a clue to show to the player. This clue consists of a villain's image and a clue message from that villain. You can insert this DOM node into your page using the `appendNode` method to display it on the screen. Calling this method reduces the time left by 2 hours.

`game``.isRunning()` , *`game`*`.win()` , *game*.lose()
Each of these methods returns a Boolean value indicating whether the game is in progress, has been won, or has been lost, respectively.

`game``.startGame();`
Instructs the game to start itself and reset its state to initial values.

`game``.reportGoogleSearch();`
Your code should call this method on the game every time the player performs a Google search, so that the game knows to decrease its time left. Calling this method reduces the time left by 1 hour.

Because Javascript does not allow proper encapsulation, all of the above properties can be modified directly by your code; however, you should not do so. Instead, modify the object's state only by calling methods on it, never by directly changing its properties. The `CarmenGame.js` file has some other functions in it that you can use if you like, but you need only the members shown in this document.

Implementation and Grading:

Submit your assignment online from the turnin area of the course web site. You will not need to ZIP it; just turn in the `.js` file. For reference, our solution has 102 lines of Javascript.

Your Javascript code should follow reasonable stylistic guidelines similar to those you would follow on a CSE 14x programming assignment. In particular, you should minimize the number of global variables, utilize parameters and return values properly, correctly utilize the HTML DOM objects, correctly use indentation and spacing, and place a comment header at the top of your Javascript file and atop every function explaining that function's behavior. You should only use material that has been discussed through the lecture on which this program was assigned, unless given explicit permission from the instructor.

Three aspects of your Javascript style deserve particular emphasis. You should minimize redundant code. You should minimize the number of global variables in your program. You should also exercise good procedural decomposition, breaking down lengthy operations into multiple functions, including use of parameters and returns over global variables whenever possible.

Please do not place a solution to this assignment online on a publicly accessible (un-passworded) web site.