**Grid View Drag & Drop**

In grid view drag and drop, widgets can be drag and dropped in a grid format without overlapping with each other within a Box (Container with fixed size).

# **Widget Tree Structure**

* Scaffold
  + App Bar (With fixed Size)
  + Column
    - Container (Seat Type Container with fixed size)
    - Column
      * Container (Fixed Height)
      * Sized Box (Grid Height)
        + Single Child Scroll View (Get Scroll offset)

Sized Box (Dynamic Height according to main axis count)

Stack

Grid View

Container (Having Borders)

Stack (Seat Container with horizontal padding)

Positioned (Multiple Numbers)

Stack (wheel Container)

Stack (Door Container)

* + - * Sized Box (Fixed Height)
    - Container (Buttons container with fixed Size)

Multiple positioned widgets are stored inside the stack widget with different x and y coordinates. Whenever the draggable containers are dragged their x and y coordinates are updated.

Note: - Whenever the draggable widget is dragged, it will take the offset (x and y coordinates) according to the whole screen size even if it is inside a container less than the size of the screen. But whenever the dragged widget is placed inside the fixed size container than the position of the widget will be according to the size of the container. So that spacing around the container should be removed from the offset of the dragged widget and scroll amount should be added in order to place the widget at exact place where it is dropped.

# **Requirements**

1. Cross Axis Count (Constant Integer)
2. Vehicle width (Feet or Inch)
3. Grid Container top margin (Space between Seat type container and grid container)
4. Angle (degree to which seats will be rotated)
5. Horizontal padding (For grid container)
6. Grid Gap (Constant Integer according to the screen size)
7. Seat Type container Height
8. Button container Height
9. Grid Height (The height of the grid which could be scrolled and be used to display the seats)
10. Grid Width
11. Main Axis Count (Dynamic Integer)
12. Grid scrollable height (Defined by main axis count)
13. Scroll Controller (To get the amount scrolling done)
14. App Bar Height (If app bar is available)
15. List of seat types
16. List of seats
17. List of wheels
18. List of doors

These variables are most required and will be used to display the Grid View with correct alignment. The other variables that will be required along with these variables to calculate the position of the dragged widget are screen height and width and height and width of seats.

Note: - Height and width of the seats are dynamic. Their sizes are calculated according to the grid width and vehicle width.

# **Adding widget and updating position**

While dragging and dropping the widget, the offset given by the draggable widget is modified and added inside the list of the seats.

1. New widget will be created only If the dragged widget is dropped inside the grid container
2. The size of the widget is calculated while adding new widget
3. The minimum position of x and y coordinates is 0, maximum position of x coordinate is less than the grid width (Grid width – seat width), and maximum position of y coordinate is less than grid height (Grid height – seat height) so that the dragged seat stay inside the container
4. Wheels are only positioned at the left edge of the grid
5. Doors are only positioned at the edge of both left and right side of the grid
6. If scrolling is done then the scroll amount is added to the y coordinate
7. The new coordinates are then checked for alignment within the grid lines. If the seat position lies between the grid gap, then it is aligned to the nearest grid lines.
8. While updating the position, the difference between previous and new offset of the seat are calculated and then checked. Wheels and doors position is updated according to y-coordinate.
9. Then the modified offset is checked with another widget’s offset to find out whether it touches other widget available inside the grid or not. “for” loop is used to check with every widget inside the list of seats. While updating the position, the widget is not checked with itself.
10. If the dragged widget reaches the end of grid container, then new lines equal to the size of the dragged widget is added at the bottom of the grid container and it will be scrollable
11. Then new seats are saved again

# **Showing previous data (also inside different screen size)**

If previous data is available, then the data are shown inside the grid view. If the data are being showed in different screen, then the grid view is scaled according to the screen width but the main axis count modified If previous main axis count is greater.

1. The grid gap, main axis count, seats, wheels, and doors are saved
2. If the saved grid gap or main axis count does not match with the current ones, then the dimensions and coordinates of the previous seats are modified
3. If previous main axis count is greater than the current one, then the previous one is assigned to the current one.
4. Then the seats’ coordinates and sizes are also changed according to the new grid dimensions
5. Then new dimensions with new seat coordinates are saved
6. Same processes are followed for wheels and doors

# **Updating widget property and size**

The size of each widget can be changed.

1. The widget new height and width are calculated
2. The widget with new height and width is checked for overlapping with other widgets inside the grid area, if the new size off the widget overlaps with other than its size is not updated.
3. If the overlapping does not occur, then the size is updated without exceeding the grid area.
4. Then the property of the widget is also updated and seats are saved again.