**Lab 8. Interaction Design**

What you’ll learn in this lab:

* Interaction design
* Apply appropriate constraints to avoid user errors

**1. Readings**

Suggested reference readings

1. 10 Steps to Interaction Design (IxD)

<https://uxdesign.cc/10-steps-to-interaction-design-ixd-6abe778cb8b8>

1. Principles of Interaction Design: What It Takes to Create Positive Human-Computer Interactions

<https://www.altexsoft.com/blog/uxdesign/principles-of-interaction-design/>

**2. Lab Exercises**

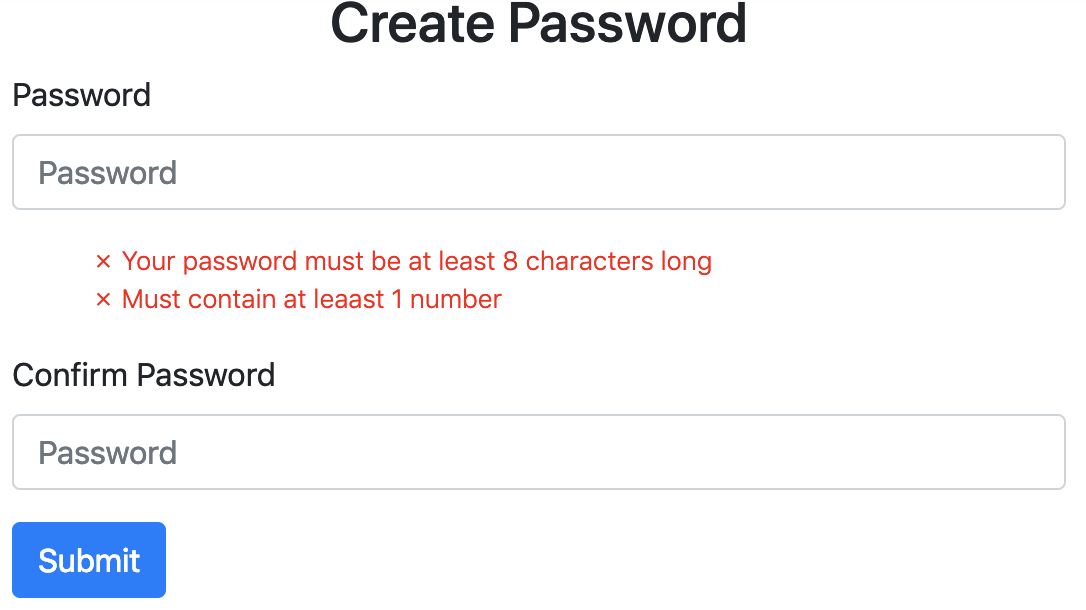
On your own, follow the instructions to finish the exercises. For short text questions, put down your answers in the answer textboxes. For coding exercises, you may edit your code with any text editor. Put your solution code in the solution textboxes when finished.

In the lecture, you have learned different user interface design guidelines. These guidelines are useful for us to design good UIs for different applications. For the lab exercises, we will use the guidelines to evaluate existing UIs and suggest for improvement.

**Exercise 1**

Recall the previous “Create password” lab exercise. Let us assume that another programmer added a “Submit” button to send a request to a server that creates a user account with the entered password.

The new layout of the UI is as follows:



Download **T08Ex01.html** file of the above create password application. Open the file with a web browser and try it out.

You may find that the application allows the user to click on the “Submit” button even if the password does not meet the requirements (or having entered different passwords for confirmation).

|  |
| --- |
| 1. You imagine that the above UI design has violated one of the UI design guidelines as stated in Page 12 of the L04 (UI Design - Guidelines, Principles & Theories (Part 2)) lecture slides. What is it? |
| Your answer here…  5. General guidelines for preventing slips – include helpful constrains  because when the user input the wrong password and input, it will also be submit. |

|  |
| --- |
| 1. To get rid of the design issue in A., you decide to manipulate a countermeasure that avoids user from clicking on the “Submit” button with invalid passwords. What is such a countermeasure? **Hints**: look up the headings of the corresponding guideline. |
| Your answer here…  The countermeasure is that the “Submit” button will not displayed until the user input valid password or different confirmation. |

You will then apply the countermeasure in B. to the above UI. Make a copy of the **T08Ex01.html** file and name it as **T08Ex01b.html**. Open **T08Ex01b.html** file with a text editor.

Firstly, you want to create a function called disableButton()that disables or enables the “Submit” button. The function has a string parameter status that accepts either a “yes” or “no” argument. You may use the following template to implement the function:

function disableButton(status) {

if (status == "yes") {

// statements for disabling the button

} else {

// statements for enabling the button

}

}

To disable a button, we may set the disabled property of the button in the following syntax:

buttonObject.disabled = true|false

, which specifies whether a button should be disabled or not

* true - The button is disabled
* false - Default. The button is not disabled

You may search for the usage examples on the internet in how to disable a button.

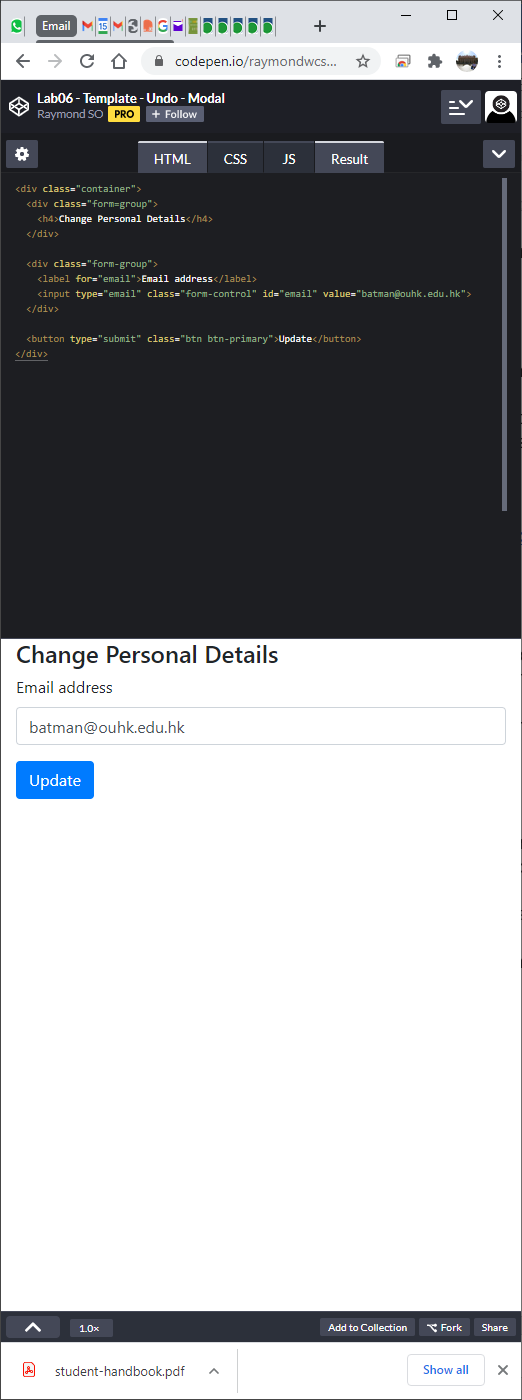
|  |
| --- |
| 1. Complete the disableButton() function and paste it here: |
| Your code here...  function disableButton(status) {  if (status == "yes") {  document.getElementById("btn").disabled = true;  // statements for disabling the button  } else {  // statements for enabling the button  document.getElementById("btn").disabled = false;  }  } |

After completing the function, add appropriate function calls to call it from other functions. **Hints**: adding function calls in the checkSamePassword() functions should be the simplest solution.

|  |
| --- |
| 1. Paste the updated checkSamePassword() function here: |
| Your code here...  function checkSamePassword() {  if (document.getElementById("password1").value === document.getElementById("password2").value) {  showConfirmPasswordHint("no")  disableButton("no");  }  else {  showConfirmPasswordHint("yes")  disableButton("yes");  }  } |

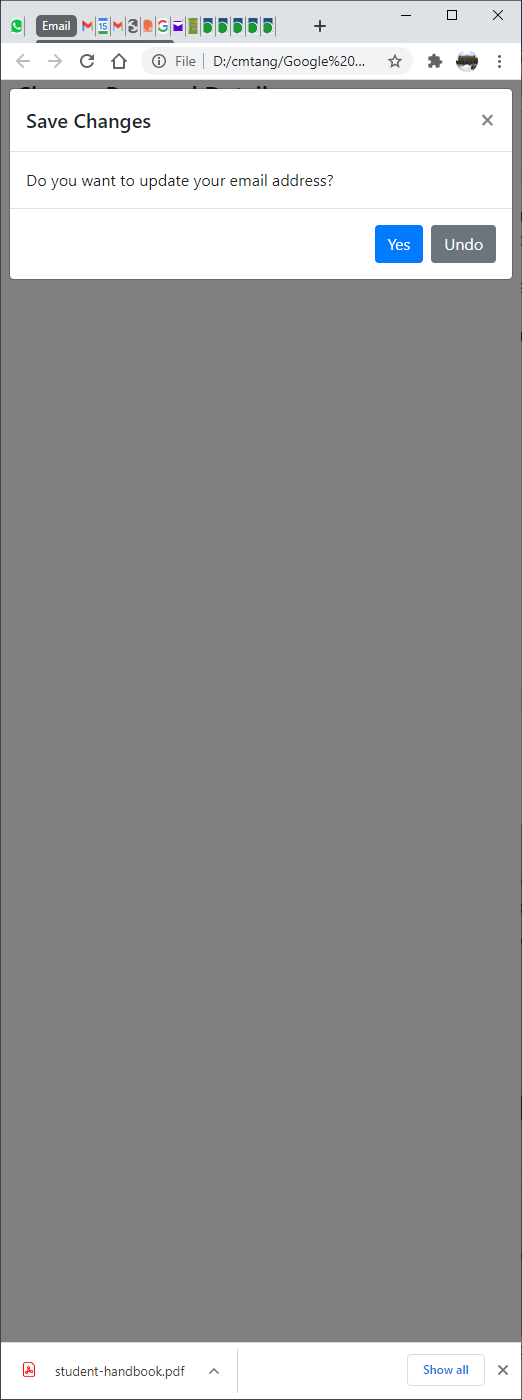
**Exercise 2**

One of the rules of interface design by Shneiderman is to “permit easy reversal of actions”. Shneiderman suggests that actions should be reversible as much as possible. This relieves anxiety, since users know that errors can be undone, and encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data-entry task, or a complete group of actions, such as entry of a name-address block.



The above application allows users update their email addresses. The most obvious action that can be reversed is to undo the changes to the email address. Download **T08Ex02.html** file of the above application. Open it with a web browser and try it out.

A carefully designed Bootstrap *modal* can be used to permit easy reversal of actions. The following is an example modal for reversing the update of the email address:



Make a copy of the **T08Ex02.html** file and name it as **T08Ex02b.html**. Open **T08Ex02b.html** file with a text editor.

The following is the code excerpt for the above modal dialog:

<!-- Modal -->

<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalLabel">Save Changes</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">Do you want to update your email address?</div>

<div class="modal-footer">

<button type="button" class="btn btn-primary" data-dismiss="modal">Yes</button>

<button type="button" class="btn btn-secondary" data-dismiss="modal">Undo</button>

</div>

</div>

</div>

</div>

Integrate it to the <body> section of the **T08Ex02b.html** file.

You also need to link up the “Update” button with the modal dialog so that the dialog can pop up when the button is clicked. Replace the definition of the “Update” button with the follows:

<button type="submit" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal">Update</button>

Open the **T08Ex02b.html** file with a web browser and try it out.

|  |
| --- |
| 1. What happen if the “Update” button is being clicked? |
| Your answer here…  There will appear an dialog box which ask user whether confirmed |

Finally, you want to revert the email address if the “Undo” button is clicked. Add the following statement into the <script> section to retrieve the original email address from the input text field:

let originalEmail = document.getElementById("email").value;

|  |
| --- |
| 1. Implement a function called undoChange() that restore the original email address if the “Undo” button is clicked |
| Your code here...  function undoChange() {  document.getElementById(("email").value = originalEmail;  } |

You also need to add the following event hander to the definition of the “Undo” button:

onclick="undoChange()"

|  |
| --- |
| 1. Can the original email address be restored when clicked on the “Undo” button? (Yes/No) |
| Your answer here…  Yes |

**Submission**

You will get 0 to 3% of OCAS score upon completing the lab exercises.

1. Put your solution code inside the answer boxes and save the file.
2. Upload this file via the corresponding submission link on OLE.

**Note:**

1. The submission deadline is 23:59 of the forthcoming Thursday. The submission link will be closed after the deadline.
2. We mainly focus on the correctness and quality when grading your work.
3. Make sure you submitted a correct file. You can resubmit your work before the deadline.

1. DO NOT modify the layout and format of this document, and DO NOT save it as other file formats, i.e., PDF. Your work will be extracted using an automated document parsing tools for grading.

--- End ---