Matthew Doles

Professor Casado

CGS3850 Project 3

Documentation of United States Geography Quiz Application

BASIC

CSS

- body
    - width : 850 pixels, decreased width of page body content to a suitable size
    - margin : auto, center page body content
    - padding-top : 20px, flush body content down a smidge for readability
    - background-color : #003c60 (dark blue), adds a little life compared to a dull white

JavasScript

- #tabContainer div
    - .tabs() jQuery UI method to create stylized tabbed panel
- #accordionCtrl div
    - .accordion() jQuery UI method to create collapsible panels for displaying content
        - active : false, no active opened div on page load
        - collapsible : true, makes div's/panels collapsible
        - heightStyle : content, make the height of each div equal to the size of the content on expansion

HOME/CONTACT

HTML

- div - #tabContainer
    - Division for tabbed panels
    - ul, unordered list used to create tabs
        - li/a, Home tab
        - li/a, Contact tab
    - div - #home
        - Division to hold Home tab content
        - h3, welcome text
        - p, site information, purpose, and operations
        - img, relevant image of a map of the United States
    - div - #contact
        - Division to hold Contact tab content
        - 3 p elements, simple contact information

CSS

- .home class
    - HTML elements with applied class include paragraph and image on home tab (everything but h3 header)
    - display : inline-block & vertical-align : top, display paragraph text and image inline and for horizontal alignment
    - img.home CSS
        - border : 3 pixel solid black, customized styling purposes

        - margin-left : 100 pixel, push image left of paragraph text for readability
        - margin-bottom : 15 pixel, adds white space so image is not touching bottom
    - p.home CSS
        - width : 350 pixels, limit width so text does not push image below paragraph element
- No CSS applied for Contact tab

CAPITALS QUIZ

HTML

- h3, header for Capitals Quiz
- div, division to hold content for Capitals Quiz
    - h4, relevant text - ‘Do you know your state capitals?’
    - h4 #questionNumber, displays current question number; dynamically changes in script
    - p #question, displays current question; dynamically changes in script
    - input #userAnswer, captures user input for each question
    - button #next, displays next question on click
    - button #reset, resets quiz on click

JavaScript

- questions [ ], array holding quiz questions
- answers[ ], array holding quiz answers
- index, integer variable used for indexing purposes; instantiated to 0

- correct, integer variable used to track number of correct guesses; instantiated to 0
- $("#questionNumber").text(), display current question number; “Question " + (index + 1) + " out of 5”
- $("#question").text(), display current question; questions[index]
- $("#next").click(function()), handles on click event for the next button
    - userAnswer, variable used to capture the users answer from text input ($('#userAnswer').val())
    - Determine if captured answer is equal to the correct stored value
        - if (userAnswer === answers[index])
        - if the two strings are equal, increment correct (correct++)
    - After handling input, set up page to prompt next question
    - First increment index (index++)
    - Then, delete the text input from the previous question; $('#userAnswer').val("")
    - Determine correct text to display
        - if (index < 5), quiz is not completed
            - ❖ $("#questionNumber").text(), display current question number; “Question " + (index + 1) + " out of 5”
            - ❖ $("#question").text(), display current question; questions[index]
        - else, when index is 5 quiz is completed
            - ❖ $("#questionNumber").text("Quiz Completed!");
            - ❖ Display number of correct responses, $("#question").text("You got " + correct + " out of 5 correct.")
            - ❖ $("#userAnswer").hide(), hide text input; no longer needed

            - $(this).prop(disabled, true), disable next button

- $("#next").click(function()), handles on click event for the reset button
    - Re-instantiate index and number correct back to 0
        - index = 0
        - correct = 0
    - Prompt original question number and question
        - $("#questionNumber").text("Question " + (index + 1) + " out of 5")
        - $("#question").text(questions[index])
    - Re-enable input text and next button if reset was after quiz completion
        - $("#userAnswer").show()
        - $("#next").prop(disabled, false)

DRAG AND DROP STATE MATCHING

HTML

- h3, header for drag and drop state matching
- div, division to hold content for Drag and Drop State Matching
    - h4, relevant text - ‘Do you know your state capitals?’
    - p, user instructions
    - div #gameContainer, holds state images to be organized in order
        - 6 div, each div has a unique id used to determine the correct order
            - 6 images [1], each displaying a unique state shape corresponding to the div’s id
    - table #stateNames, order in which state images are to be organized

        - 6 tr, one for each state shape image
            - 6 td, each tr has one td that holds a unique id; used for styling purposes

CSS

- #gameContainer div
    - display : inline-block, used to show inline with stateNames table
- #stateNames table
    - display : inline-block, used to show inline with gameContainer div
    - border-collapse : collapse, rids of extra white space between table rows
    - #stateNames td CSS
        - border : 3 pixel solid black
        - text-align : center, center text within table cell]
        - height : 200 pixels & width : 200 pixels, matches the height and width of each image
        - padding-top : .25em & padding-bottom : .25em, extra padding needed to get size of each table cell to match the size of each image
        - background-color : #ef5f5f (soft red, default red to bright), changes color when an image is paired correctly

JavaScript

- answersStates [ ], array holding correct order of states; joined together as one long string
- function coordinates(event, ui), needed for sortable function
- $('#gameContainer').sortable(), jQuery UI to enable div/images to be sorted
    - cursor : pointer, pointer cursor displayed when dragging

- axis : y, items can only be dragged vertically
- stop : function(event, ui)
    - Condition to stop sorting when all the correct state images are matched with their corresponding state names
    - First, store order of IDs after an image is drag and dropped joined together as one long string, current order = $(this).sortable("toArray").join(‘’)
    - When currentOrder is equal to the correct order (answerStates); currentOrder == answersStates
        - Then matching is complete and sorting is disabled, $(this).sortable(disable)
- update : function(event, ui)
    - Used to display a table cells background in green when placed in correct position, and in red when incorrect
    - sortedIDs variable, store current array list in temporary variable; $(this).sortable("toArray").join(‘’)
    - Determine if each 3 character substring is equal to the correct; ex. sortedIDs.substring(0, 3) == answerStates[0]
        - If in correct position, change background color to green, $('#tdIND').css('background-color', '#61d85b')
        - When not, change background color to red, $('#tdIND').css('background-color', '#ef5f5f')
    - Repeats for every substring in increments of 3. Would have ideally like to done in a for loop, but hard to change unique CSS td stylings

REGIONS OF THE UNITED STATES

HTML

- h3, header for Regions of the United States
    - div, division to hold content for Regions of the United States
    - h4, relevant text – 'Do you know the regional divisions of the United States?'
    - p #regions, displays either instructions or correct number of guesses; dynamically changes in script
    - 4 input type radio, one for each region of the United States
    - 4 ul, used to display all 50 states in alphabetical order
        - Each list item is a checkbox, which the value of each is the state's abbreviation
        - First 3 ul each contain 15 list items, and the last one the remaining 5
    - button #submit, submits and grades results
    - button #resetRegions, unchecks all selected checkboxes

CSS

- ul
    - display : inline-block, display all 4 ul inline with one another
    - list-style-type : none, rids of bulleted points added by default
    - vertical-align : top, pushes last, shorter, list to the top for horizontal alignment

JavaScript

- northeast [ ], midwest [ ], south [ ], west [ ] - arrays holding correct state abbreviations for each region

- selectedRegion, variable to determine the current selected region; on page load it is Norhteast (selectedRegion = 1)
- $("#regions").text(), text prompting for user instructions
- $("input[type='radio']").click(function()), handle click event when a radio button is clicked (region changed)
    - previousValue, retrieve the previously selected value; $(this).attr(previousValue')
    - Un-check that value and no longer make it the previous value
        - $(this).removeAttr(checked) & $(this).attr(previousValue, false)
    - selected, variable to hold the value of newly selected radio button; $(this).val()
    - Possible selections: Northeast - 1, Midwest - 2, South - 3, West – 4
    - Loop through possibilities to determine selected region, for (var i = 1; i < 5; i++)
        - When selected found, update selectedRegion
        - if (selected == i), then selectedRegion = i
    - Un-check all checkboxes when new region is selected
        - $('input[type=checkbox]').each(function ())
        - For each checkbox, set checked to false, $(this).prop(checked, false)
- $("#submit").click(function()), handle on click event for submit button
    - checked, variable to hold all selected checkbox values
    - correct, variable to hold number of correct guesses; instantiated to 0
    - $('input[type=checkbox]').each(function ()), retrieve the uses input / selected checkboxes
        - if checkbox is checked, append to checked string; separate by comma
            - ❖ if (this.checked), then checked += $(this).val() + ", "

- Determine results/grade based on the currently selected region
- if selectedRegion == 1, Northeast is the selected region
  - Loop through every index of the norhteast array
  - For each iteration, determine if checked string contains the current index value in northeast, and increment correct if match found
    - if checked.search(northeast[i]) != -1, then correct++
    - search() returns the string index if occurrence is found, and returns -1 when no match is found
  - Display the number of correct guesses after for loop completion
  - Determine proper message to display based on results
    - if correct == northeast.length, then $("#regions").text() displays congrats message
    - else $("#regions").text() displays try again message
- This process is repeated, and hardcoded again for each selectedRegion. Again, hard to accomplish through for loop with different arrays.
- $("#resetRegions").click(function()), handles on click event for reset regions button
  - Un-check all checkboxes when new region is selected
    - $('input[type=checkbox]').each(function ())
    - For each checkbox, set checked to false, $(this).prop(checked, false)
  - $("#regions").text(), re-instantiate text prompting user instructions

[1] Images retrieved from: 50States.com, http://www.50states.com/maps/