Step 2 – Todo UI

Building Stateful Widgets and Dynamic UI

Stateful Widgets

- Reviewing the core counter app code, we can see that when we want the internal state (and therefore the view) to refresh we call a setState() function.
- This function tells flutter to reload the UI build methods and determine any changes it needs to make.
- However, let's say we have a list of items that we want to update, the UI will need to respond to an additional widget if a new item is added.
- To do this we use a builder to infer the collection of widgets using the number of items.

Create a list

- Inside the _TodoHomePageState class, add a final list of todos like this:

```dart
final List<Todo> todos = <Todo>[
  Todo(name: "Shopping", description: "Pick up groceries"),
  Todo(name: "Paint", description: "Recreate the Mona Lisa"),
  Todo(name: "Dance", description: "I wanna dance with somebody"),
]; // <Todo>[]
```

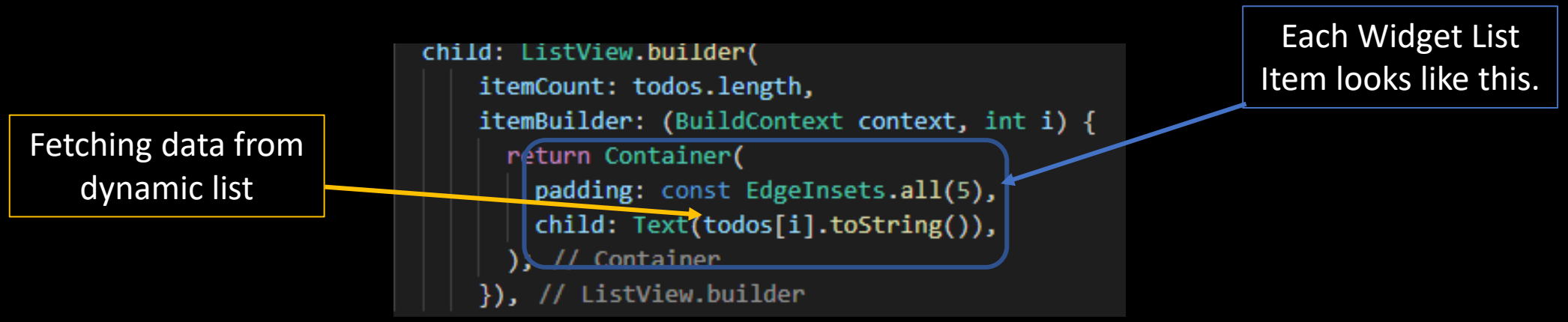
- This will give us some temporary/fake data to work with in our application.
- Now we can start building our UI using an ItemBuilder

ListView

- In the body of the scaffold, add a new centre widget.
- In its child we're going to create a listview by dynamically calling the ListView.builder() function.
- This takes a few parameters that will allow us to generate a template style widget.
- For now, we're going to just output the list.
- The builder function takes a few options, but the once we're going to add are the itemCount and itemBuilder.

itemBuilder

- ItemCount: tells the ListView.Builder() how many iterations
- Inside the itemBuilder parameter, we need to pass a function that flutter uses to build the UI elements. It returns the widgets by looping over a collection.
- ItemBuilder function looks like this:



Exercise – Customise View

- Update the returned widget by combining other widgets to create a visually pleasing UI for each element.
- Feel free to utilise the properties of the Todo class and position the information appropriately within the widget structure.
- Suggested widgets:
  - BoxDecoration
  - BoxShadow
  - Colors
  - Margin/Padding: EdgeInsets.all() / EdgeInsets.fromLTRB()
- Also add additional styling to the application from the scaffolding.