

COM S/SE 319 : Construction of User Interfaces

Spring 2020

Homework 7

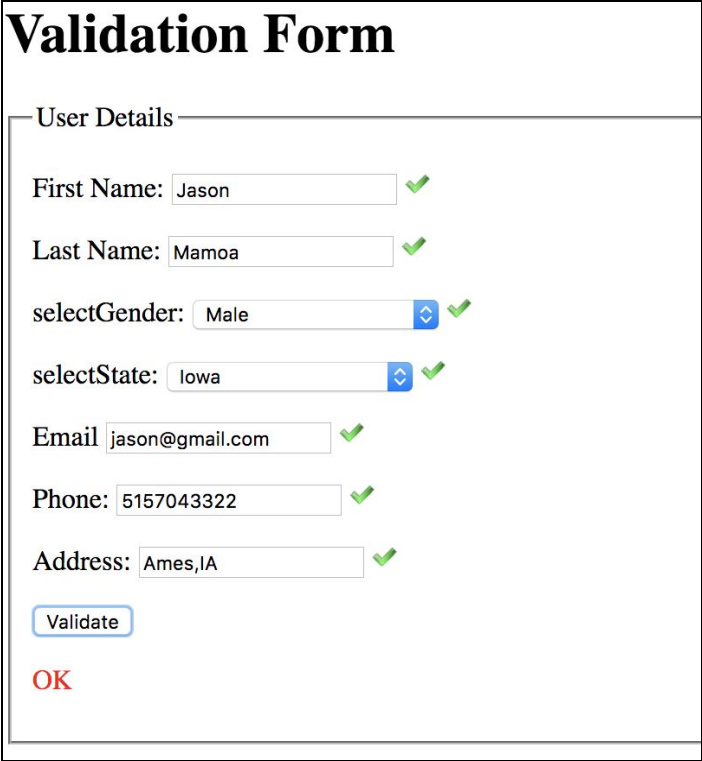
[Total Points: 50]

[Extra Credits - Optional]

Assignment Due: Wednesday, March 25, 2020, 11:59 PM

In this homework, you will do automated testing for an HTML form and an AngularJS application.

Task 1 [25 points]: Validating functions of the **validation.html** in the source code file. You should provide 2 tests that automatically open a **Chrome** browser, one validates the use case that users input correctly and one verifies the use case that users input incorrect. The scenario of 2 use cases are shown in Figure 1 and Figure 2. In JUnit test, you **should make the tests pass**.



The screenshot shows a web form titled "Validation Form". Below the title is a section labeled "User Details". The form contains several input fields, each followed by a green checkmark, indicating successful validation:

- First Name: Jason
- Last Name: Mamoa
- selectGender: Male (dropdown menu)
- selectState: Iowa (dropdown menu)
- Email: jason@gmail.com
- Phone: 5157043322
- Address: Ames, IA

At the bottom of the form, there is a "Validate" button and a red "OK" message.

Figure 1. User inputs correctly.

Validation Form

User Details

First Name: ✓

Last Name: ✓

selectGender: ✓

selectState: ✓

Email: ✓

Phone: ✗ Phone Must be in the form xxx-xxx-xxxx or xxxxxxxxxx. x should be numeric!

Address: ✓

Error

Figure 2. User inputs incorrectly.

Hint:

- To perform the click button, use Selenium to interact with button with ID **“btnValidate”**.
- To check the status of input for users, you can use Selenium to check the value of **“labelNotifytxtFinalResult”** variable. This label has value **“OK”** and **“Error”** with correct and incorrect input.

```
validate >  
  ▼ <p id="FinalResult">  
    .. <label id="labelNotifytxtFinalResult" class="errorMessage">Error</label> == $0  
    </p>
```

Check list:

1. Test for validating correct input [10 pts].
2. Test for validating incorrect input [10 pts].
3. In the report, you should specify a brief description about how you make the tests passed in your implementation [5 pts]

Task 2 [20 points]: Validating the “Year” column’s head and “Manufacturer” of the **viewCars.html**. You will write a program to automatically open viewCars.html and write 2 tests. In the first test, your program should perform the click on “Year” column’s head by one-time click. In the second test, your program should perform the click on the “Manufacturer” select tag (by one-time click) to filter the output to “Toyota” only. The scenario of two tests should be shown like Figure 3 and Figure 4. **The failed conditions of the unit tests is when the code runs to exceptions.**

Before

Manufacturer	Model	Year	Stock	Price	Option
Toyota	Rav4	2008	3	\$8,500.00	Increment
Toyota	Camry	2009	2	\$6,500.00	Increment
Toyota	Tacoma	2016	1	\$22,000.00	Increment
BMW	i3	2012	5	\$12,000.00	Increment
Chevy	Malibu	2015	2	\$10,000.00	Increment
Honda	Accord	2013	1	\$9,000.00	Increment
Hyundai	Elantra	2013	2	\$7,000.00	Increment
Chevy	Cruze	2012	2	\$5,500.00	Increment
Dodge	Charger	2013	2	\$16,000.00	Increment
Ford	Mustang	2009	1	\$8,000.00	Increment

=>

After

Manufacturer	Model	Year	Stock	Price	Option
Toyota	Tacoma	2016	1	\$22,000.00	Increment
Chevy	Malibu	2015	2	\$10,000.00	Increment
Honda	Accord	2013	1	\$9,000.00	Increment
Hyundai	Elantra	2013	2	\$7,000.00	Increment
Dodge	Charger	2013	2	\$16,000.00	Increment
BMW	i3	2012	5	\$12,000.00	Increment
Chevy	Cruze	2012	2	\$5,500.00	Increment
Toyota	Camry	2009	2	\$6,500.00	Increment
Ford	Mustang	2009	1	\$8,000.00	Increment
Toyota	Rav4	2008	3	\$8,500.00	Increment

Figure 3. Test 1: Clicking on “Year” column’s title

Before

Manufacturer	Model	Year	Stock	Price	Option
Toyota	Rav4	2008	3	\$8,500.00	Increment
Toyota	Camry	2009	2	\$6,500.00	Increment
Toyota	Tacoma	2016	1	\$22,000.00	Increment
BMW	i3	2012	5	\$12,000.00	Increment
Chevy	Malibu	2015	2	\$10,000.00	Increment
Honda	Accord	2013	1	\$9,000.00	Increment
Hyundai	Elantra	2013	2	\$7,000.00	Increment
Chevy	Cruze	2012	2	\$5,500.00	Increment
Dodge	Charger	2013	2	\$16,000.00	Increment
Ford	Mustang	2009	1	\$8,000.00	Increment

=>

After

Manufacturer	Model	Year	Stock	Price	Option
Toyota	Rav4	2008	3	\$8,500.00	Increment
Toyota	Camry	2009	2	\$6,500.00	Increment
Toyota	Tacoma	2016	1	\$22,000.00	Increment

Figure 4. Test 2: Clicking on “Manufacturer” brand and set it to “Toyota”

Hint:

- The labels of “Year” column’s head and the “Manufacturer” select are: **"lblYearColumn"** and **"selectManufacturer"**. You can use these ids to interact with the **label** and **select** tag by Selenium.

Check list:

4. Test for validating function of the “Year” column’s head **[8 pts]**.
5. Test for validating filtering to “Toyota” manufacturer by the “Manufacturer” button **[8 pts]**.
6. In the report, you should specify a brief description about how you make the tests pass in your implementation [4 pts]

You have to implement this task using **Html, Javascript, Selenium and JUnit Testing**. We will use **Chrome** web browser to test your solution

Check list:

Please find the attached HW 7 zip files **HW7-Files.zip** on Canvas.

Submit requirement [5 points]

Submit via Canvas a **compressed file (.zip)** [rename it with your LAST NAME] containing the following folders and files:

- **src**: Attaching your project (which includes the source code) of your implementation
- **README** file explaining how to compile and run your program
- **Report** (.docx or .pdf) file describing your solution approach and **screenshots** of every required output.