Title: Alert handling

Steps:

1. Click on the “Alerts (JavaScript)” link in the “Alert” list element.
2. Click on the “Show alert box” button, in the displayed alert click on the “Ok” button.

Expected results:

1. Page opened.
2. Alert displayed and had text ”I am an alert box!”.

Title: Confirm handling

Steps:

1. Click on the “Alerts (JavaScript)” link in the “Alert” list element.
2. Click on the “Show confirm box” button, in the displayed alert click on the “Cancel” button.

Expected results:

1. Page opened.
2. The text “False You clicked Cancel, confirm returned false.” appears under the button.

Title: Prompt handling

Steps:

1. Click on the “Alerts (JavaScript)” link in the “Alert” list element.
2. Click on the “Show prompt box” button, In the displayed prompt type random text and click on the “Ok” button.

Expected results:

1. Page opened.
2. The text “{Your text} You clicked OK. 'prompt' returned {Your text}” appears under the button.

Title: Handling and interaction with fake alerts

Steps:

1. Click on the “Fake Alerts” link in the “Alert” list element.
2. Click on the “Show alert box” button.
3. In the displayed modal dialog click on the “Ok” button.

Expected results:

1. Page opened.
2. Fake alert displayed.
3. Fake alert closed.

Title: Handling and interaction with modal dialog

Steps:

1. Click on the “Fake Alerts” link in the “Alert” list element.
2. Click on the “Show modal dialog” button.
3. In the displayed modal dialog click on background.

Expected results:

1. Page opened.
2. Modal dialog displayed.
3. Modal dialog closed.

Title: Work with ’ iframe’ elements

Steps:

1. Click on the “iFrames test page” link in the “Frames” list element.
2. Inside “iFrame Example List” frame get id of element with text “iFrame List Item 40”.

Expected results:

1. Page opened.
2. Element id equal to “`iframe40`”.

Title: Work with links inside ‘iframe’ elements

Steps:

1. Click on the “iFrames test page” link in the “Frames” list element.
2. Inside the “iFrame Example Header” frame click on “Index” link.

Expected results:

1. Page opened.
2. Main page opened.

Title: JavaScript events triggering

Steps:

1. Click on the “Events (JavaScript)” link inside the “JavaScript” list element.
2. Focus on the “On blur” button, then blur.
3. Click on the “On click” button.
4. Right click on the “OnContextMenu” button.
5. Double click on the “OnDoubleClick” button.
6. Focus on the “OnFocus” button.
7. Focus on the “OnKeyDown” button, key down any key.
8. Focus on the “OnKeyUp” button, key down then key up any button.
9. Focus on the “OnKeyPress” button, press any key.

10. Mouse over on the “OnMouseOver“ button.
11. Mouse over on the “OnMouseLeave” button, then mouse out.
12. Mouse down on the “OnMouseDown” button.

Expected results:

1. Page opened.
2. Under the button appears the text ”Event triggered”.
3. Under the button appears the text ”Event triggered”.
4. Under the button appears the text ”Event triggered”.
5. Under the button appears the text ”Event triggered”.
6. Under the button appears the text ”Event triggered”.
7. Under the button appears the text ”Event triggered”.
8. Under the button appears the text ”Event triggered”.
9. Under the button appears the text ”Event triggered”.
10. Under the button appears the text ”Event triggered”.
11. Under the button appears the text ”Event triggered”.
12. Under the button appears the text ”Event triggered”.

Title: Handling elements inside other elements with ‘hover’ pseudo class

Steps:

1. Click on the “Hover Test Page” link inside the “CSS Pseudo Classes” list element.
2. Trigger hover on the “Hover Para” button.

Expected results:

1. Page opened.
2. Under the button appears a text element.

Title: Click on link inside element which use ‘hover ’ pseudo class

Steps:

1. Click on the “Hover Test Page” link inside the “CSS Pseudo Classes” list element.
2. Trigger hover on the “Hover Div” button.
3. Click on the “Can you click me?” link.

Expected results:

1. Page opened.
2. Under the button appears text and link.
3. New page opened.

Title: Clicking on buttons that appears some time after clicking on another button

Steps:

1. Click on the “Dynamic Button Challenge 01” inside the “Synchronization” list element.
2. Click on the “Start” button.
3. Click on the “One” button.

4. Click on the “Two” button.
5. Click on the “Three” button.

Expected results:

1. Page opened.
2. A new button appears.
3. Under the buttons appears the text “Wait…”, after a few seconds a new button appears.
4. Under the buttons appears the text “Wait…”, after a few seconds a new button appears.
5. Text above the buttons changed from “Click all 4 buttons” to “All buttons clicked”.

Title: Clicking on buttons that become enabled some time after clicking on another button

Steps:

1. Click on the “Dynamic Button Challenge 02” inside the ‘’Synchronization” list element.
2. Click on the “Start” button.
3. Click on the “One” button.
4. Click on the “Two” button.
5. Click on the “Three” button.

Expected results:

1. Page opened.
2. After a few seconds, button “One” became enabled.
3. Under the buttons appears the text “Wait…”, after a few seconds, button “Two” became enabled.
4. Under the buttons appears the text “Wait…”, after a few seconds, button “Three” became enabled.
5. Text above the buttons changed from “Click Buttons in order” to “All buttons clicked”.

Title: Sending client-server form with an HTTP request

Steps:

1. Click on “Client Server Form Input Validation” inside the “Micro apps” list element.
2. Send POST request to ‘https://testpages.herokuapp.com/validate/input-validation’ with form.
3. Open returned page.

Expected results:

1. Page opened.
2. Returned response with body.
3. The page contains the data that was sent in the previous step.

Title: Cookies handling

Steps:

1. Click on “Cookies Controlling Page Access” inside the “Cookies” list element.
2. Add a new cookie with key ‘loggedin’ and value ‘Admin’.
3. Reload page.

Expected results:

1. Page opened. Page doesn’t contain cookies.
2. —
3. Page reloaded, new cookie added, page changed to admin view.

Title: Handling download file

Steps:

1. Click on “File Download“ inside the “Files” list element.
2. Click on the “Direct Link Download” button, wait until the download finishes.

Expected results:

1. Page opened.
2. File downloaded, contains “This is a text file.” text in it and is 142 KB in size.

Title: Interactions with HTML 5 forms

Test Data:

Steps:

1. Click on “HTML5 Element Form Test Page” inside the “Forms & Windows” list element.
2. Fill form using test data.
3. Click on the “Submit” button.

Expected results:

1. Page opened.
2. —
3. Results page contains form data from the previous step.

Title: Sending file

Steps:

1. Click on “File Upload Example Page” inside the “Files” list element.
2. Choose random file in the file input, click on the “Upload button”

Expected results:

1. Page opened.
2. New page opened, under “You uploaded this file:” is the name of file.

Title: Drag and drop elements

Steps:

1. Click on “Drag And Drop Test Page (JavaScript)” inside the “JavaScript” list element.
2. Press any button.
3. Drag the first draggable element to the first droppable element, and do the same actions with second pair elements.

Expected results:

1. Page opened, the first droppable element have text ‘Drop here’, the second droppable element have text ‘No Drop here’.
2. First and second droppable elements have text ‘Drop Here’
3. First and second droppable elements have text ‘Dropped!’

Title: Work with ajax page and select element

Steps:

1. Click on “Ajax JavaScript Examples” inside the “JavaScript” list element.
2. In ‘Category’ select ‘Desktop’.
3. In ‘Language’ select ‘Assembler’.
4. Click on submit button.

Expected results:

1. Page opened. Page doesn’t contain an ‘option’ element with value ‘10’.
2. On the page appeared an ‘ajax busy’ image and ‘option’ element from the previous step.
3. —
4. Opened new page with header ‘Processed Form Details’

Title: Explicit Waitings

Steps:

1. Click on “Multiple Progress Bars” inside the “Synchronization” list element.
2. Wait until the value of the third progress bar 'value' equals '100' without using the implicit expectation.

Expected results:

1. Page opened.
2. Above the bars appeared the text ‘Stopped’.

Title: Changing the size of the window to display the content

Steps:

1. Set window size to 1400x1200
2. Click on “CSS Media Query Based On Size” inside the “CSS” list element.
3. Set window size to 1399x1200
4. Set window size to 999x1200

Expected results:

1. —
2. Page opened and paragraphs with the text '1400 min width', '1000 min width' and others appeared on it.
3. Paragraph with text ‘1400 min width’ is hidden
4. Paragraph with text ‘1000 min width’ is hidden.

Title: Working with several tabs

Steps:

3. Click on “Windows Link Test Page” inside the “Forms & Windows” list element.
4. Click on ‘Alerts In A New Window From JavaScript’ link.
5. Close new tab.

Expected results:

3. Page opened, count of active tabs is 1.
4. Page with header ‘Alert Box Examples’ opened in the new tab, count of active tabs is 2.
5. Tab is closed, the remaining tab contains the same page as the first step