CS385 Mobile Application Development 2023/2024

Lab Session #2 - 13th October 2023

There are engagement continuous assessment marks linked to this assignment. You can complete this assignment outside of lab times. Please refer to Lecture 1 for full details.

You must upload your solutions to Moodle (The link is on the Moodle Section for this Lab) by 15:59 Tuesday 17th October 2023. Remember, you can upload partial solutions. You do not need to upload fully working solutions. A video of how to upload your solutions is available on Microsoft Stream (link) This is also available on Moodle.

Please name your files as .txt files when you are uploading to Moodle.

Lab Description

Please refer to CS385 Lecture 5 materials for guidance. It is strongly suggested that you use this material as guidance rather than using Google or StackOverflow.

You should use the React/Javascript TEMPLATE as provided in `Lab2_Template.txt` from the Moodle page. This will be copied directly into **codesandbox.io** – when you have successfully copy-pasted this into codesandbox you are ready to start working.

`Lab2_Template` shows an **App.js** file with a parent component (App) and two child components called **Robert** and **Jennifer**. Both child components are defined correctly and at the moment simply render some HTML.

TASK 1 – using props to faciliate parent child communication.

Write code into the template `Lab2_Template.txt` Mobile Application for the following:

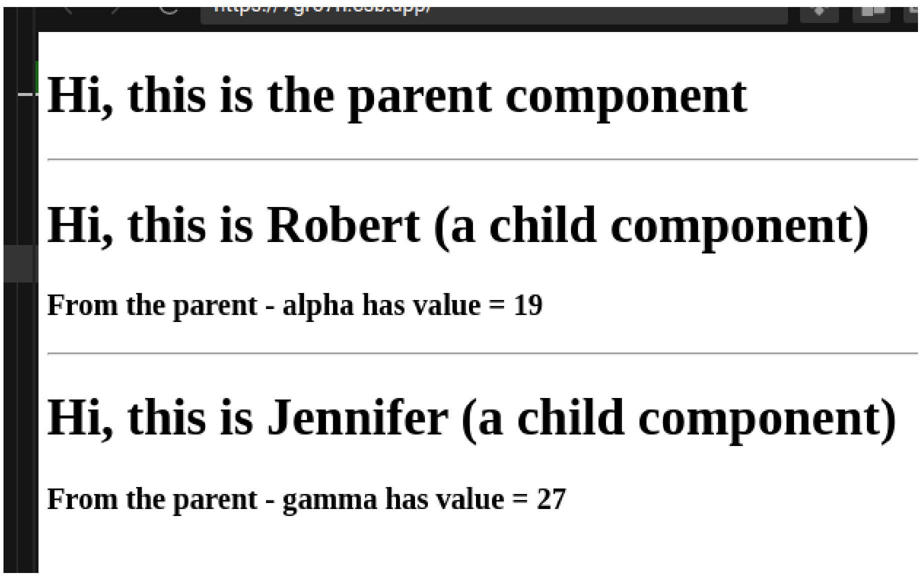
Declare two state variables (using **useState**) in the parent component – the variables should be called **alpha** and **gamma**. Both variables should have an appropriate set function. Both variables have an initial state value of 0.

Using **props**, communicate the current variable value of **alpha** to the child component called Robert. Using **props**, communicate the current variable value of **gamma** to the child component called Jennifer.

All you need to do at this point is to sucessfully render the values of **alpha** and **gamma** within the child components as described above.

Change the initial values of alpha and gamma in the parent and see that the child components successfully render the new values.

Your solution to task 1 should create something similiar to the image below. Whilst it is not required, you can add formatting and styling if you wish.



Hi, this is the parent component

Hi, this is Robert (a child component)

From the parent - alpha has value = 19

Hi, this is Jennifer (a child component)

From the parent - gamma has value = 27

Save your code – this can be uploaded to Moodle.

This code will be needed for task 2.

TASK 2 – Using button events to change state variables in the parent component

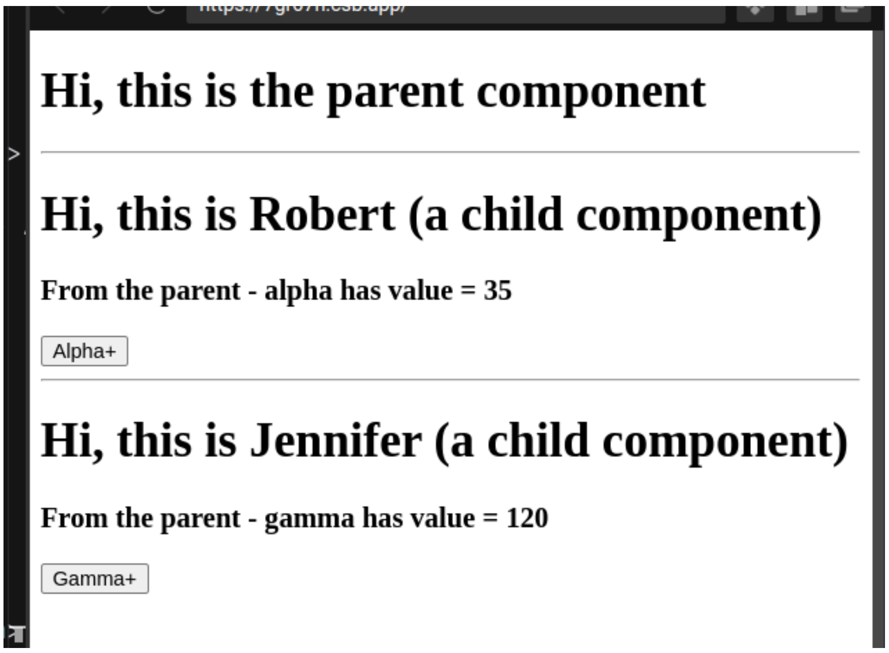
You can use your code from Task 1 as the starting point for this task. We need to complete the following steps.

- We are going to place a button within the Child component **Robert** which increments the value of **alpha** by a variable amount **i.**
- We are going to place a button within the Child component **Jennifer** which increments the value of **gamma** by a variable amount **i.**
- **Step 1** – create two event handler functions called **changeGamma** and **changeAlpha** to allow changes to the state variables **gamma** and **alpha**. Please refer to Slide 21 Lecture 5 slides for an example (`changeMyVar()`) and Slide 30 Lecture 5 (`changeMyVar()`) - you will need to use the two set functions created in task 1.
- **Step 2** – using **props**, pass a handler to your **changeGamma** and **changeAlpha** functions to each of the Child components. Again, you are referred to Lecture 5 – Slide 30.
- **Step 3** – within each component – create a button with an `onClick` event – you will need to connect this button to the **props** for that component. See Lecture 5 – Slide 30
- **Step 4** – test your application. Configure your application so that the child component **Robert** changes the state variable **alpha** by +5 on each mouse click of

the button in the component. Configure your application so that the child component **Jennifer** changes the state variable **gamma** by +10 on each mouse click of the button in the component.

- **Step 5** – save and upload your code. This is the end of the mandatory lab.

Your application should look something like the screenshot below (after several mouse clicks of both buttons)



Hi, this is the parent component

Hi, this is Robert (a child component)

From the parent - alpha has value = 35

Hi, this is Jennifer (a child component)

From the parent - gamma has value = 120

TASK 3 – OPTIONAL – DO NOT UPLOAD THIS TASK TO MOODLE

If you want to try out some additional parent-child communication code – you can attempt the following example – **PLEASE DO NOT UPLOAD TASK 3 TO MOODLE.** This is optional and is not part of your Lab 2 assessment.

TASK – Change your code from Task 2 as follows:

- Change the **Robert** component so that it is given an extra button which decrements (or reduces) the value of **gamma** by a variable amount.
- Change the **Jennifer** component so that it is given an extra button which decrements (or reduces) the value of **alpha** by a variable amount.

Test your code for different variable values.

UPLOADING YOUR SOLUTIONS

- You can upload separate files for each task or a single file into Moodle. It is strongly advised that you simply copy your code from CodeSandBox.io into a notepad (or otherwise) text editor. Save your files as txt files. This is the simplest way to do this. DO NOT USE WORD DOCUMENTS. Programmers do not copy-paste code into MS Word Documents.