Forms in Flutter
TextField, TextFormField
Radio, Checkbox

TextField

- TextField widget allows users to input and edit text.

```
TextField(
 decoration: InputDecoration(  // Adds label and border to enhance UI.
   border: OutlineInputBorder(), // Creates a border around the TextField.
   labelText: 'Username'), // Title
 onChanged: (value) {
   // Do something with the user input.
 onSubmitted: (value) {
   // Do something with the user input.
 })
```

Other TextField Properties

- **`controller`**: Allows retrieval and manipulation of text.
- **`decoration`**: Customizes appearance (hint, labels, borders).
- **`keyboardType`**: Changes keyboard layout (text, number, email).
- **`obscureText`**: Hides input for passwords.

Handling User Input

- **Controllers** allow real-time access to the TextField’s content.
- Useful for validation, clearing text, and managing state.

```
final TextEditingController _controller = TextEditingController();

print(_controller.text); // To get text

_controller.clear(); // To clear text
```

Customizing TextField Appearance

- Use `InputDecoration` to customize the appearance.

```
decoration: InputDecoration(
   border: OutlineInputBorder(),
   icon: Icon(Icons.person),
   labelText: 'Age',
   hintText: 'Enter your age',
   suffixText: 'years',
   fillColor: Colors.lightBlue[50],
   filled: true),
```

Validation in TextFields

Use `TextFormField` in forms to apply validation logic.

```
TextFormField(
 controller: passwordController,
 decoration: const InputDecoration(labelText: 'Password'),
 obscureText: true,
 validator: (value) {
   if (value == null || value.isEmpty) {
     return 'Please enter your password';}
   if (value.length < 6) {
     return 'Password must be at least 6 characters long';}
   return null;
 },),
```

Radio field

Allows users to select one option from a list of choices.

```
String? role = 'Student';
Radio<String>(
 value: 'Student',
 groupValue: _role,
 onChanged: (String? value) {
   setState(() {
     role = value;
   });
 },
),
```

Checkbox field

Allows the user to make (yes/no) choices.

```
bool agree = false;
Checkbox(
 value: agree,
 onChanged: (bool? value) {
   setState(() {
     agree = value ?? false;
   });
 },
),
```

Form widget

- `Form` widget is used to group and manage multiple form fields (like TextFormField, DropdownButton, etc.).
- simplifies validation and submission of input fields in app.
- Commonly used in login forms, registration forms, and other user data collection interfaces.
- Easily manage *form states* using `GlobalKey`.

Form widget Example

```
final _formKey = GlobalKey<FormState>();
Form(
 key: _formKey,
 child: Column(
   children: [
     TextFormField(
       decoration: ... // Text field decoration
       validator: ... // validate input
     ElevatedButton(
       onPressed: () {
         if (_formKey.currentState!.validate()) {
           // Process the input
         }},
       child: Text('Submit'),),],),)
```

_formKey: a unique identifier for the Form widget.

currentState!: the current state of the Form is not null.

validate(): validate all Form Fields widgets within the Form. (True = all validators return null)

Practice

- Make this form.

- To calculate the ideal weight:

Male:

```
idealWeight = 50 + 0.91*(height - 152.4) + (age - 20) / 4;
```

Female:

```
idealWeight = 45.5 + 0.91*(height - 152.4) + (age - 20) / 4;
```