| Sprint 10, Assignment 3.11 Please also update the doc name with correct numbers. | |
| --- | --- |
| Assignment type: JS Interactive | |
| Assignment name  Can remain the same as the assignment topic name, or…  can be phrased in the active/imperative voice (depending on the task), e.g. “Create your own X” | “Add More Event Handlers” |
| Description  The essential tasks a student must perform in order to complete the assignment  ?+ a general, colorful introduction to the assignment? | Now that you've set up your html, and stored useful elements, it's time to add event listeners and handlers that can make the sign-up form more engaging. Make sure every input field   * can change background and text color when selected, * revert to normal when not selected, and * leave an indication that a field has been used at least once   Note: the client really wants to optimize the UX, and has asked you to continue to aggressively-log the user's interaction with the site: add listeners for when the user mouses over and away-from the input fields, too. |
| Link to full code file (Answer)  Push a full code file to our [GitHub repo](https://github.com/bitdegree-foundation/academy-assignments-code-files) for this sprint’s assignment and paste a link to that file here. Please write the file in a concise and clear format, according to the “[example-js-code-file](https://github.com/bitdegree-foundation/academy-assignments-code-files/blob/master/example-js-code-file.js)” found in our repo. | (Upload) |

| # | Step  Write each small step of the task. These assignments should be doable by students on their own (without BitDegree-instructor help). It can be any number of steps, but keep them pretty clear and separate (don’t combine 2 steps into 1 step). | 1x hint  Write brief text or partial code that will help the student figure out how to either move forward or completely accomplish the given step. | Output Expected code that our platform will be able to run & check against student submissions |
| --- | --- | --- | --- |
| 1 | (In your .js file …)  Add event listeners for user's mousing-over input fields, and have them log something like "User considering 'x input' field" | Hint: you have done nearly the same thing for the 'form' element | fName.addEventListener("mouseenter", function(){console.log("User considering first name field...");});  lName.addEventListener("mouseenter", function(){console.log("User considering last name field...");});  address1.addEventListener("mouseenter", function(){console.log("User considering address1 field...");});  address2.addEventListener("mouseenter", function(){console.log("User considering address2 field...");});  userCity.addEventListener("mouseenter", function(){console.log("User considering city field...");});  userState.addEventListener("mouseenter", function(){console.log("User considering state field...");});  userPhone.addEventListener("mouseenter", function(){console.log("User considering phone number field...");}); |
| 2 | Add event listeners for user's mousing-away from input fields, and log something like "User LEFT 'x input' field" | Hint: you covered this at the end of the last assignment, too | fName.addEventListener("mouseleave", function(){console.log("User LEFT first name field...");});  lName.addEventListener("mouseleave", function(){console.log("User LEFT last name field...");});  address1.addEventListener("mouseleave", function(){console.log("User LEFT address1 field...");});  address2.addEventListener("mouseleave", function(){console.log("User LEFT address2 field...");});  userCity.addEventListener("mouseleave", function(){console.log("User LEFT city field...");});  userState.addEventListener("mouseleave", function(){console.log("User LEFT state field...");});  userPhone.addEventListener("mouseleave", function(){console.log("User LEFT phone number field...");}); |
| 3 | Add event listeners for selection of input field selections that 1) log something like "User \*\*SELECTED\*\* "x input" field...", 2) change the styling of the field's text color to white, and 3) the background color to black. | Hint: though you can achieve the above result in multiple ways, try listening specifically for .onclick. Remember - it's written just a bit differently than .addEventListener  Hint: use setAttribute  Hint: change the "style" attribute - specifically its color and background  Note: if you aren't already, now is a great time to practice your editor's shortcuts. What key inputs allow you to duplicate a line of code? Insert a line above that code? Can you select all "console.log" entries at the same time? Can you paste the suggested text simultaneously for all of them? etc etc | fName.onclick = function(){  console.log("User \*\*SELECTED\*\* first name field...");  fName.setAttribute("style", "color:white; background:black");  };  lName.onclick = function(){  console.log("User \*\*SELECTED\*\* last name field...");  lName.setAttribute("style", "color:white; background:black");  };  address1.onclick = function(){  console.log("User \*\*SELECTED\*\* address1 field...");  address1.setAttribute("style", "color:white; background:black");  };  address2.onclick = function(){  console.log("User \*\*SELECTED\*\* address2 field...");  address2.setAttribute("style", "color:white; background:black");  };  userCity.onclick = function(){  console.log("User \*\*SELECTED\*\* city field...");  userCity.setAttribute("style", "color:white; background:black");  };  userState.onclick = function(){  console.log("User \*\*SELECTED\*\* state field...");  userState.setAttribute("style", "color:white; background:black");  };  userPhone.onclick = function(){  console.log("User \*\*SELECTED\*\* phone field...");  userPhone.setAttribute("style", "color:white; background:black");  }; |
| 4 | Add event listeners for the de-selection of the input fields, and have them reset the background color and text to normal | Hint: Use onblur | fName.onblur = function(){  fName.setAttribute("style", "color:black; background:white;)");  };  lName.onblur = function(){  lName.setAttribute("style", "color:black; background:white;");  };  address1.onblur = function(){  address1.setAttribute("style", "color:black; background:white;");  };  address2.onblur = function(){  address2.setAttribute("style", "color:black; background:white;");  };  userCity.onblur = function(){  userCity.setAttribute("style", "color:black; background:white;");  };  userState.onblur = function(){  userState.setAttribute("style", "color:black; background:white;");  };  userPhone.onblur = function(){  userPhone.setAttribute("style", "color:black; background:white;");  }; |
| 5 | Update your onblur listeners to make the borders of the fields also turn a new color once they've been de-selected (effectively showing they've been selected at least once!) | Hint: try looking up css colors and using them for the new border | fName.onblur = function(){  fName.setAttribute("style", "color:black; background:white; border-color: rgb(119, 136, 153)");  };  lName.onblur = function(){  lName.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  };  address1.onblur = function(){  address1.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  };  address2.onblur = function(){  address2.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  };  userCity.onblur = function(){  userCity.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  };  userState.onblur = function(){  userState.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  };  userPhone.onblur = function(){  userPhone.setAttribute("style", "color:black; background:white; border-color:rgb(119, 136, 153)");  }; |
| ... |  |  |  |