DOCUMENTATION OF DISTRIBUTED SYSTEMS PROJECT 2013

I have implemented the whole project of Distributed Systems in Javascript and PHP. I have used Javascript for client side and have used PHP for server side coding. My project runs both in Firefox web browser and in Chromium web browser. My project is best viewed in Firefox web browser.

1. Simple calculator

- In this step, the source code files contains calculator1.html and calc.php
- There are 2 input fields and 1 selector of operations and 1 submit button.
- Compilation – The operator can perform only "+,-,*, /" operations. The input fields can take into input any numbers like for example:-
  1 + 2 or
  -0.1 * 9 or
  3 / -9.8 or
  12.1 - 13.09
  And on clicking the submit button, it will show the correct result.
- Source code explanation –
  1. At first in the client side i.e. in calculator1.html file, through the form the user inputs 2 numbers, one in each field and selects the required operator. On clicking the submit button, ajax function is called which is here named as ajaxFunction.
  2. The ajax function catches the inputs of the user and sends it to the calc.php file in server side.
  3. In server side (i.e. in calc.php file), on receiving the input, it matches the operator and performs the required operation.
  4. After performing the operation, server side i.e. calc.php sends back the result to client side.
  5. On receiving the reply from the server side, calculator1.html i.e. client side displays the result, along with the past history (if applicable).

2. Step 1 – Client side

- In this step, the source code files contains calculator2.html and calc.php
- There are 1 input field and 1 submit button.
- Compilation –The input field can take into input the numbers and operators together like for example:-
  1+2 *3/4 or
  -0.1*9--2+0.9 or
  3/-9.8-1+0.2*-8
  And on clicking the submit button, it will show the correct result. It takes negative numbers as valid input.
- Source code explanation –

1. At first in the client side i.e. in calculator2.html file, through the form the user inputs numbers and operators together as mentioned above. On clicking the submit button, cal1 function is called.
2. The cal1 function parses the given input and takes the first 2 numbers and 1 operator with the help of regular expression.
3. After that, ajax function is called which is here named as ajaxFunction.
4. The ajax function catches the inputs of the user and sends it to the calc.php file in server side.
5. In server side (i.e. in calc.php file), on receiving the input, it matches the operator and performs the required operation.
6. After performing the operation, server side i.e. calc.php sends back the result to client side.
7. On receiving the reply from the server side, calculator2.html i.e. client side displays the result, along with the past history (if applicable) and also replaces arg1, arg2 and op with the value returned from server.
8. Again, the steps from 2 is repeated until no operator is found in the input expression.

3. Step 2 – Client side

- In this step, the source code files contains calculator3.html and calc.php
- There are 1 input field and 1 submit button.
- Compilation –The input field can take into input the numbers and operators together like for example:-
  1+(2 *3/4)   or
  (-0.1*9)-(-2+0.9)   or
  ((3/-9.8-1)+(0.2*-8))+0.9+4
  And on clicking the submit button, it will show the correct result. It takes negative numbers as valid input.

- Source code explanation –
  1. At first in the client side i.e. in calculator3.html file, through the form the user inputs numbers and operators together as mentioned above. On clicking the submit button, calculator1 function is called.
  2. The calculator1 function which checks for parenthesis. If parenthesis is present, then it parses the given input and takes the first 2 numbers and 1 operator from inside the parenthesis with the help of regular expression. If, parenthesis is not present then it directly calls the cal1 function which operates as the previous program.
  3. Then, it extracts only the expression from the parenthesis into 2 numbers and 1 operator and calls the ajax function, which is here named as ajaxFunction.
  4. After that, ajax function is called.
  5. The ajax function catches the inputs of the user and sends it to the calc.php file in server side.

6. In server side (i.e. in calc.php file), on receiving the input, it matches the operator and performs the required operation.
7. After performing the operation, server side i.e. calc.php sends back the result to client side.
8. On receiving the reply from the server side, calculator3.html i.e. client side displays the result, along with the past history (if applicable).
9. Again, the steps from 2 is repeated until no operator is found in the input expression and also replaces arg1, arg2 and op with the value returned from server.
10. After, the operations inside each parenthesis is performed, the values are replaced in the main expression.
11. After, all the parenthesis operations are performed; the cal1 function is again called to repeat the steps as the previous program.

4. Step 3 – Client side – First variant

- In this step, the source code files contains calculator4.html and calc_server.php
- There are 1 input field and 1 submit button.
- Compilation –The input field can take into input only sin(x) or cos(x) like for example:-
  sin(x) or
  cos(x)
  And on clicking the submit button, it will show the correct graph. It takes only small letters sin(x) and cos(x) as valid input.
- Source code explanation –
  1. At first in the client side i.e. in calculator4.html file, through the form the user inputs either cos(x) or sin(x).
  2. On clicking the submit button, ajax function is called which is here named as ajaxFunction.
  3. The ajax function catches the inputs of the user and sends it to the calc_server.php file in server side.
  4. In server side (i.e. in calc_server.php file), on receiving the input, it matches the input and plots the required graph with the help of GNU plot.
  5. After plotting the graph, server side i.e. calc_server.php saves the graph in out.png file. Converts it with base 64 encode and sends back the result to client side.
  6. After that, the server side deletes the file png generated.
  7. On receiving the reply from the server side, calculator4.html i.e. client side displays the graph, along with the past history (if applicable).

5. Step 3 – Client side – Second variant

- In this step, the source code file contains only calculator5.html

- There are 1 input field and 1 submit button.
- Compilation –The input field can take into input only sin(x) or cos(x) like for example:-
  sin(x) or
  cos(x)
  And on clicking the submit button, it will show the correct graph. It takes only small letters sin(x) and cos(x) as valid input.
- Source code explanation –
  1. At first in the client side i.e. in calculator5.html file, through the form the user inputs either cos(x) or sin(x).
  2. The canvas width and length is described in the form itself.
  3. On clicking the submit button, series function is called.
  4. Series function checks whether the input is sin(x) or cos(x) and then calls sine function or cosine function respectively.
  5. On calling the sine function, canvas drawing is started.
  6. X-axis and y-axis is plotted first.
  7. Then for loop is executed from -3.14 to +3.2 (+3.14 is not taken as in taking upto only 3.14 all the points are not getting plotted and hence the graph is not getting complete).
  8. With the help of taylor series, sine function is plotted in the canvas.
  9. Required scaling is also done in the canvas, so that the curve becomes large enough to be visible and fit appropriately according to the canvas size.
  10. The same steps are repeated from 5, in case of calling cosine function. Only change is, cos functional graph is drawn with help of taylor series.
  11. Here, server is not contacted at all.

6. Step 3 – Client side – Third variant

- In this step, the source code files contains calculator6.html and calc.php
- There are 1 input field and 1 submit button.
- Compilation –The input field can take into input only sin(x) or cos(x) like for example:-
  sin(x) or
  cos(x)
  And on clicking the submit button, it will show the correct graph. It takes only small letters sin(x) and cos(x) as valid input.
- Source code explanation –

1. At first in the client side i.e. in calculator6.html file, through the form the user inputs either cos(x) or sin(x).
2. The canvas width and length is described in the form itself.
3. On clicking the submit button, series function is called.
4. Series function checks whether the input is sin(x) or cos(x) and then calls sine function or cosine function respectively.
5. On calling the sine function, canvas drawing is started.
6. X-axis and y-axis is plotted first.

7. Then for loop is executed from -3.14 to +3.2 (+3.14 is not taken as in taking upto only 3.14 all the points are not getting plotted and hence the graph is not getting complete).
8. The taylor series function calculation is send to the server with the help of re-using the above program that handles parenthesis i.e. by calling calculator1 function.
9. The server i.e. calc.php does the required operation and sends the value back to the client.
10. Then the graph plotting starts.
11. Required scaling is also done in the canvas, so that the curve becomes large enough to be visible and fit appropriately according to the canvas size.
12. The same steps are repeated from 5, in case of calling cosine function. Only change is, cos functional graph is drawn with help of taylor series.