Jerry Cain
CS 106AX

Handout #23S
November 29th, 2023

Section Handout #8 Solutions

Solution Un: Deuxlingo

```
/**
 * Function: Deuxlingo
 * -------------------
 * Defines the controller need to interact with the Deuxlingo
 * web application.
 */
const LANGUAGE_CODE = "fr"; // change to whatever you want
const ENDPOINT_URL =
   "https://web.stanford.edu/class/cs106ax/cgi-bin/translate.py";
function Deuxlingo() {
   let textArea = document.getElementById("textarea");
   let sourceDiv = document.getElementById("source-div");
   let targetDiv = document.getElementById("target-div");
   let editButton = document.getElementById("edit-button");
   let translateButton = document.getElementById("translate-button");

   /*
    * Function: showEditor
    * --------------------
    * Toggles the visibility of the primary HTML elements so that
    * the text area and the translate button are invisible, but
    * the source and target divs are visible (as is the edit button).
    *
    * For fun, I'm using a JavaScript feature that allows you to
    * invoke an array method called forEach, which takes a one-argument
    * function that should be called for each element in the array.
    */
   function showEditor(e) {
      textArea.value = "";
      [textArea, translateButton].forEach(function(elem) {
         elem.classList.remove("invisible");
      });
      [sourceDiv, targetDiv, editButton].forEach(function(elem) {
         elem.classList.add("invisible");
      });
   }

   /**
    * Function: showTranslations
    * -------------------------
    * Provided the textarea has something that can really be translated,
    * showTranslations assembles the relevant URL structure to perform
    * the translation of interest, schedules the success handler to be
    * invokes once the translation comes back, then sends the request.
    */
   function showTranslations(e) {
      let text = textArea.value.trim();
      if (text === "") return;
      let req = AsyncRequest(ENDPOINT_URL);
      req.addParams({ source: text, to: LANGUAGE_CODE });
      req.setSuccessHandler(showTranslationElements);
      req.send();
   }

   /**
    * Function: showTranslationElements
    * -------------------------
    * Handles the server response to show the original and
    * translated texts.  Note that the one argument is of
    * type AsyncResponse, and showTranslationElements is installed
    * as an success handler.
    */
   function showTranslationElements(response) {
      let info = JSON.parse(response.getPayload());
      embedText(sourceDiv, info.source);
      embedText(targetDiv, info.target);
      [textArea, translateButton].forEach(function(elem) {
         elem.classList.add("invisible");
      });
      [sourceDiv, targetDiv, editButton].forEach(function(elem) {
         elem.classList.remove("invisible");
      });
   }

   /**
    * Function: embedText
    * ------------------
    * Clears out the identified div and inserts the supplied text.
    */
   function embedText(div, text) {
      while (div.lastChild != null) div.removeChild(div.lastChild);
      let tn = document.createTextNode(text);
      div.appendChild(tn);
   }

   /* Install the event handlers needed to toggle between two views */
   editButton.addEventListener("click", showEditor);
   translateButton.addEventListener("click", showTranslations);
}

document.addEventListener("DOMContentLoaded", Deuxlingo);
```

Solution Deux: Client-Side JavaScript

```
function testVideoUpload(video) {

   /**
    * Function: onSuccessStatus
    * ------------------------
    * Invoked whenever the server responds with progress
    * report stating how much of a recently uploaded video
    *
    * has been processed.
    * If the video hasn't been fully processed, then another
    * request for a follow-up progress report is scheduled
    * to be called five seconds later.
    */
   let onSuccessStatus = function(response) {
        let info = JSON.parse(response.getPayload());
        console.log(info.id + ": " + info.percent + "% processed.");
        if (info.percent === 100) { return; }
        setTimeout(function() {
            monitorUpload(info.id);
        }, 5000);
    };

   /**
    * Function: monitorUpload
    * ----------------------
    * Issues an async request for a video upload status report.
    * The supplied parameter of the id of the video in question.
    */
    let monitorUpload = function(id) {
        AsyncRequest("api/upload/" + id + "/status")
            .setSuccessHandler(onSuccessStatus)
            .send();
    };

   /**
    * Function: onSuccessUpload
    * ------------------------
    * Invoked once a video upload has been received and
    * post-processing has been initiated.
    */
    let onSuccessUpload = function(response) {
        let info = JSON.parse(response.getPayload());
        console.log("Video (id: " + info.id + ") upload initiated.");
        setTimeout(function() {
            monitorUpload(info.id);
        }, 5000);
    };

    AsyncRequest("api/upload")
        .setMethod("POST")
        .setPayload(video)
        .setSuccessHandler(onSuccessUpload)
        .send();
}
```