Di Lan
Lauren Rabe
Stephen Scheid

Research Lab: Forms and Strings

Question #A.1: Open the simple_tax_form.html file with the browser of your choice by double clicking on it. What is HTML? What is CSS?

HTML stands for HyperText Markup Language. It's the standard markup language for creating webpages and web applications. A markup language is a system for annotating text so that the computer can manipulate the text. CSS stands for Cascading Style Sheets. It's a language used for describing the presentation of a document written in a markup language.

Question #A.2: Now that you have a basic understanding of HTML and CSS, open your HTML file in a text-editor of your choice and examine the code. What is an HTML tag element and what are the two parts? List and describe the functionality for three different types of HTML tags that can be found in your code (do NOT use the HTML tag).

An HTML element is an individual component of an HTML document or webpage. Most elements consist of a start tag and an end tag. A start tag is composed of the name of the element surrounded by angle brackets. An end tag is composed of the name of the element surrounded by angle brackets, with a forward slash after the opening angle bracket. Three types of HTML tags in our code are metadata tags, content sectioning tags, and form tags. Metadata contains information about the webpage, including information about styles, scripts, and data to help software render and use the page Content sectioning tags help organize the document content into logical parts. Sectioning tags are used to create a broad outline for the page content. Form tags can be used together to create forms that the user can fill out and submit to the website or application.

Question #A.3: There are multiple input tags in the HTML form. What are the different input types and why do they differ?

There is the text input type and the number input type. The text input type defines a single-line text field, and the number input type defines a field for entering a number.

Question #A.4: In order to see how several of these input types work, test your code by filling out the form with false input data and providing an invalid number. Submit your form. What error message did you receive?

When an invalid number is entered in the salary field, the error message is "Please enter a number."

Question #A.5: What is an HTML input attribute? List and describe three attributes.

An HTML input attribute is used to define the characteristics of an input element. Attributes are always specified in the start tag. An attribute consists of two parts: a name and a value. The name is the property to be set, and the value is what the property is set to. The value is always put within quotation marks. Three input attributes are the type attribute, the placeholder attribute, and the name attribute. The type attribute specifies the type of input element to display. The default type is text. The placeholder attribute specifies a short hint that describes the expected value of an input field. The hint is displayed in the input field before the user enters a value. The name attribute specifies the name of an input element. The name attribute is used to reference elements in JavaScript or to reference form data after a form is submitted. Only form elements with a name attribute have their values passed when a form is submitted.

Question #A.7 + coding: This form can be submitted with no values entered. What attribute can be included to make each field required? Be sure to implement the attribute within your HTML code for each input field in the form. Save your code and refresh your HTML page. Test your code by trying to submit your form without any input. Did you receive an error message? If so, what error message did you receive? Do you receive this error message for each field? How did you test this?

The required attribute can be included to make each field required. When I tried to submit the form without any input, I received an error message that read "Please fill out this field." I do receive this error message for each field. I tested this by filling one field at a time without deleting any previous input, then pressing the submit button.

Question #A.8 + coding: Your form has additional validation requirements. Your first and last name should be at least 2 characters in length and can only contain uppercase and lowercase letters. Which HTML attributes should you use to implement this? Implement this attribute for the two input fields. Save your code and refresh your page. Test your code by submitting input characters less than what is required and invalid values.

The pattern attribute should be used to implement these validation requirements.

Question #A.9 + coding: More requirements. The salary field can only accept values within the range 0 - 999,999. Which HTML attribute(s) should you use to implement this? Implement this attribute for the input field. 2 Save your code and refresh your page. Test your code.

The min and max attributes should be used to implement this requirement.

Question #A.10 + coding: More requirements. The social security field must be in the format xx-xxx-xxxx and can only include the characters 0-9. Which HTML attribute(s) should you use to implement this? Implement this attribute for the input field. Save your code and refresh your page. Test your code.

The pattern attribute should be used to implement these requirements.

Question #A.11 + coding: More requirements. The Employee Id field must have at least 6 digits and can only include the characters 0-9. Which HTML attribute(s) should you use to implement this? Implement this attribute for the input field. Save your code and refresh your page. Test your code.

The pattern attribute should be used to implement these requirements.

Question #A.12 + coding: More requirements. The Date of Birth field must have be in the format xx/xx/xxxx or x/x/xxxx and can only include the characters 0-9. The year can only start with 19 or 20. How do you do "or" in regex? Which HTML attribute(s) should you use to implement this? Implement this attribute for the input field. Save your code and refresh your page. Test your code.

The expression ($x|y$) allows us to do “or” in regex. The pattern attribute should be used to implement these requirements.

Question #B.1: What is JavaFX?

JavaFX is a software platform for creating and delivering desktop applications and rich Internet applications that can run over a wide variety of devices.

Question #B.2: What is a pane? What is a scene? What is a stage? How do each of these elements relate to each other in a JavaFX form?

Answer: A pane lays out the user interface by setting the position and size properties for each user interface element. The Scene class is the container for all content in a scene graph. The scene graph is the starting point for constructing a JavaFX application. It's a hierarchical tree of nodes that represents all of the visual elements of the application's user interface. A node is a single element in a scene graph. The scene graph can handle input and can be rendered. The Stage class is the top-level JavaFX container. A stage has a scene, and panes can be used within a scene graph.

Question #B.3: Now that you have a better understanding of a JavaFX form, analyze the start method found in the SimpleTaxJavaFX class. Briefly describe what is occurring in lines 26-60.

The code from lines 26 - 60 is part of an overwritten version of the start method, which is the main entry point for all JavaFX applications. These lines create a VBox and a GridPane, both of which are layout panes. The VBox is formatted, the Labels error and title are formatted, and error and title are added to the VBox. The GridPane is formatted, and labels and text input fields are added to it. Also, the submit button is created and added to the GridPane.

Question #B.4: Although your boss does not understand the code, he knows that when you submit the form with empty fields, no validation occurs. Based on this information, you noticed there is an action method (called a handler method) that has an object reference (EventHandler object) as a parameter. What is an EventHandler? Give three examples of events that should be handled as they occur. What should happen when the submit button is pressed? Briefly describe this in a few Sentences.

Answer: An EventHandler is a handler for events of a specific class/type. In JavaFX applications, an event is said to have occurred whenever a user interacts with the application. If an event occurs, event handling is the mechanism that controls the event and decides what should happen. This mechanism has the code that's executed when an event occurs. This code is known as an event handler. Three examples of events that should be handled as they occur are clicking a button, pressing a key, and dragging the mouse. When the submit button is pressed, the input entered in each field of the signup form is assigned to a separate String variable. Also, the text of the error label is set to an empty String. Then, the input should be validated using the String variables. The text of the error label should remain an empty String only if all of the input is valid. If the text of the error label is an empty String, the primary stage is hidden and the results stage is displayed.

Question #B.5: List all fields you need to validate in the form and the corresponding String reference variables.

The fields that need to be validated are first, last, salary, ssn, empId, and dob, and the corresponding String reference variables are fName, lName, inSalary, inSSN, inEmpId, and inDob, respectively.

Question #C.1: As there can be multiple errors (due to multiple fields), you will need to accumulate your error messages. In other words, you should see the error messages for any fields that do not meet requirements. What method do you have available to you that retrieves the current value stored in the error label? What method do you have that allows you to update the value in the error label?

The getText() method is used to retrieve the current value stored in the error label, and the setText() method is used to update the value stored in the error label.

Question #C.2 - coding only: Create a method called checkFirst that takes a String parameter for the first name and does not return anything. If the String parameter for the first name is empty, the method should update the text of the label named error to read "First name required". Otherwise, if the first name has fewer than 2 characters or contains values other than A-Z and a-z, the method should update the error text to "First name invalid". Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.3 - coding only: Create a method called checkLast that takes a String parameter for the last name and does not return anything. If the String parameter for the last name is empty, the method should update the text of the label named error to read "Last name required". Otherwise, if the first name has fewer than 2 characters or contains values other than A-Z and a-z, the method should update the error text to "Last name invalid". Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.4 - coding only: Create a method called checkSalary that takes a String parameter for the salary and does not return anything. If the String parameter for the salary is empty, the method should update the text of the label named error to read "Salary required". Otherwise, if the salary is not in the range 0-999999 the method should update the error text to "Required salary range: 0 - 999999". Note that the salary must contain only digits. Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.5 - coding only: Create a method called checkSSN that takes a String parameter for the social security number and does not return anything. If the String parameter for the ssn is empty, the method should update the text of the label named error to read "SSN required". Otherwise, if the ssn is not in the format xxx-xx-xxx (with only digits) the method should update the error text to "SSN format: xx-xxx-xxx". Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.6 - coding only: Create a method called checkEmpId that takes a String parameter for the employee Id and does not return anything. If the String parameter for the employee Id is empty, the method should update the text of the label named error to read "Employee Id required". Otherwise, if the employee ID is not at least 6 digits (no other characters allowed) the method should update the error text to "Employee ID: Minimum 6 digits". Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.7 - coding only: Create a method called checkDob that takes a String parameter for the date of birth and does not return anything. If the String parameter for the date of birth is empty, the method should update the text of the label named error to read "Date of Birth required". Otherwise, if the date of birth is not in the format xx/xx/xxxx or x/x/xxxx (and using only the digits 0-9), the method should update the error text to "Date of Birth format: xx/xx/xxxx or x/x/xxxx". Note that the year can only start with 19 or 20. Call this method in the handle method after the code that initializes the error label. Compile your code, test and fix any errors.

Question #C.8: Based on what you had to do manually with the Java code, what do you think is happening behind the scenes with the HTML attributes for validation?

We think that HTML has something equivalent to a JavaFX label. Initially, the HTML label-equivalent is empty. The user input (or lack thereof) is compared to the attributes for validation one at a time, in a certain order. If the user input does not match the value of an attribute for validation, then the label-equivalent is updated to contain a certain message based on the name, and sometimes also the value, of that attribute.

Question #D.1: What was the most challenging part of this research lab for your group?

The most challenging part was learning about JavaFX.

Question #D.2: What did your group learn/find the most useful by doing this research lab?

The most useful thing that we learned was how to do “or” in regex.

Question #D.3: Did your group find validation with the HTML web form easier to implement or validation in the JavaFX file easier to implement? Why?

We found that validation with the HTML web form was easier to implement than validation in the JavaFX file because we had to write less code for validation with the HTML web form.

Question #D.4: What was the most fun aspect of doing this research lab?

The most fun aspect of doing this research lab was writing the code.