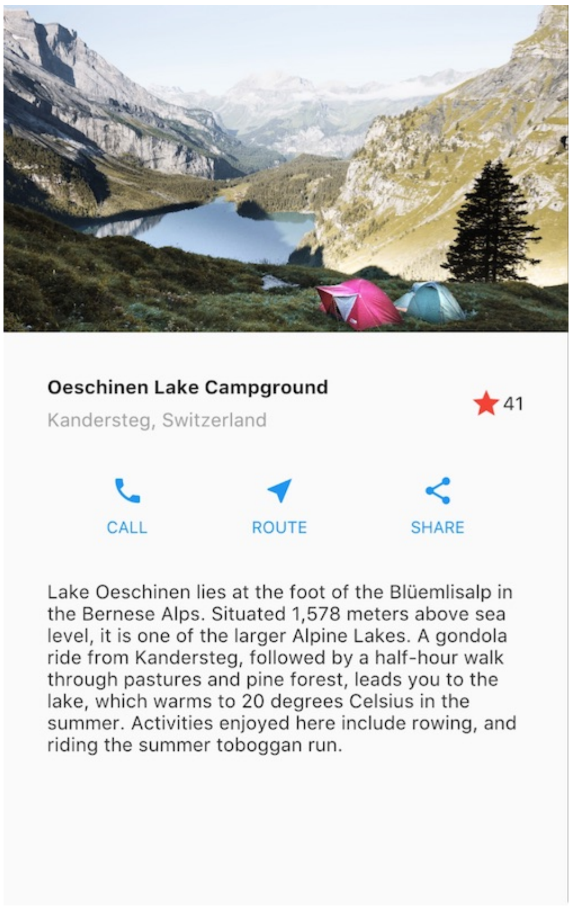
Learning Outcomes / Goals

- How Flutter's layout mechanism works.
- How to lay out widgets vertically and horizontally.
- How to build a Flutter layout.

At the end of this lab you should be able to build the layout shown on the right

You can use the second application you have created for the first lab. Just, change the app bar title and the app title to **'Flutter layout app'**



Oeschinen Lake Campground
Kandersteg, Switzerland
★41

Lake Oeschinen lies at the foot of the Blüemlisalp in the Bernese Alps. Situated 1,578 meters above sea level, it is one of the larger Alpine Lakes. A gondola ride from Kandersteg, followed by a half-hour walk through pastures and pine forest, leads you to the lake, which warms to 20 degrees Celsius in the summer. Activities enjoyed here include rowing, and riding the summer toboggan run.

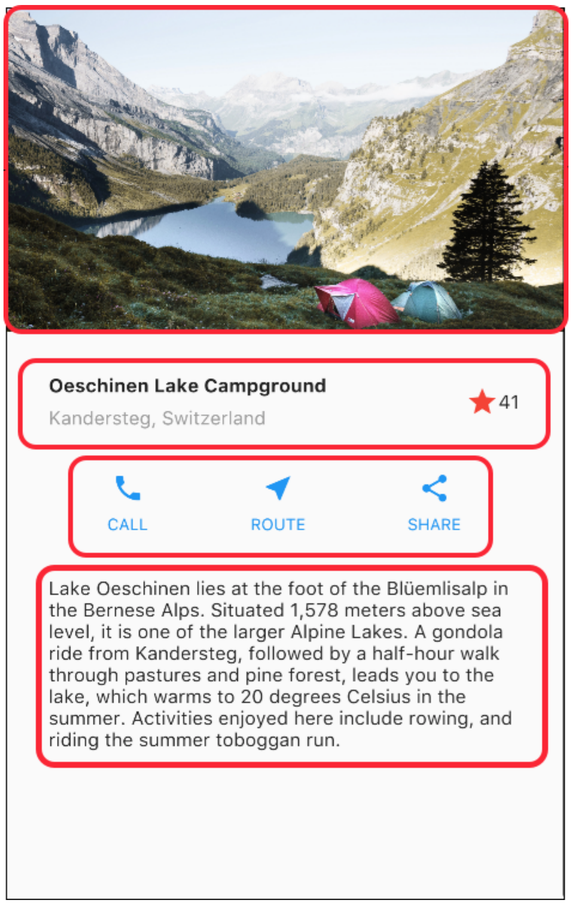
Step 1: Diagram the layout

The first step is to break the layout down to its basic elements:

- Identify the rows and columns.
- Does the layout include a grid?
- Are there overlapping elements?
- Does the UI need tabs?
- Notice areas that require alignment, padding, or borders.

Basic Elements

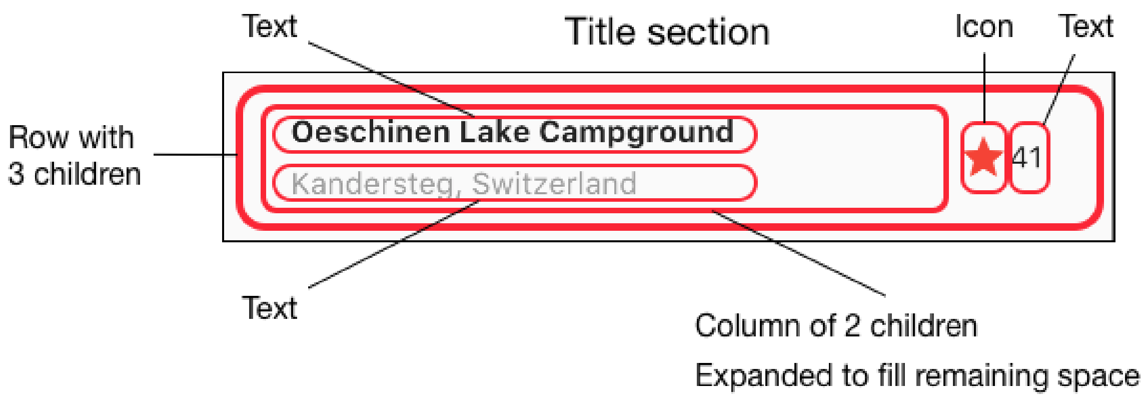
- Column with four children
    - Image
    - Two rows
    - Text



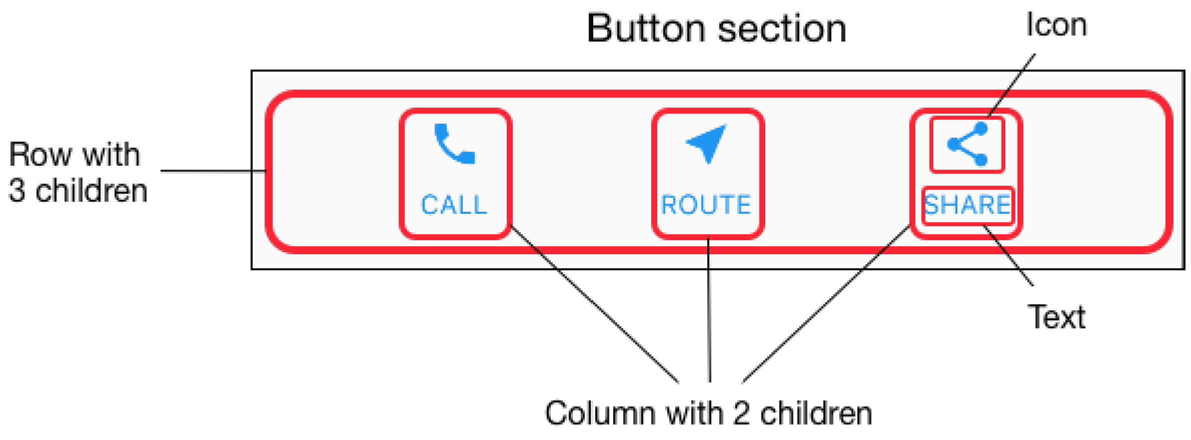
Oeschinen Lake Campground
Kandersteg, Switzerland
★41

Lake Oeschinen lies at the foot of the Blüemlisalp in the Bernese Alps. Situated 1,578 meters above sea level, it is one of the larger Alpine Lakes. A gondola ride from Kandersteg, followed by a half-hour walk through pastures and pine forest, leads you to the lake, which warms to 20 degrees Celsius in the summer. Activities enjoyed here include rowing, and riding the summer toboggan run.

Title Row



Buttons Row



Step 2: Implement the first row

Add the following code at the top of the build() method of the MyApp class

```
Widget titleSection = Container(
  padding: const EdgeInsets.all(32),
  child: Row(
    children: [
      Expanded(
        /*1*/
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            /*2*/
            Container(
              padding: const EdgeInsets.only(bottom: 8),
              child: Text(
                'Ullamco labore ullamco reprehenderit',
                style: TextStyle(
                  fontWeight: FontWeight.bold,
                ),
              ),
            ),
            Text(
              'Aliqua cillum sit ex ipsum',
              style: TextStyle(
                color: Colors.grey[500],
              ),
            ),
          ],
        ),
      ),
      /*3*/
      Icon(
        Icons.star,
        color: Colors.red[500],
      ),
      Text('41'),
    ],
  ),
);
```

/*1*/ Putting a **`Column`** inside an **`Expanded`** widget stretches the column to use all remaining free space in the row. Setting the **`crossAxisAlignment`** property to **`CrossAxisAlignment.start`** positions the column at the start of the row

/*2*/ Putting the first row of text inside a **Container** enables you to add padding. The second child in the **Column**, also text, displays as grey

/*3*/ The last two items in the title row are a star icon, painted red, and the text "41". The entire row is in a **`Container`** and padded along each edge by 32 pixels. Add the title section to the app body like this

Step 3: Implement the second row

The second row contains 3 columns that use the same layout—an icon over a text. The columns in this row are evenly spaced, and the text and icons are painted with the primary color.

Since the code for building each column is almost identical, let us create a private helper method named **`buildIconColumn()`**, which takes a **`Color`**, an **`IconData`** and **`string label`**, and returns a column with its widgets painted in the given color.

Put the **`buildIconColumn()`** method inside **`MyApp`** class, below the **`build()`** method as shown next

```
class MyApp extends StatelessWidget {
```

```
@override
Widget build(BuildContext context) {
   // ···
}

Column _buildIconColumn(Color color, IconData icon, String label)
{
    return Column(
      mainAxisSize: MainAxisSize.min,
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Icon(icon, color: color),
        Container(
          margin: const EdgeInsets.only(top: 8),
          child: Text(
            label,
            style: TextStyle(
              fontSize: 12,
              fontWeight: FontWeight.w400,
              color: color,
            ),
          ),
        ),
      ],
    );
  }
}
```

The function adds the icon directly to the column. The text is inside a `Container` with a top-only margin, separating the text from the icon.

Build the row containing these columns by calling the function and passing the `Color`, `Icon`, and text specific to that column. Align the columns along the main axis using `MainAxisAlignment.spaceEvenly` to arrange the free space evenly before, between, and after each column. Add the following code just below the `titleSection` declaration inside the `build()` method

```
Color color = Theme.of(context).primaryColor;

Widget iconSection = Container(
  child: Row(
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: [
      _buildButtonColumn(color, Icons.call, 'CALL'),
      _buildButtonColumn(color, Icons.near_me, 'ROUTE'),
      _buildButtonColumn(color, Icons.share, 'SHARE'),
    ],
  ),
);
```

Add the icon section to the body:

```
   return MaterialApp(
     home: Scaffold(
       appBar: AppBar(
         title: Text('Flutter Layout'),
       ),
       body: Column(
         children: [
           titleSection,
           iconSection,
         ],
       ),
     ),
   );
```

Note: the above code is the body of the **`build`** method.

Step 4: Implement the text section

Define the text section as a variable. Put the text in a **`Container`** and add padding along each edge. Add the following code just below the **`buttonSection`** declaration:

```
Widget textSection = Container(
  padding: const EdgeInsets.all(32),
  child: Text(
    'Proident enim et duis qui excepteur commodo amet cillum velit
velit ullamco. Adipisicing esse officia ullamco consectetur ex.
Reprehenderit proident enim deserunt laboris. Dolor consequat
occaecat tempor qui deserunt sint sint nulla commodo laboris
deserunt sunt excepteur laboris',
    softWrap: true,
  ),
);
```

By setting `softwrap` to true, text lines will fill the column width before wrapping at a word boundary.

Add the text section to the body:

```
return MaterialApp(
     home: Scaffold(
       appBar: AppBar(
         title: Text('Flutter Layout'),
       ),
       body: Column(
         children: [
           titleSection,
           iconSection,
           textSection,
         ],
       ),
     ),
   );
```

Step 5: Implement the image section

- Create an `images` directory at the top of the project.
- Add **lake.jpg**.

Update the **`pubspec.yaml`** file to include an **`assets`** tag. This makes the image available to your code.

```
flutter:
 uses-material-design: true

 assets:
   - images/lake.jpg
```

- Note that `pubspec.yaml` is case sensitive, and sensitive to white space

Add the Image widget to the body:

```
return MaterialApp(
     home: Scaffold(
       appBar: AppBar(
         title: Text('Flutter Layout'),
       ),
       body: Column(
         children: [
           Image.asset(
             "images/lake.jpg",
             width: 600,
             height: 240,
             fit: BoxFit.cover,
           ),
           titleSection,
           iconSection,
           textSection,
         ],
       ),
     ),
   );
```

`BoxFit.cover` tells the framework that the image should be as small as possible but cover its entire render box.

Step 6: Final touch

In this final step, arrange all of the elements in a `ListView`, rather than a `Column`, because a `ListView` supports app body scrolling when the app is run on a small device.

Change the `Column` in the body to `ListView`

```
  return MaterialApp(
    home: Scaffold(
      appBar: AppBar(
        title: Text('Flutter Layout'),
      ),
      body: ListView(
        children: [
          Image.asset(
            "images/lake.jpg",
            width: 600,
            height: 240,
            fit: BoxFit.cover,
          ),
          titleSection,
          iconSection,
          textSection,
        ],
      ),
    ),
  );
```

Exercise: Construct Widget Tree for this App