

Module 2: Description of default flexdashboard and creating columns/rows of different orientations

contributed by

Mr. Debatosh Chakraborty
Project Research Assistant,
R Team, FOSSEE, IIT Bombay

Ms. Usha Viswanathan
Sr. Project Manager,
FOSSEE, IIT Bombay

Objective

1. A description of the dashboard elements created in Module 1(refer to Figure 1).
2. Rename columns on the dashboard
3. Resize the created columns
4. Different orientation properties of flexdashboard
5. Creating some orientations

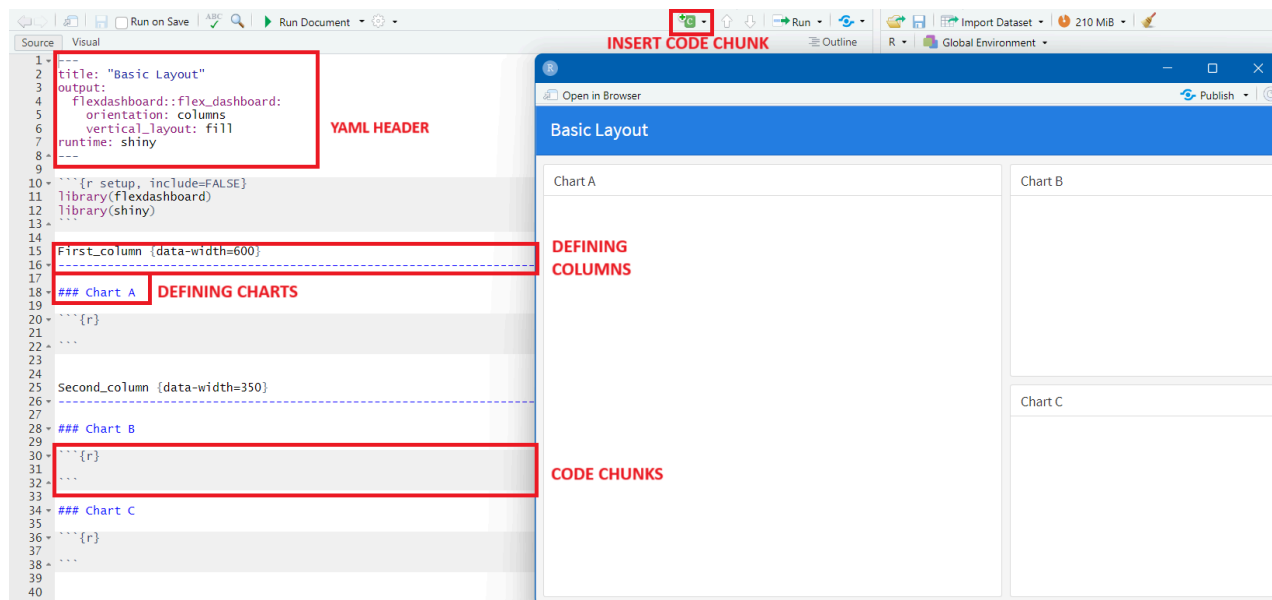



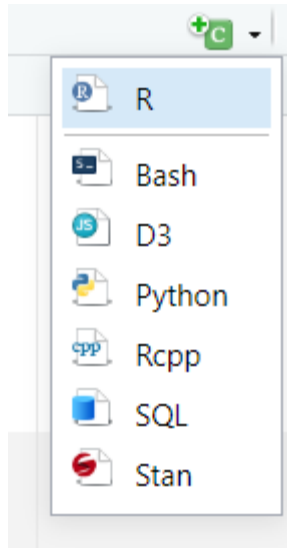
Figure 1: Default layout of the dashboard created in module 1.

1. Description of the default layout

- a. The default orientation of the dashboard is specified as column-wise, and the layout is specified as 'fill', i.e., non-scrollable.
- b. This is specified in the YAML header of the document.

Figure 2: YAML header of the dashboard.

- c. The dashboard consists of two columns and three charts, one chart on the left and two on the right, as shown in Figure 1.
- d. **Columns:**
 - i) Level 2 markdown headers (-----) within dashboards define columns.
 - ii) First line Column name; second line a few dashes (-----) create a column (see Figure 1).
 - iii) Alternatively, first line ## Column Name.
- e. **Chart:**
 - i. Any chart is a placeholder for different elements like plots, tables etc.
 - ii. By default, the individual charts are stacked vertically within each column.
 - iii. And the name of the chart is specified in Level 3 headers (###).
 - iv. The code for the graph to be displayed is specified in the code chunk below the chart name.
- f. **Inserting a code chunk:**
 - i. The process is identical to inserting a code chunk in R markdown.
 - ii. You can click on +C icon () on the top bar. (Figure 3)
 - iii. Then choose between any code script, that you want to generate.
 - iv. The choices can be R, Python, Cpp, SQL, etc. (Figure 3)



v. Alternatively, press 'Ctrl+Alt+I' to insert the R code chunk.

2. Rename columns on the dashboard.

- A. Nomenclature of columns to be done as per the analysis in that column.
- B. Nomenclature of columns does not change the output.
- C. It is for programmers' convenience.
- D. Renamed the existing 'Column' to 'First_Column' and 'Second_Column'. (see Figure 3).

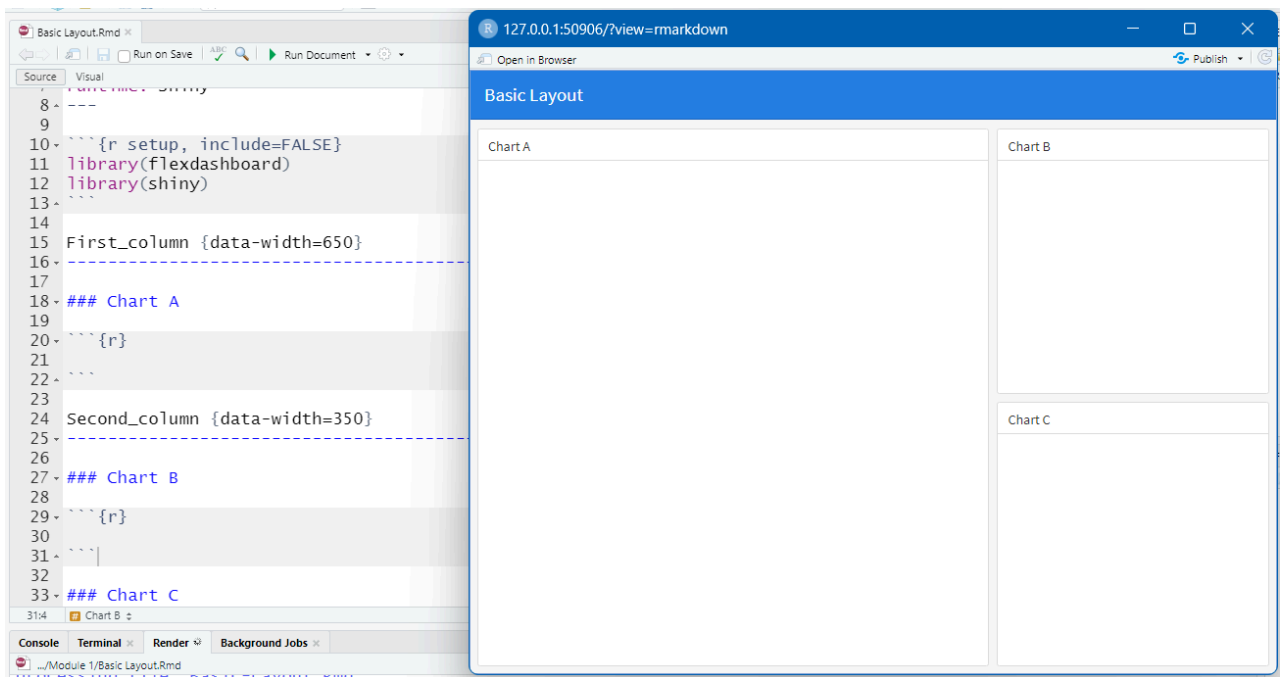


Figure 3: Renaming column names.

3. Resize the created columns on the dashboard

B. Column/Row size:

- A. The orientation determines the control of the height or width parameter.
- B. If the **orientation is 'row'**:
 - i. The width is automatically set to 100%.
 - ii. The number of rows defined takes up the entire space.
 - iii. The controllable parameter is **'data-height'**.
- C. If the **orientation is 'column'**:
 - i. The height is automatically set to 100%.
 - ii. The number of columns defined takes up the space.
 - iii. The controllable parameter is **'data-width'**.
- D. To change the **width of a column**,
 - i. Beside the column name write **{data-width=<value>}**
 - ii. The value is the required width.
 - iii. **Caution:** No spaces in between the '{}'.
- E. To change the **height of a row**,
 - i. Beside row name write **{data-height=<value>}**
 - ii. The value is the required height.
 - iii. **Caution:** No spaces in between the '{}'
- F. The **<value>** defines the **proportional size of the columns** defined,
 - i. **By default, the value of the size parameter is 576.**
 - ii. The total width/height of the columns/rows takes the available space.
 - iii. The ratio of the size parameters defined will determine the size of the individual columns/rows.
 - iv. For example, defining three columns with **{data-width=1} ; {data-width=2}; {data-width=3}** **will produce three columns in the ratio 1:2:3.**
 - v. *In the backend, The columns/rows of the dashboard are created by the flexbox property of CSS.*
 - vi. *The <value> of {data-height} or {data-width} controls the grow and shrink parameters of flexbox.*
- G. If the values of the 'data-width' parameters of the First and Second Columns are interchanged, the output Shows that Chart B and Chart C will take up more space than Chart A (see Figure 4).

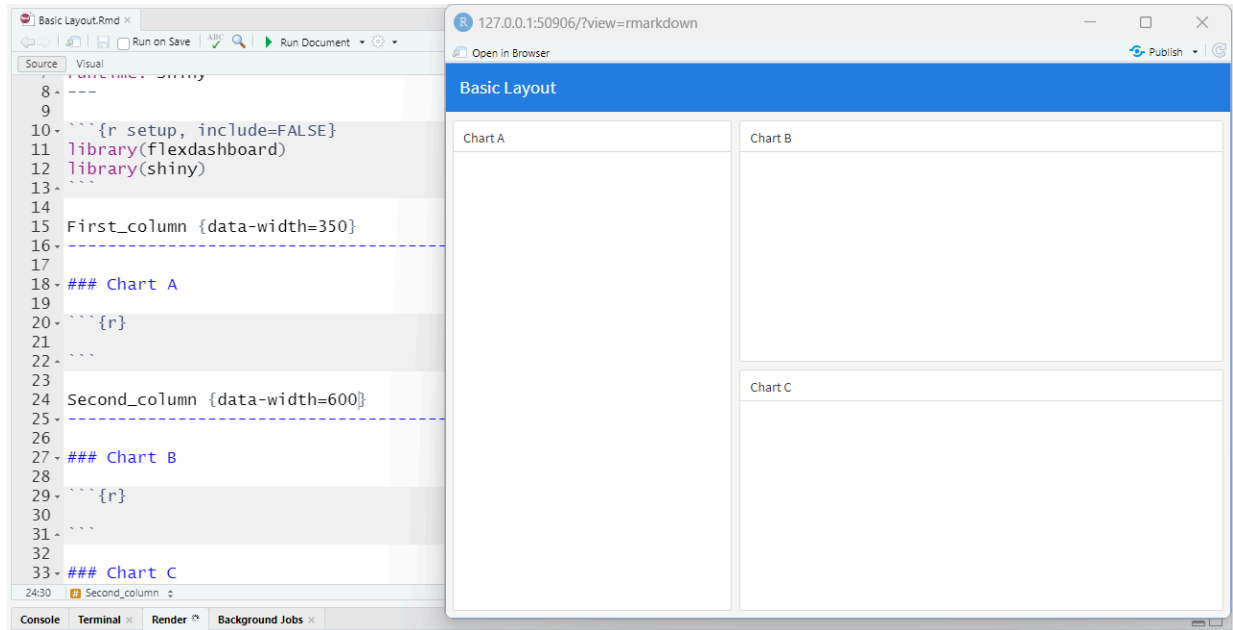


Figure 4: Reversing the width of the columns.

3. Different orientations of flexdashboard

The orientation/arrangement for **the entire dashboard** is controlled by **the orientation option in the YAML header**.

A. Orientation: column

1. Every separate level 2 header (-----) defines a new **column** for the **entire dashboard**.
2. Anything defined in the **column** is placed in **separate rows** in the **column only**.
3. **Use Case:** When the number of **columns** is **fixed** and the number of **rows in the columns** is **variable**.

B. Orientation: row

1. For **row orientation**, every separate level 2 header (-----) defines a new **row** for the **entire dashboard**.
2. Anything defined on the **rows** is placed in **separate columns** in the **row only**.
3. **Use Case:** When the number of **rows** is **fixed** and the number of **columns in the rows** is **variable**.

4. Creating some layouts

A. First column 2 rows, second column 1 row

- a. **Orientation: column**
- b. Define 2 columns with level 2 header.
- c. Define 2 charts inside the first column.
- d. The code is given as (Figure 5)

First_column

Chart A

```
```{r}
```

```
....
```

### Chart B

```
```{r}
```

```
....
```

Second_column

Chart B

```
```{r}
```

```
....
```

```

title: "Columns"
output:
 flexdashboard::flex_dashboard:
 orientation: column
 vertical_layout: fill
runtime: shiny

```

```
```{r setup, include=FALSE}  
library(flexdashboard)  
library(shiny)  
---
```

First_column

Chart A

```
```{r}
```

```
....
```

### Chart B

```
```{r}
```

```
....
```

Second_column

Chart B

```
```{r}
```

```
....
```

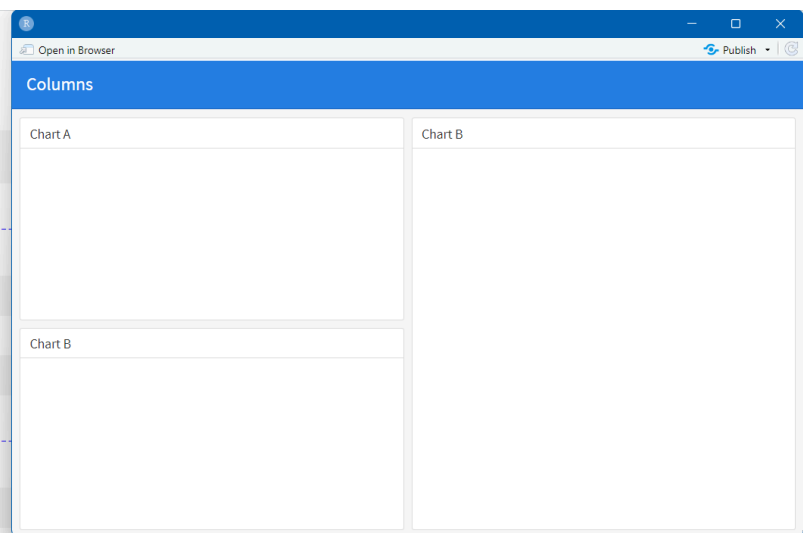


Figure 5: First column 2 rows, second column 1 row

## B. First row 2 columns, second row 1 column

- a. **Orientation: row**
- b. Define 2 rows with level 2 header.
- c. Define 2 charts inside the first row.
- d. The code is given as (Figure 6)

```
First_row

Chart A
```{r}
```

Chart B
```{r}
```

Second_row

Chart B
```{r}
```
```

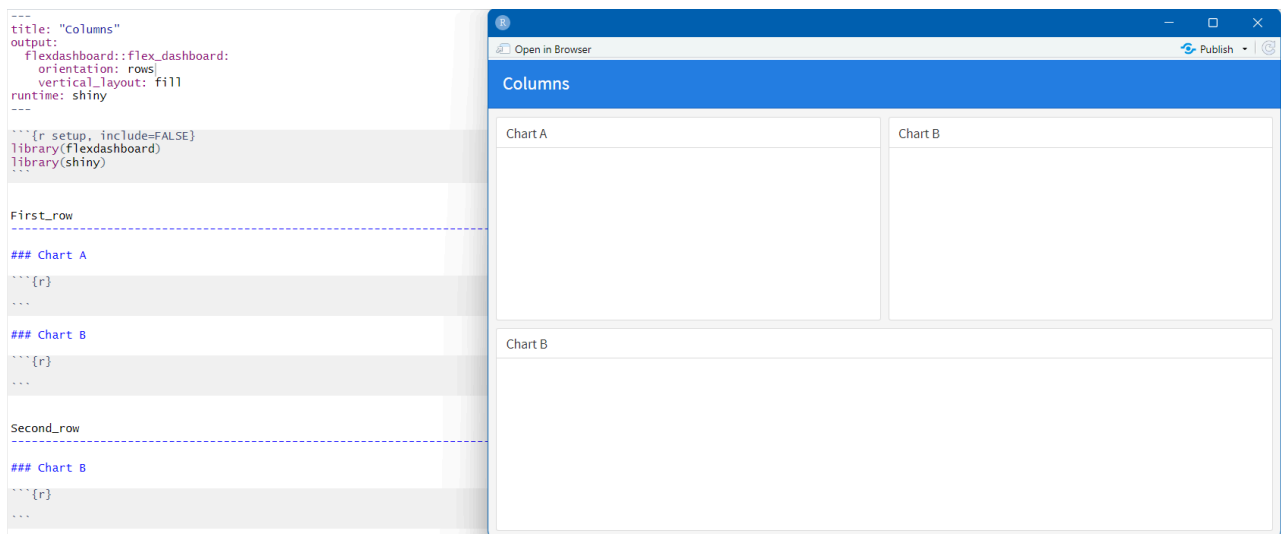


Figure 6: First row 2 columns, second row 1 column

C. 2 x 2

- a. **Orientation: row/columns**
- b. Define 2 level 2 headers.
- c. Define 2 charts inside the first and second header.
- d. The code is given as (Figure 7)

First\_header

---

### Chart A

```
```{r}
```

```
....
```

Chart B

```
```{r}
```

```
....
```

Second\_header

---

### Chart A

```
```{r}
```

```
....
```

Chart B

```
```{r}
```

```
....
```

```

title: "Columns"
output:
 flexdashboard::flex_dashboard:
 orientation: rows
 vertical_layout: fill
runtime: shiny
```

```
```{r setup, include=FALSE}  
library(flexdashboard)  
library(shiny)  
....
```

First_row

Chart A

```
```{r}
```

```
....
```

### Chart B

```
```{r}
```

```
....
```

Second_row

Chart A

```
```{r}
```

```
....
```

### Chart B

```
```{r}
```

```
....
```

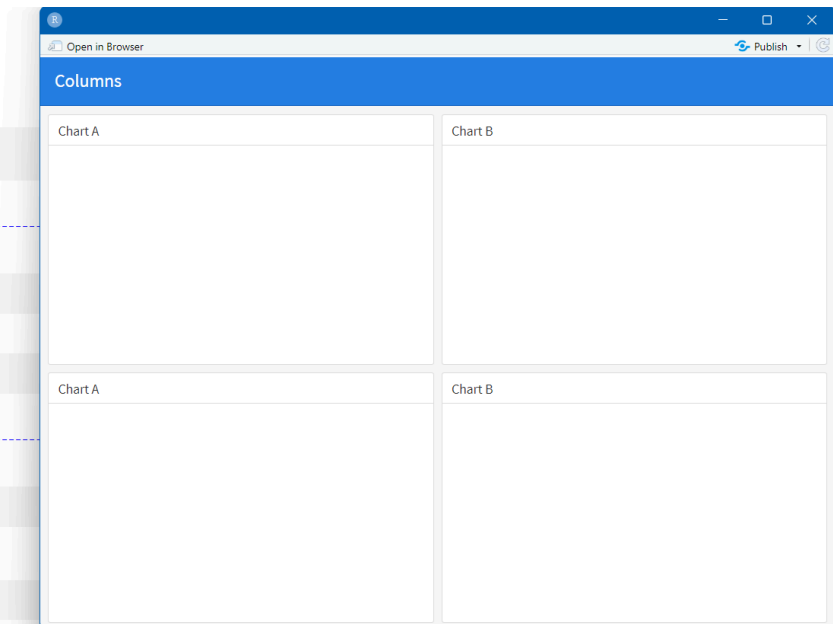


Figure 7: 2x2 orientation