CSE 154: Web Programming Autumn 2017

Midterm Exam Solutions

Name: ______________________________

UWNet ID: ______________________________@uw.edu

TA (or section): ______________________________

Rules:

- You have 50 minutes to complete this exam.
- You will receive a deduction if you keep working after the instructor calls for papers.
- This test is open-book and open note.
- You may not use any electronic or computing devices, including calculators, cell phones, smartwatches, and music players.
- Unless otherwise indicated, your code will be graded on proper behavior/output, not on style.
- Do not abbreviate code, such as writing ditto marks ("") or dot-dot-dot marks (...). You may not use JavaScript frameworks such as jQuery or Prototype when solving problems.
- You may use JavaScript function aliases like $ or qs **only if** you define them in your code.
- You may use the fetch syntax used in class in any JavaScript programs you write.
- If you enter the room, you must turn in an exam and will not be permitted to leave without doing so.
- You must show your Student ID to a TA or instructor for your submitted exam to be accepted.

Question	Score	Possible
0. HTML		5
1. CSS		5
2. JavaScript		10
3. JavaScript/Ajax		15
4. Short Answer		5
Total		40

0. What's Wrong with my H⊥ML?

```
1  <!DOCTYPE html>
2      <head>
3          <h1>Mowgli's Magical Muffins</h1>
4          <link src=“mypage.css” type=“text/css” rel=“stylesheet” />
5      </head>
6      <body>
7          <p>
8              For Doggies' Best Friends:
9          </p>
10         <ul>
11             <li>Multi-grain Melody</li>
12             <li>Merry-Mint-Chip</li>
13         </ul>
14         For Doggies:
15         <ul>
16             <li>The Malt-ese</li>
17             <li>Malamint Magic<li>
18             <li>Meow Meows</li>
19         </ul>
20     </body>
21 </!DOCTYPE html>
```

This HTML document won't validate, and would generate errors and warnings in the W3C Validator. However, it is possible to make 5 modifications to the HTML to make it pass validation. Each modification might result in multiple text "changes" to the HTML document, but is considered one modification because it is addressing the same root problem. Indicate the 5 modifications we need in order to make it pass validation. Write directly on the HTML.

Below, briefly describe the changes that you made, and why you had to make each change in order to validate. If it is unclear what changes go with which explanations, feel free to number your changes on the HTML. No need for an essay — 10 words or less should be plenty.

Solution:

We accepted 6 possible validation errors:

1. `<h1>` should be `<title>`
2. `link` tag should have `href` as attribute, not `src`
3. "For Doggies:" on line 14 needs to be in a block element (e.g. `p` or `div`)
4. Needs closing `</li>` on line 17
5. Missing `<html>` and `<html>`
6. Illegal `</!DOCTYPE html>` on line 21

1. You Selected the Right Class.

Consider the following HTML:

```
<html>
    <heading>
        <title>CSE 154 Course Web Page</title>
    </heading>
    <body>
        <header id="title-1">
            <h1 id="title-2"><em id="em-1">All the CSE 154 Course Stuffff Ever</em></h1>
        </header>
        <p id="subtitle-1">Topics:</p>
        <ul id="list-1">
            <li id="topic-1">What is the Internet</li>
            <li id="topic-2">How to do the Internet</li>
            <li id="topic-3">How to make the Internet</li>
            <li id="topic-4">Make cool projects:
                <ol id="list-2">
                    <li id="hw-1">Make Pies</li>
                    <li id="hw-2">Watch Lion King</li>
                    <li id="hw-3">Read <em id="em-2">rly rly rly</em> fast</li>
                    <li id="hw-4">Push squares around</li>
                    <li id="hw-5">Catch 'em all!</li>
                </ol>
            </li>
        </ul>
        <div id="div-1">
            <img id="img-1" src="mowgli.jpg">Our course mascot!</img>
        </div>
    </body>
</html>
```

Write the ID's of the elements selected by each of the given selectors:

1. `p` : **`subtitle-1`**
2. `ol li` : **`hw-1, hw-2, hw-3, hw-4, hw-5`**
3. `li em` : **`em-2`**
4. `ul > li` : **`topic-1, topic-2, topic-3, topic-4`**
5. `li li` : **`hw-1, hw-2, hw-3, hw-4, hw-5`**

2. Gotta Make That 📎📎📎

Write a JavaScript program to add to the given HTML and CSS (next page). This program lets a user run their own paperclip factory at the click of a button! Who knew your new web development skills could come in handy?

```
<!DOCTYPE HTML>
<!-- HTML for Problem 2 -->
<html>
    <head>
        <script src="paperclips.js" type="text/javascript"></script>
        <link href="paperclips.css" type="text/css" rel="stylesheet" />
    </head>
    <body>
        <h1>Paperclip-It</h1>
        <p>Paperclips: <span id="count">0</span></p>
        <p>Highscore: <span id="highscore">0</span></p>
        <button id="paperclip-it">Create Paperclip</button>

        <div class="hidden" id="secret-div">
            <hr />
            <button id="lucky-button">Feeling Lucky?</button>
        </div>
    </body>
</html>
```

```
/* CSS for Problem 2 */
body {
  margin: auto auto;
  text-align: center;
  width: 30%;
}

.hidden {
  display: none;
}
```

Solution:

```
"use strict";
(function() {
  let highscore = 0;

  window.onload = function() {
    $("paperclip-it").onclick = makePaperclip;
    $("lucky-button").onclick = feelingLucky;
  };

  function makePaperclip() {
    let count = getPaperclipCount();
    if (count > highscore) {
      highscore = count;
    }
    if (count > 49) {
      $("secret-div").classList.remove("hidden");
    }
    setPaperclipCount(getPaperclipCount() + 1);
    updateHighscore();
  }

  // unlocked at 50 ppclips
  function feelingLucky() {
    let chance = Math.random();
    if (chance < 0.25) {
      setPaperclipCount(getPaperclipCount() * 2);
      updateHighscore();
    } else {
      setPaperclipCount(0);
    }
  }

  function getPaperclipCount() {
    return parseInt($("count").innerText);
  }

  function setPaperclipCount(n) {
    $("count").innerText = n;
  }

  function updateHighscore() {
    let paperclipCount = getPaperclipCount();
    if (paperclipCount > highscore) {
      highscore = paperclipCount;
      $("highscore").innerText = highscore;
    }
  }

  function $(id) { return document.getElementById(id); }
})();
```

3. Fetch ALL the Courses!

Write a JavaScript program `courses.js` which fetches and displays information about current course offerings at UW. This program will be used with the HTML and CSS provided on the following page.

```
<!DOCTYPE HTML>
<!-- HTML for Problem 3 -->
<html>
    <head>
        <script src="courses.js" type="text/javascript"></script>
        <link href="courses.css" type="text/css" rel="stylesheet" />
    </head>
    <body>
        <h1>UW Course Archives</h1>
        <p>
            Enter a UW department code (e.g., "BIOL" or "CSE") and a past academic
            quarter in the format of the first two letters of the quarter
            (e.g., "AU" for Autumn) and the last two digits in the year (e.g., 17 for 2017).
            Then click on the "Find Courses" button and find all courses offered by
            that department for the given quarter!
        </p>
        <fieldset>
            <p>
                Department: <input placeholder="ex: BIOL" id="dept" type="text" />
            </p>
            <p> Quarter/Year <input placeholder="ex: AU17" id="qtr" type="text" /> </p>
            <button id="search">Find Courses!</button>
        </fieldset>
     <h2 id="results-summary"></h2>
     <ul id="course-results">
     </ul>
   </body>
</html>

/* CSS for Problem 3 */
body {
  font-family: Helvetica, Arial, sans-serif;
  margin: auto auto;
  width: 50%;
}

fieldset p {
  text-align: right;
}

h1 {
  text-align: center;
}

input {
  margin-left: 10px;
}
```

```
input, button {
  float: right;
}
```

Solution:

```
 "use strict";
(function() {
  window.onload = function() {
    $("search").onclick = fetchAjax;
  };

  function fetchAjax() {
    let url = "https://uw.edu/courses.php?qtr=" + $("qtr").value +
              "&dept=" + $("dept").value;
    fetch(url)
     .then(checkStatus)
     .then(JSON.parse)
     .then(populateCourses)
     .catch(function(error) {
       $("results-summary").innerHTML = "No results found. Please try again.";
       $("course-results").innerHTML = "";
     });
  }

  function populateCourses(response) {
    $("results-summary").innerHTML = response["dept_name"] + ", " +
                                     response["qtr"] + ":";
    let courses = response["courses"];
    for (let i = 0; i < courses.length; i++) {
      let li = document.createElement("li");
      li.innerHTML = courses[i];
      $("course-results").appendChild(li);
    }
  }

  function $(id) {
    return document.getElementById(id);
  }

  function checkStatus(response) {
    if (response.status >= 200 && response.status < 300 ||
        response.status == 0) {
      return response.text();
    } else {
      return Promise.reject(
        new Error(response.status + ": " + response.statusText));
    }
  }
})();
```

4. Short Answers

1. List 2 inline elements and 2 block elements (do not use `<img>` as any of your answers - it's a special "inline-block" element).

   Solution:

   Possible inline elements: `a`, `img`, `q`, `span`
   Possible block elements: `p`, `div`, `section`, `h1-6`, `blockquote`

---

2. Why do we always want to include an alt attribute on `img` tags?

   Solution:

   Possible answers:

   - Users who cannot see the image due to vision impairment can have a textual description of the image (which can be spoken aloud by a screenreader)
   - If the image fails to load (connection, broken path, etc.), the alt text is displayed instead
   - SEO (Search Engine Optimization) benefits

---

3. What are 2 different JS debugging strategies/tools we have used in lecture and/or section?

   Solution:

   Possible answers:

   - `alert()` or `console.log()` statements
   - Chrome or Firefox Inspector tools, breakpoints
   - HTMLValidator
   - JSLint

---

4. What is the difference between `setInterval` and `setTimeout`?

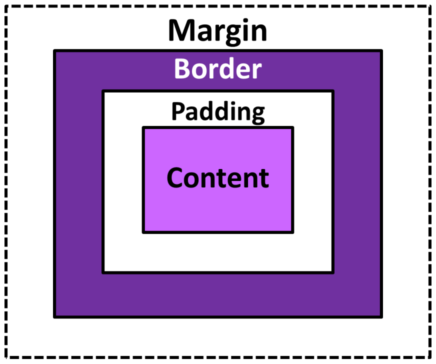
   Solution:

   `setInterval` repeats a function for a a given interval (time), `setTimeout` executes the function only once.

---

5. What's the difference between margin, borders, and padding? (You may provide a labeled diagram)

Solution:


Margin
Border
Padding
Content
