

Worksheet 12: DOM Events

1. Create the following web page:

DOM Events

Input 1:
Input 2:
Input 3:
Input 4:

2. You need to create DOM Event Handlers in order to handle different events which can occur on the above text fields.

You need to use the **window.onload** event to declare and assign variables to the text field elements, as well as to assign the elements' events to functions:

```
window.onload = function()  
{  
    //declaring and assigning variables to elements  
  
    //assigning elements' events to your defined functions  
}
```

3. These are the events which you need to handle for this page:
 - a) If the user **clicks** on the first text box, an alert should be displayed with the text "Click Event".
 - b) If the user **moves** the mouse over the first text box, text should be displayed in the text box as follows:

DOM Events

Input 1:

Input 2:

Input 3:

Input 4:

Note that the text colour is pink.

- c) Once the user **hovers** over text box 2, text should be displayed in the text box as follows:

DOM Events

Input 1:

Input 2:

Input 3:

Input 4:

Note that this event should be different from the one fired in b.

- d) When the user **moves** the mouse either **out** of text box 1 or text box 2, the text should be cleared. Moreover, the text colour should change back to black.

DOM Events

Input 1:

Input 2:

Input 3:

Input 4:

- e) When the third text box becomes in **focus** (either by clicking on the text box or by pressing the tab key), text should be displayed in the text box as follows:

DOM Events

Input 1:

Input 2:

Input 3:

Input 4:

- f) Once text box 3 **loses focus**, the text "this is focused!" should be removed.
- g) Use the *mousedown* event on the fourth text box so that when this event is fired, the following dialogue box appears:

DOM Events

Input 1:

Input 2:

Input 3:

Input 4:

This page says: ×

Similar to click event!

☐ Prevent this page from creating additional dialogs.

OK

4. Add another label and text field so that the user can input his/her username and password:

DOM Events

Input 1:

Input 2:

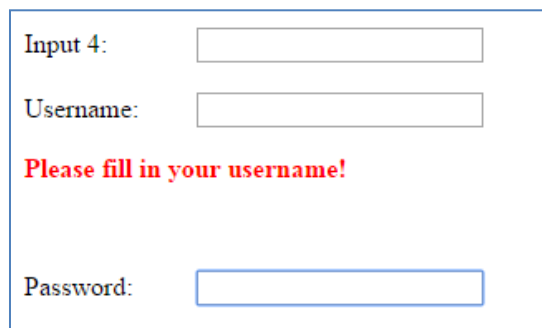
Input 3:

Input 4:

Username:

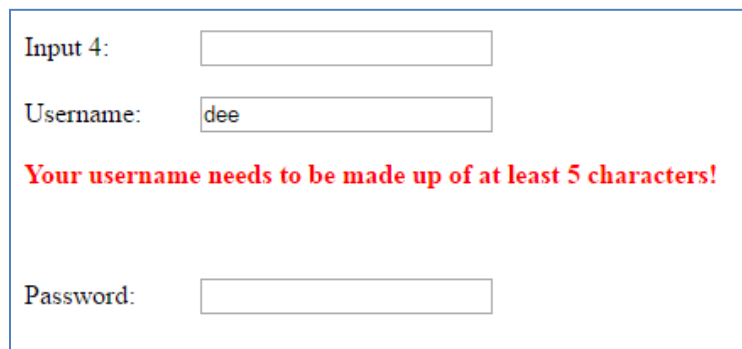
Password:

- a) Once the username text field **loses focus**, an event needs to be fired and a method, `checkUsername()`, needs to be called.
- b) Use DOM Event Handlers in order to perform this procedure.
- c) `checkUsername()` needs to extract the input so as to perform the following checks:
 - i. There needs to be an input. Therefore, if the user has not entered anything yet, an error message should be displayed just beneath the text field:



A screenshot of a web form with three input fields. The first field is labeled 'Input 4:' and is empty. The second field is labeled 'Username:' and is also empty. Below the 'Username:' field, the text 'Please fill in your username!' is displayed in red. The third field is labeled 'Password:' and is empty.

- ii. A valid username needs to have at least 5 characters. If the input has less than 5 characters, an error message needs to be displayed:



A screenshot of the same web form. The 'Input 4:' field is empty. The 'Username:' field now contains the text 'dee'. Below the 'Username:' field, the text 'Your username needs to be made up of at least 5 characters!' is displayed in red. The 'Password:' field remains empty.

NOTE: You **need** to use a paragraph element (**p** tag) in order to display the error message.

Since this will take up space from your web page, you need to ensure that it is only **displayed** once the error has been noted.

d) Similarly to the username text field, once the password text field loses focus, an event needs to be fired and a method, *checkPassword()*, needs to be called.

e) The password text field should not show the user's input:



Password:

f) DOM Event Handlers also need to be used to perform this procedure.

g) *checkPassword()* needs to extract the input so as to perform the following checks:

- i. There needs to be an input. Therefore, if the user has not entered anything yet, an error message should be displayed just beneath the text field:

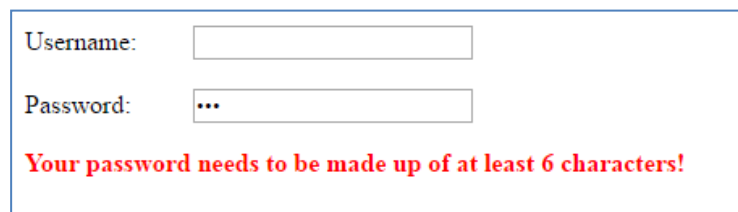


Username:

Password:

Please fill in your password!

- ii. A valid password needs to have at least 6 characters. If the input has less than 6 characters, an error message needs to be displayed:

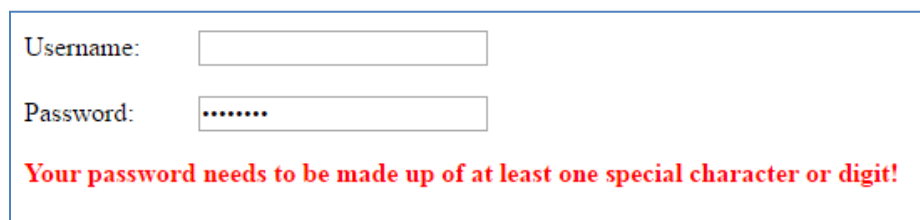


Username:

Password:

Your password needs to be made up of at least 6 characters!

- iii. A check also needs to be done to ensure that the password contains at least one special character or digit:



Username:

Password:

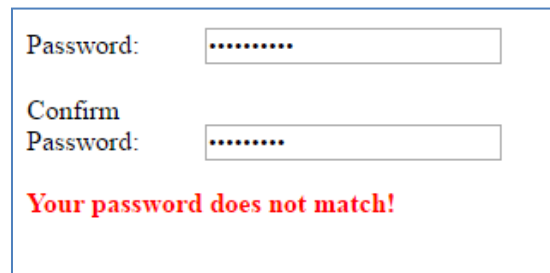
Your password needs to be made up of at least one special character or digit!

NOTE: You need to use **regular expressions** to perform the last check.

Moreover, you need to use the same technique used in c in order to show the error message.

h) Every time the username and password text fields become in focus again, the error message *paragraph* needs to be cleared (and hidden).

5. Include another text field to allow the user to verify the correct password:



The image shows a web form with a blue border. It contains two text input fields. The first field is labeled "Password:" and contains seven dots. The second field is labeled "Confirm Password:" and also contains seven dots. Below the second field, the text "Your password does not match!" is displayed in red.

a) Once the *Confirm Password* text box **loses focus**, an event should fire and call a method `verifyPassword()`.

b) `verifyPassword()` should display the message shown above if the password does not match with the one written in the *Confirm Password* text box.