6G Special layout widgets Lab

We have one major task to focus on, the adding of seats to our cart.

Showing the cart contents

We have one last major step, allowing the user to add seats to their cart. Let's start by displaying the cart contents on the Checkout widget. They can review their seats, their total and can check out by hitting the FAB.

Notice that that cart is in tabular form. I *guess* that we could try to recreate that with Row()s and Column()s but that's the wrong tool. For tabular data, we should reach for a Table() widget.

1. Edit Checkout. Find your placeholder that says "Cart will go here". You'll want to create a Table() widget in that spot.

Notice that all Table()s have a children property which must be populated with a List<TableRow>. And of course you realize that each TableRow must have a children property which will be the cells or columns. So it's a List of Lists. Your mission will be to create that List and assign it to the Table()'s children property.

2. You can do that any way that you like, but it might be good to write a method that returns back a List of TableRow()s. Maybe write this:

```
Table(children: _makeTableRows()),
```

3. And then write the _makeTableRows() method. See if you can do it without any help. But if you get stuck, here's one way to do it:

```
List<TableRow> _makeTableRows() {
  List<TableRow> rows = <TableRow>[];
  double subtotal = 0.00;
  for (var seat in _cart['seats']) {
    subtotal += seat['price'];
    rows.add(TableRow(children: [
      Text('Table ${seat['table_number']} Seat ${seat['seat_number']}'),
      const Text(''),
      Text('${seat['price']}'),
    ]));
  }
  rows.add(TableRow(children: [
    const Text(''),
    const Text('Subtotal:'),
    Text(subtotal.toString()),
  ]));
  double tax = subtotal * 0.0825;
  rows.add(TableRow(children: [
    const Text(''),
```

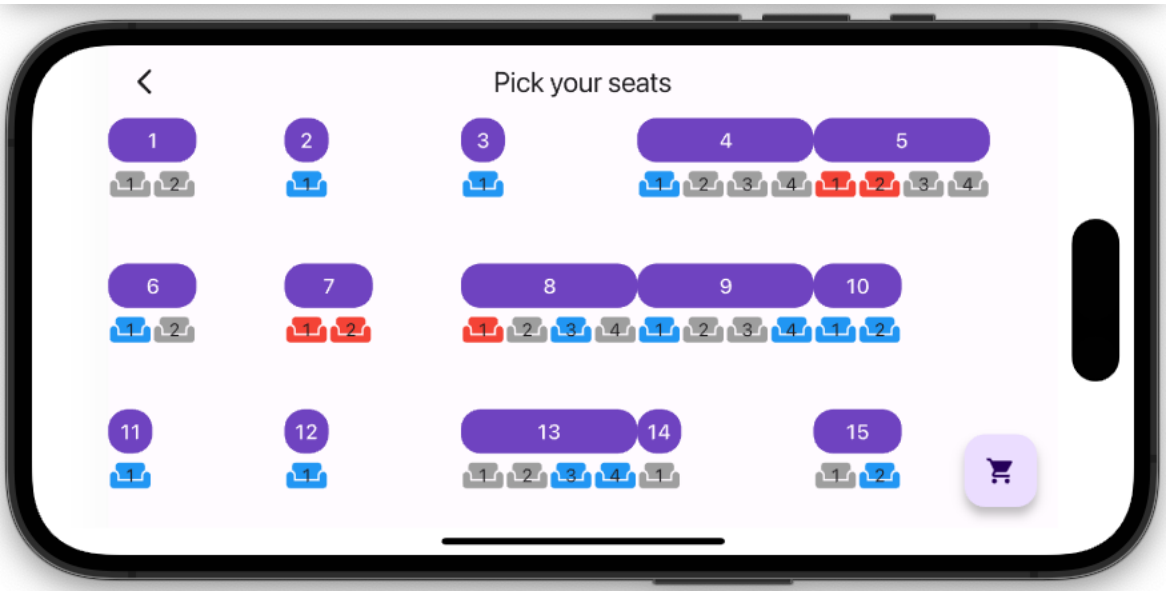
```
    const Text('Tax:'),
    Text(tax.toString()),
  ]));
  rows.add(TableRow(children: [
    const Text(''),
    const Text('Total:'),
    Text((subtotal + tax).toString()),
  ]));
  return rows;
}
```

4. Run and test. Tune as needed just to get the Table() with data on your screen.

5. Bonus! Using the Dart language, format the numbers to look like currency. (Hint: import the intl package and use NumberFormat.currency().)

Drawing a seat map in PickSeats

This would be a big one if we didn't give you some starter code! We're going to create this:



We need a widget that has a List of children. Should we use a Row? Nope, we don't want them side-by-side. Column? No again. The widget must allow us to draw the tables and seats by placing them with X and Y coordinates.

This is a job for a Stack!

6. Edit pick_seats.dart. Inside your Container, remove the Text() placeholder and add a Stack(). Its children property should be this:

```
children: theater["tables"]
    .map<Widget>((table) => daam_table.TheaterTable(
          table: table,
          usableHeight: height,
```

```
          usableWidth: width,
        ))
    .toList(),
```

7. Obviously we need a usableWidth and a usableHeight. Get these from a MediaQuery:

```
var screenSize = MediaQuery.sizeOf(context);
var usableHeight = screenSize.height - AppBar().preferredSize.height;
var usableWidth = 0.8 * screenSize.width;
```

8. We also need a TheaterTable() widget. Look in the starters folder. Copy theater_table.dart.starter into your lib folder and remove the .starter file name extension.

9. import theater_table.dart into pick_seats.dart.

10. Alright, run and test. You should see a map of the seats while in landscape mode.

Adding to the cart

If you run and test with your current code, you'll see that no matter which film or how many seats the user chooses, you always end up with the same four things in the cart. Why? Because we hard-coded the cart in the FAB's onPressed event in PickSeats.

11. In PickSeats, find the FAB's onPressed event.

12. Delete these lines:

```
// For debugging only - Load the cart up with data
cart = {
  "showing_id": 100,
  "seats": [7, 8, 10, 22],
};
rawState.set("cart", cart);
```

Now when you check out, the cart won't be clobbered and replaced.

13. Give it a try. Pick a film, showing, and a couple of seats. You should only see the tickets you reserved.

Bonus!! Understanding the provided code

Let's face it, the theater_table.dart.starter you included has a ton of code pre-written to make it easy and that code isn't super-straightforward.

14. Take some time to look through it to see how we created that seat map. See if you can figure out how each feature was done.

15. As an extra bonus, see if you can duplicate the functionality without copying any lines from the provided code starter. It's a big challenge, but what a learning opportunity!