**Video Script: Section 6 Video 5 – using a button to avoid frequent updates**

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Description | Action on screen | Narration |
| 1 | Introduction  (Outcome and why it is desirable)  This should give the viewer an idea of the outcome of the task at the beginning of the videos and set the stage and expectations of the viewer. | Refer to PPT | In this video, we are going to see how to use a submit button to send the content of the form to the server. |
| 2 | Context(Problem/Solution)  Present the viewer with a real-world solution and how the situation would pose as a challenge. It always helps to draw the viewer's attention using a use-case. Metadata template can be used here. |  | If your graph requires a substantial time to be calculated, updating it as soon as a control is modified will result in poor user experience. |
| 3 | Guidance (How to do it and how it works): |  | Shiny offers a submit button, which sends the content of the form to the server only when it is pressed on. This way, all the changes will be sent to the server in one go, when the user is satisfied with them. |
| 4 |  | Open R and run  library(“shiny”)  runApp(“activity\_06\_05”) | Open R and run in the command line:  library(“shiny”)  runApp(“activity\_06\_05”) |
| 5 |  |  | The graph shows the distribution of the diamonds broken down over two of the diamonds’ properties. |
| 6 |  |  | Notice how selecting properties does not update the graph immediately. |
| 7 |  |  | The user has to click on the button ‘refresh the graph’ for the plot to be re-generated. |
| 8 |  | Switch back to R, open UI.R and server.R in the editor. | Look at the source in R. |
| 9 |  | Look at UI.R  Highlight  submitButton("Refresh the graph") | The button is added to the side bar with:  submitButton("Refresh the graph") |
| 10 |  | Look at server.R | Server.R doesn’t have a reference to the button. Although the plot depends on input$firstProperty and input$secondProperty, the expression will only be evaluated when the button is pressed.  In other words, the button acts as a delay mechanism, which prevents the propagation of the changes until it is pressed. |
| 11 |  | Highlight  p <- ggplot(diamonds) +  geom\_bar( aes\_string( x = input$firstProperty, fill = input$secondProperty),  position = "dodge") +  scale\_fill\_brewer(type='qual', palette=2) +  ggtitle("Distribution of the diamonds")  print(p) | We used ggplot2 to create the graph. Note that in this case we have to explicitly print the ggplot2 object in order to return a graphical output. As usual with ggplot2, graphs are rendered only when they are printed. |
| 12 |  |  |  |
| 13 |  |  |  |
| 14 |  |  |  |
| 15 |  |  |  |
| 16 | Conclusion:The video concludes by showing the viewer that the goal has been achieved, and reminding them why they should be happy about that. A PowerPoint summary slide with the key points emphasized would make it easier for the viewer to remember what was covered in the video | Back to PPT | We saw how to use a submit button to avoid updating the results too frequently. This is especially useful if the computations take a substantial amount of time.  In the next section, we’ll learn about more interactive controls offered by Shiny and how to publish your work. |