

05

유저 입력 위젯

다양한 위젯

텍스트 필드

입력된 데이터 얻기 – TextEditingController

- 텍스트 필드(TextField)는 사용자에게 글을 입력받는 위젯
- TextEditingController를 이용하면 텍스트 필드에 입력된 데이터 획득, 변경 이벤트 감지 등의 작업

• 텍스트 필드에 입력된 데이터 얻기

```
class TextState extends State<TestScreen> {  
  final controller = TextEditingController();  
  ... (생략) ...  
  @override  
  Widget build(BuildContext context) {
```

```
    return Column(  
      children: [  
        TextField(  
          style: TextStyle(fontSize: 15.0),  
          controller: controller,  
        ),  
        ElevatedButton(  
          child: Text('submit'),  
          onPressed: () {  
            print('submit : ' + controller.text);  
          },  
        ),  
      ],  
    );  
  }  
}
```



텍스트 필드

입력된 데이터 얻기 – TextEditingController

- 텍스트 필드에 한 자 한 자 입력될 때마다 처리해야 할 로직이 있다면 `addListener()` 함수로 데이터가 변경될 때마다 실행할 함수를 지정
- 더 이상 텍스트 필드에 값을 감지할 필요가 없다면 `dispose()` 함수를 호출

• 텍스트 필드값 변경 감지 방법

```
@override
void initState() {
  super.initState();
  controller.addListener(_printValue);
}

@override
void dispose() {
  super.dispose();
  controller.dispose();
}
```

텍스트 필드

꾸미기 – InputDecoration

- labelText: 라벨 문자열
- helperText: 아래쪽에 출력되는 설명 문자열
- hintText: 입력 상자 안쪽에 출력되었다가 글 입력 시 사라지는 문자열
- errorText: 아래쪽에 출력되는 오류 문자열
- prefixIcon: 입력 앞 부분에 고정으로 출력되는 아이콘 이미지
- counterText: 아래쪽에 출력되는 문자열
- border: 테두리 지정. OutlineInputBorder, UnderlineInputBorder 중 하나 이용

• 텍스트 필드 꾸미기

```
TextField(  
  style: TextStyle(fontSize: 15.0),  
  controller: controller,  
  decoration: InputDecoration(  
    labelText: 'Name',  
    prefixIcon: Icon(Icons.input),  
    border: OutlineInputBorder(),  
    hintText: "Hint Text",  
    helperText: "이름을 입력하세요.",  
    counterText: "$textCounter characters",  
    // errorText: "error text"  
  ),  
)
```

▶ 실행 결과



텍스트 필드

액션 버튼 - `textInputAction`

- 소프트웨어 키보드에서 오른쪽 아래에 있는 액션 버튼만은 앱에서 제어
 - `TextInputAction.next`: 다음 위젯으로 포커스 이동
 - `TextInputAction.previous`: 이전 위젯으로 포커스 이동
 - `TextInputAction.search`: 검색 버튼
 - `TextInputAction.send`: 전송 버튼

• 액션 버튼 설정

```
TextField(  
  style: TextStyle(fontSize: 15.0),  
  controller: controller,  
  decoration: InputDecoration(  
    ... (생략) ...  
  ),  
  textInputAction: TextInputAction.search,  
),
```

▶ 실행 결과



텍스트 필드

키보드 유형 - keyboardType

- 소프트 키보드가 나타날 때 키보드 유형을 설정
 - •TextInputType.number: 숫자 입력
 - •TextInputType.text: 문자 입력
 - •TextInputType.phone: 전화번호 입력
 - •TextInputType.emailAddress: 이메일 주소 입력
 - •TextInputType.url: URL 입력

• 키보드 유형 설정

```
TextField(  
  style: TextStyle(fontSize: 15.0),  
  controller: controller,  
  decoration: InputDecoration(  
    ... (생략) ...  
  ),  
  // textInputAction: TextInputAction.search,  
  keyboardType: TextInputType.number,  
),
```

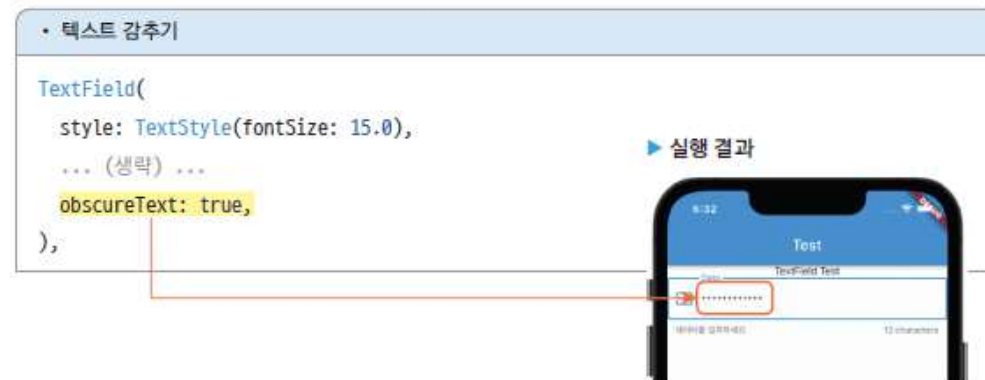
▶ 실행 결과



텍스트 필드

텍스트 감추기 - obscureText

- 사용자가 입력하는 글이 텍스트 필드에 ****처럼 보여야 할 때



텍스트 필드

여러 줄 입력 – maxLines, minLines

- `maxLines`: 3으로 설정하면 텍스트 필드가 3줄 크기로 나오지만 그 안에서 세로로 여러 줄을 입력
- `minLines`와 함께 사용하면 텍스트 필드가 화면에 출력될 때 `minLines`에 설정한 줄 수 크기로 출력되다가 글을 채우면 `maxLines`에 설정한 크기만큼 늘어납니다.



체크박스, 라디오 버튼, 슬라이더, 스위치


체크박스 – Checkbox

- 사용자가 체크 상태를 변경할 때 호출할 이벤트 콜백 함수는 onChanged 속성에 등록

• 체크박스 출력하기

```
Row(  
  children: [  
    Checkbox(  
      value: isChecked,  
      onChanged: (bool? value) {  
        setState(() {  
          isChecked = value;  
        });  
      },  
    ),  
    Text('checkbox value is $isChecked')  
  ],  
)
```

▶ 실행 결과



체크박스, 라디오 버튼, 슬라이더, 스위치

라디오 버튼 – Radio

- groupValue 속성이 똑같은 위젯 가운데 하나만 선택

• 라디오 버튼 출력하기

```
Row(  
  children: [  
    Radio(  
      value: "android",  
      groupValue: selectPlatform,  
      onChanged: (String? value) {  
        setState(() {  
          selectPlatform = value;  
        });  
      },  
    ),  
    Text('android')  
  ],  
)
```

```
Row(  
  children: [  
    Radio(  
      value: "ios",  
      groupValue: selectPlatform,  
      onChanged: (String? value) {  
        setState(() {  
          selectPlatform = value;  
        });  
      },  
    ),  
    Text('ios')  
  ],  
)  
  
Text('select platform is $selectPlatform')
```

▶ 실행 결과



체크박스, 라디오 버튼, 슬라이더, 스위치

슬라이더 - Slider

- 슬라이더 위젯은 min, max 속성으로 값을 설정하며 사용자가 막대를 왼쪽이나 오른쪽으로 밀면 그 사이의 값이 onChanged에 지정한 함수의 매개변수에 전달

• 슬라이더 출력하기

```
Slider(  
  value: selectValue,  
  min: 0,  
  max: 10,  
  onChanged: (double value) {  
    setState(() {  
      selectValue = value;  
    });  
  },  
),
```

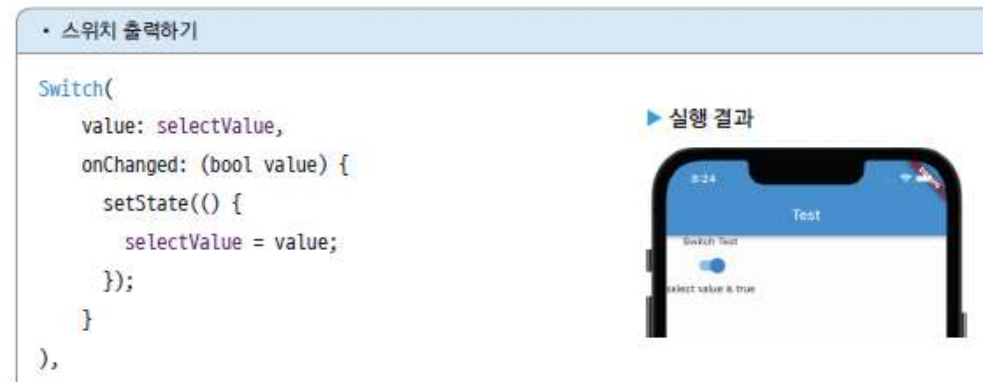
▶ 실행 결과



체크박스, 라디오 버튼, 슬라이더, 스위치

스위치 - Switch

- 스위치도 사용자에게 true나 false를 입력받는 위젯



폼 이용하기

- 폼을 이용할 때는 `FormField<T>` 형태로 사용자 입력 위젯을 폼 하위에 추가해서 연동
- Form 에서 사용하기 위한 `TextFormField`를 사용할 수도 있습니다.

폼에 키값 대입하기

- 폼 하위에 추가한 위젯들의 데이터 유효성 검증과 저장이 필요할 때 key값으로 Form 객체를 얻어서 `FormState` 객체의 함수를 호출해 유효성 검증(`validate()`)이나 입력 데이터를 저장(`save()`)

• 폼에 키값 대입하기

```
class MyFormState extends State<TestScreen> {  
  
  final _formKey = GlobalKey<FormState>();  
  
  @override  
  Widget build(BuildContext context) {  
    return Column(  
      children: [  
        Form(  

```

```
          key: _formKey,  
          ... (생략) ...  
        ),  
      ],  
    );  
  }  
}
```

폼 이용하기

유효성 검증과 데이터 저장하기

- validator와 onSave 속성에 설정한 함수가 호출되는 시점은 FormState가 제공하는 같은 이름의 함수가 실행될 때 호출

• 폼의 유효성 검증과 데이터 저장 함수 사용하기

```
Form(  
  key: _formKey,  
  child: Column(  
    children: [  
      TextFormField(  
        decoration: InputDecoration(  
          labelText: 'FirstName'  
        ),  
        validator: (value) {  
          if (value?.isEmpty ?? false) {  
            return 'Please enter first name';  
          }  
          return null;  
        },  
        onSave: (String? value){  
          firstName = value;  
        },  
      ),  
    ],  
  ),  
)
```

```
),  
],  
,  
,  
,  
  
ElevatedButton(  
  onPressed: () {  
    if (_formKey.currentState?.validate() ?? false) {  
      _formKey.currentState?.save();  
      print('firstName: $firstName, lastName: $lastName');  
    }  
  },  
  child: Text('submit')  
)  
,
```

▶ 실행 결과



폼 이용하기

유효성 검증과 데이터 저장하기

- validator에 설정한 함수가 null을 반환하면 사용자가 입력한 데이터가 유효하다는 의미
- 문자열을 반환하면 유효하지 않다는 의미
- 모든 TextFormField의 validator 속성에 설정한 함수가 호출되며 이 함수들이 모두 null을 반환하면 모든 데이터가 유효하다는 의미입니다. 그러면 validate() 함수는 true를 반환
- TextFormField의 validator에 설정한 함수 중 하나라도 문자열을 반환하면 유효하지 않다는 의미이므로 FormState의 validate() 함수는 false를 반환



감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.



윌리엄 셰익스피어
William Shakespeare