

## Listingsverzeichnis

1	Schritt 1 Die Klasse LetzterStatus . . . . .	5
2	Schritt 1 Die Klasse Choice . . . . .	6
3	Schritt 1 Die Menge letzterStatusChoices . . . . .	6
4	Schritt 1 Die Klasse Choices . . . . .	6
5	built_value Live Template . . . . .	7
6	Schritt 1 Der Werte-Typ Massnahme . . . . .	8
7	Schritt 1 Der Werte-Typ Identifikatoren . . . . .	8
8	Schritt 1 Der Werte-Typ LetzteBearbeitung . . . . .	9
9	Schritt 1 Der Serialisierer für Massnahme und Storage . . . . .	9
10	Schritt 1 Ein automatisierter Testfall überprüft . . . . .	9
11	Schritt 1 Ein automatisierter Testfall überprüft . . . . .	10
12	Schritt 1 Der Werte-Typ Storage . . . . .	10
13	Schritt 1 Ein automatisierter Testfall überprüft . . . . .	11
14	Schritt 1 Ein automatisierter Testfall überprüft . . . . .	11
15	Schritt 1 Die Klasse MassnahmenJsonFile . . . . .	12
16	Schritt 1 Die Klasse MassnahmenPool . . . . .	13
17	Schritt 1 Klasse MassnahmenMasterScreen Struktur . . . . .	14
18	Schritt 1 Ausgabe der finalen Maßnahmen . . . . .	15
19	Schritt 1 Bedingung der Entwurf-Maßnahmen . . . . .	15
20	Schritt 1 Die Klasse MassnahmenTable . . . . .	16
21	Schritt 1 Die Klasse MassnahmenFormViewModel . . . . .	17
22	Schritt 1 Klasse MassnahmenDetailScreen Struktur . . . . .	18
23	Schritt 1 Die Ausgabe der Formularfelder . . . . .	19
24	Schritt 1 Die Funktion createMassnahmenTitelTextFormField . . . . .	19
25	Schritt 1 Die Funktion saveRecordAndGoBackToOverviewScreen . . . . .	19
26	Schritt 1 Die Funktion createMultipleChoiceSelectionScreen . . . . .	20
27	Schritt 1 Die Build Methode der SelectionCard . . . . .	21
28	Schritt 1 Die Klasse SelectionCard . . . . .	22
29	Schritt 1 Der Integration Test Driver . . . . .	22
30	Schritt 1 Initialisierung des Integrations Tests . . . . .	22
31	Schritt 1 Initialisierung des Widgets für den Integrations Tests . . . . .	23
32	Schritt 1 Die Hilfsmethode tabSelectionCard . . . . .	23
33	Schritt 1 Die Hilfsmethode tabOption . . . . .	24
34	Schritt 1 Die Hilfsmethode fillTextFormField . . . . .	24
35	Schritt 1 Der Button zum Kreieren einer Maßnahme wird ausgelöst . . . . .	24
36	Schritt 1 Der letzte Status wird ausgewählt . . . . .	24
37	Schritt 1 Der Maßnahmentitel wird eingegeben . . . . .	25
38	Schritt 1 Der Button zum Speichern wird ausgelöst . . . . .	25
39	Schritt 1 Der Button zum Speichern wird ausgelöst . . . . .	25
40	Schritt 2 Der Integrationstest klickt 5 weitere Karten . . . . .	25
41	Schritt 2 Das erwartete Test-Ergebnis wird erweitert . . . . .	26
42	Schritt 2 Die Klasse FoerderklasseChoice . . . . .	26
43	Schritt 2 Die Menge foerderklasseChoices . . . . .	26
44	Schritt 2 Die Klasse KategorieChoice . . . . .	26
45	Schritt 2 Die Menge kategorieChoices . . . . .	27
46	Schritt 2 Die Klasse ZielflaecheChoice . . . . .	27
47	Schritt 2 Die Menge zielflaecheChoices . . . . .	27
48	Schritt 2 Die Klasse ZieleinheitChoice . . . . .	27
49	Schritt 2 Die Menge zieleinheitChoices . . . . .	28
50	Schritt 2 Die Klasse ZielsetzungLandChoice . . . . .	28
51	Schritt 2 Die Menge hauptzielsetzungLandChoices . . . . .	28
52	Schritt 2 massnahmenCharakteristika wird Massnahme hinzugefügt . . . . .	28

53	Schritt 2 Der Werte-Typ Massnahmencharakteristika . . . . .	29
54	Schritt 2 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	29
55	Schritt 2 Maßnahmencharakteristika werden dem ViewModel hinzugefügt .	30
56	Schritt 2 Maßnahmencharakteristika werden dem Tabellenkopf hinzugefügt .	30
57	Schritt 2 Maßnahmencharakteristika werden dem Tabellenkörper hinzugefügt	31
58	Schritt 3 errorText wird der SelectionCard hinzugefügt . . . . .	31
59	Schritt 3 errorText wird der SelectionCard hinzugefügt . . . . .	32
60	Schritt 3 errorText wird ausgegeben . . . . .	32
61	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	33
62	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	33
63	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	33
64	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	34
65	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	34
66	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	34
67	Schritt 3 Die Maßnahmencharakteristika Selektionskarten werden ergänzt .	35
68	Schritt 4 XXXX . . . . .	36
69	Schritt 4 XXXX . . . . .	36
70	Schritt 4 XXXX . . . . .	37
71	Schritt 4 XXXX . . . . .	37
72	Schritt 4 XXXX . . . . .	37
73	Schritt 4 XXXX . . . . .	37
74	Schritt 4 XXXX . . . . .	38
75	Schritt 4 Die Ausgabe der Formularfelder . . . . .	38
76	Schritt 4 Die Ausgabe der Formularfelder . . . . .	38
77	Schritt 4 Die Ausgabe der Formularfelder . . . . .	38
78	Schritt 4 XXXXX . . . . .	39
79	Schritt 4 XXXXX . . . . .	39
80	Schritt 4 XXXXX . . . . .	39
81	Schritt 4 XXXXX . . . . .	40
82	Schritt 4 XXXXX . . . . .	41
83	Schritt 4 XXXXX . . . . .	42
84	Schritt 5 XXXX . . . . .	43
85	Schritt 5 XXXX . . . . .	43
86	Schritt 5 XXXX . . . . .	43
87	Schritt 5 XXXX . . . . .	44
88	Schritt 5 XXXX . . . . .	44
89	Schritt 5 XXXX . . . . .	45
90	Schritt 5 XXXX . . . . .	45
91	Schritt 5 XXXX . . . . .	45
92	Schritt 5 XXXX . . . . .	45
93	Schritt 5 XXXX . . . . .	45
94	Schritt 5 XXXX . . . . .	45
95	Schritt 5 XXXXX . . . . .	45
96	Schritt 5 XXXXX . . . . .	46
97	Schritt 5 XXXX . . . . .	46
98	Schritt 5 XXXX . . . . .	46
99	Schritt 6 XXXX . . . . .	47
100	Schritt 6 XXXX . . . . .	47
101	Schritt 6 XXXX . . . . .	47
102	Schritt 6 XXXX . . . . .	48
103	Schritt 6 XXXX . . . . .	48
104	Schritt 6 XXXX . . . . .	48
105	Schritt 6 XXXX . . . . .	48

106	Schritt 6	XXXX	49
107	Schritt 6	XXXX	49
108	Schritt 6	XXXX	49
109	Schritt 6	XXXX	49
110	Schritt 7	XXXX	50
111	Schritt 7	XXXX	50
112	Schritt 7	XXXXX	50
113	Schritt 7	XXXX	51
114	Schritt 7	XXXX	51
115	Schritt 7	XXXX	52
116	Schritt 7	XXXX	52
117	Schritt 7	XXXX	52
118	Schritt 7	XXXX	52
119	Schritt 7	XXXX	53
120	Schritt 7	XXXX	53
121	Schritt 7	XXXX	53
122	Schritt 7	XXXX	54
123	Schritt 7	XXXX	54
124	Schritt 7	XXXX	54
125	Schritt 7	XXXX	54
126	Schritt 7	XXXX	55
127	Schritt 7	XXXXX	55
128	Schritt 7	XXXXX	55
129	Schritt 7	XXXXX	56
130	Schritt 7	XXXX	56
131	Schritt 7	XXXX	57
132	Schritt 7	XXXX	57
133	Schritt 7	XXXX	58

# 1 Implementierung

## 1.1 Schritt 1 - Formular in Grundstruktur erstellen

Im ersten Schritt soll die Formular-Anwendung in ihrer Grundstruktur entwickelt werden. Das beinhaltet alle drei Oberflächen, welche in den darauf folgenden Schritten lediglich erweitert werden. Das Formular erhält noch keine Validierung. Somit sind alle Eingaben oder nicht kompatible Selektionen erlaubt. Die erste Ansicht, welche der Benutzer sieht, soll die Übersicht der bereits eingetragenen Maßnahmen sein (Abb. 1).

Zuletzt bearbeitet am	Maßnahmentitel
2021-7-9 18:44	Massnahme 1

  

Zuletzt bearbeitet am	Maßnahmentitel
2021-7-9 18:44	Massnahme 2
2021-7-9 18:44	Massnahme 3
2021-7-9 18:44	Massnahme 4
2021-7-9 18:58	Massnahme 5

**Abbildung 1:** Der Übersicht-Bildschirm zeigt in Schritt 1 zunächst nur die Maßnahmen mit ihrem Titel und Bearbeitungsdatum in den Kategorien „Abgeschlossen“ und „In Bearbeitung“. Quelle: Eigene Abbildung

Die Auflistung der Maßnahmen erfolgt in den Kategorien „In Bearbeitung“ und „Abgeschlossen“. Innerhalb dieser Rubriken werden die Maßnahmen in einer Tabelle angezeigt. Mit einem Klick auf den Button unten rechts im Bild wird der Benutzer auf die zweite Ansicht weitergeleitet: die Eingabemaske (Abb. 2).

Status  
in Bearbeitung

Maßnahmentitel  
Massnahme 7

**Abbildung 2:** Die Eingabemaske zeigt im Schritt 1 eine Karte zum Selektieren des Status und ein Eingabefeld für den Titel. Quelle: Eigene Abbildung

Sie ermöglicht die Eingabe des Maßnahmen-Titels über ein simples Eingabefeld. Darüber hinaus ist die Selektions-Karte für den Status zu sehen. Mit einem Klick auf diese Karte öffnet sich der Selektions-Bildschirm. Er ermöglicht die Auswahl der Auswahloptionen, in diesem Fall die Optionen „in Bearbeitung“ und „abgeschlossen“ (Abb. 3).



**Abbildung 3:** Der Selektions-Bildschirm für das Feld Status erlaubt die Auswahl der Optionen „in Bearbeitung“ und „abgeschlossen“. Quelle: Eigene Abbildung

### 1.1.1 Auswahloptionen hinzufügen

Dart verfügt – anders als beispielsweise Java<sup>1</sup> – nicht über Aufzählungstypen mit zusätzlichen Eigenschaften. Das Schlüsselwort `enum` in Dart erlaubt lediglich die Auflistung konstanter Symbole<sup>2</sup>. Für die Auswahl Optionen ist es jedoch notwendig, dass es zwei Eigenschaften gibt:

- die Abkürzung, die in der resultierenden Datei gespeichert werden soll
- und der Beschreibungstext, welcher in der Oberfläche angezeigt wird.

Das hat den Hintergrund, dass die Abkürzungen weniger Speicherplatz einnehmen und die Beschreibung sich in Zukunft auch ändern darf. Würde anstatt der Abkürzung die Beschreibung als Schlüssel verwendet werden, so würde eine Datei, die mit einer älteren Version des Formulars erstellt wurde, nicht mehr von neueren Versionen der Applikation eingelesen werden können. Der alte Beschreibungstext würde nicht mehr mit dem Text übereinstimmen, der als Schlüssel in der Anwendung verwendet wird.

Die beiden Zustände „in Bearbeitung“ und „abgeschlossen“ werden daher in Listing ?? als statische Klassenvariablen deklariert (Z. 6-7). Die beiden Konstruktor-Aufrufe übergeben dabei als erstes Argument die Abkürzung und als zweites Argument die Beschreibung. Der Konstruktor selbst (Z. 9-10) deklariert die beiden Parameter als positionale Parameter.

---

```

5 class LetzterStatus extends Choice {
6   static final bearb = LetzterStatus("bearb", "in Bearbeitung");
7   static final fertig = LetzterStatus("fertig", "abgeschlossen");
8
9   LetzterStatus(String abbreviation, String description)
10    : super(abbreviation, description);
11 }

```

---

**Listing 1:** Die Klasse `LetzterStatus`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/choices/choices.dart](#)

**Positionale Parameter** Im Vergleich zu den benannten Parametern ist bei den positionalen Parametern nur ihre Reihenfolge in der Parameterliste ausschlaggebend. Das Argument für die `abbreviation` steht dabei also immer an erster Stelle und das Argument für `description` immer an der zweiten (Z. 6-7). Positionale Parameter sind vorgeschrieben. Werden sie ausgelassen, so gibt es einen Compilerfehler.<sup>3</sup>

Die Klasse `LetzterStatus` erbt von der Klasse `Choice` (Z. 5). Der Konstruktor der Klasse (Z. 7) übergibt beide Parameter als Argumente an den Konstruktor der Klasse `Choice`.

<sup>1</sup>Vgl. Gosling u. a., *The Java® Language Specification Java SE 16 Edition*, S. 321.

<sup>2</sup>Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 74f.

<sup>3</sup>Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 74f.

Weil das Aufrufen des sogenannten Super-Konstruktors zum statischen Teil der Objekt-Instanziierung gehört, muss der Aufruf von `super` in der Initialisierungsliste erfolgen. Die Initialisierungsliste wird mit dem `:` nach der Parameterliste eingeleitet (Z. 10)<sup>4</sup>.

---

```
3 class Choice {
4   final String description;
5   final String abbreviation;
6
7   const Choice(this.abbreviation, this.description);
```

---

**Listing 2:** Die Klasse `Choice`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/choices/base/choice.dart](#)

Die Klasse `Choice` (Listing. 2) deklariert lediglich die beiden Felder `description` und `abbreviation` jeweils als `String` (Z. 4-5). Beide sind mit `final` gekennzeichnet, was sie zu unveränderlichen Instanzvariablen macht. Nach der Initialisierung, können sie keine anderen Werte annehmen.<sup>5</sup> Die Initialisierung der beiden Variablen muss im statischen Kontext der Instanziierung erfolgen. Mit der abgekürzten Schreibweise `this.abbreviation` und `this._description` im Konstruktor (Z. 7) werden die Parameter den Feldern zugewiesen. Dies erübrigt die Zuweisung die man Ansonsten in der Form `this.abbreviation = abbreviation` und `this.description = description` in der Initialisierungsliste erreichen würde<sup>6</sup>.

---

```
13 final letzterStatusChoices = Choices<LetzterStatus>(
14   {LetzterStatus.bearb, LetzterStatus.fertig},
15   name: "Status");
```

---

**Listing 3:** Die Menge `letzterStatusChoices`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/choices/choices.dart](#)

---

```
10 class Choices<T extends Choice> extends UnmodifiableSetView<T> {
11   final String name;
12   final Map<String, T> choiceByAbbreviation;
13
14   T? fromAbbreviation(String? description) => choiceByAbbreviation[description];
15
16   Choices(Set<T> choices, {required this.name})
17     : choiceByAbbreviation = {
18       for (var choice in choices) choice.abbreviation: choice,
19     },
20     super(choices);
21 }
```

---

**Listing 4:** Die Klasse `Choices`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/choices/base/choice.dart](#)

## 2 Ein Test soll verifizieren, dass die Daten korrekt abgelegt werden

Doch damit die Daten angezeigt und verändert werden können, müssen sie zunächst serialisierbar sein, sodass sie auf einen Datenträger geschrieben und von dort auch wieder gelesen werden können. Die zwei bekanntesten Bibliotheken zum Serialisieren in Dart heißen `json_serializable` und `built_value`. Beide haben gemeinsam, dass sie Quellcode generieren, welcher die Umwandlung der Objekte in JSON übernimmt. `built_value` bietet

---

<sup>4</sup>Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 42.

<sup>5</sup>Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. S16.

<sup>6</sup>Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 40f.

---

```

6 part '$file_name$.g.dart';
7
8 abstract class $ClassName$ implements Built<$ClassName$, $ClassName$Builder> {
9     $todo$
10
11     $ClassName$. _();
12     factory $ClassName$([void Function($ClassName$Builder) updates]) = _$$$ClassName$;
13 }

```

---

**Listing 5:** Live Template für die Erstellung von built\_value Boilerplate-Code in Android Studio, Quelle: JetBrains Marketplace Built Value Snippets Plugin

im Gegensatz zu JSON Serializable jedoch die Möglichkeit unveränderbare Werte-Typen - sogenannte immutable value types - zu erstellen. Da diese unveränderbaren Werte noch bei der Erstellung des sogenannten ViewModels - Mehr dazu im Kapitel XXX - hilfreich werden, wurde sich für diese Bibliothek entschieden.

Ein Werte-Typ für built\_value erfordert etwas Boilerplate-Code, um den generierten Quellcode mit der selbstgeschriebenen Klasse zu verknüpfen. Dieser Boilerplate-Code kann durch das Live Template für Android Studio in Listing 5 generiert werden.<sup>1</sup>

\$ClassName\$ Wird dabei jeweils durch den gewünschten Klassennamen ersetzt. Android Studio erlaubt, dass bei Einfügen des live Templates der Klassenname einmalig eingegeben werden muss. Anschließend wird mithilfe des Templates der Boilerplate Code generiert. In Listing 6 ist der Werte-Typ „Maßnahme“ zu sehen. Die Zeilen 11 bis 13, sowie 23 bis 28 wurden dabei automatisch erstellt. Die Zeilen 14 bis 21 wurden hinzugefügt. Zunächst soll die Maßnahme über die „guid“ - Kurzform von General Unique Identifier - eindeutig identifiziert werden können. Die Attribute „letzteBearbeitung“ und „identifikatoren“ sind im Gegensatz zu dem String-Attribut guid zusammengesetzte Datentypen, die im Folgenden weiter beleuchtet werden.

Auffällig ist, dass es sich hier um eine abstrakte Klasse handelt und die drei Attribute jeweils Getter-Methoden ohne Implementierung sind. Eine solche Getter-Methode speichert keinen wert, sondern gibt lediglich den Wert eines Feldes zurück. Die dazugehörigen Felder, Setter-Methoden, die konkrete Klasse und der restliche generierte Code ist in der gleichnamigen Datei mit der Endung „.g.dart“ (Zeile 11) zu finden.

Die Klassen-Methode „\_initializeBuilder“ kann in jedem Werte-Typ hinterlegt werden, um Standardwerte für Felder festzulegen. Die Methode wird intern von built\_value aufgerufen. Bei dem Feld „guid“ handelt es sich um einen String, der keine Null-Werte zulässt. Könnte das Feld auch Null-Werte annehmen, so wäre die Notation in Dart dafür stattdessen „String? get guid;“. built\_value erwartet also immer einen Wert für dieses Feld. Sollte die Datei gelesen werden, welche die Maßnahmen enthält, so enthält jede Maßnahme bei der Deserialisierung den abgespeicherten Wert für die „guid“ und somit wird das Feld gefüllt. Doch sollte eine leere Maßnahme über einen Konstruktor erstellt werden, so wäre das Feld „guid“ leer und built\_value würde einen Fehler auslösen. Aus diesem Grund wird in der Zeile 21 für das Feld „guid“ ein Standardwert festgelegt, nämlich eine zufällige generierte ID die dem Standard Uuid der Version 4 entspricht. Die Attribute „letzteBearbeitung“ und „identifikatoren“ erhalten dagegen ganz automatisch Standardwerte in Form von Instanzen der dazugehörigen Klassen. Diese wiederum konfigurieren ihre eigenen Felder und deren initialwerte.

Der Werte-Typ Identifikatoren ist in Listing 7 zu sehen. Er enthält das Attribut „massnahmenTitel“, welcher im Eingabefeld durch das Texteingabefeld gefüllt werden wird.

---

<sup>1</sup><https://web.archive.org/web/20210710140113/https://github.com/GiancarloCode/built-value-snippets/blob/master/intellij/src/main/resources/liveTemplates/snippets.xml>

---

```

6 part 'massnahme.g.dart';
7
8 abstract class Massnahme implements Built<Massnahme, MassnahmeBuilder> {
9   String get guid;
10
11   LetzteBearbeitung get letzteBearbeitung;
12
13   Identifikatoren get identifikatoren;
14
15   static void _initializeBuilder(MassnahmeBuilder b) =>
16     b..guid = const Uuid().v4();
17
18   Massnahme._();
19
20   factory Massnahme([void Function(MassnahmeBuilder) updates]) = _$Massnahme;
21
22   static Serializer<Massnahme> get serializer => _$massnahmeSerializer;
23 }

```

---

**Listing 6:** Der Werte-Typ Massnahme, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

```

25 abstract class Identifikatoren
26   implements Built<Identifikatoren, IdentifikatorenBuilder> {
27   String get massnahmenTitel;
28
29   static void _initializeBuilder(IdentifikatorenBuilder b) =>
30     b.._massnahmenTitel = "";

```

---

**Listing 7:** Der Werte-Typ Identifikatoren, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

Schließlich enthält der Werte-Typ „LetzteBearbeitung“ in Listing 8 noch die Attribute „letztesBearbeitungsDatum“ in Zeile 43 und „letzterStatus“ in Zeile 50. Im Eingabeformular wird der Selektions-Bildschirm den Inhalt des Feldes „letzterStatus“ Bestimmen. Der initiale Wert auf wird in Zeile 54 auf einen konstanten Wert gesetzt, der dem Zustand „in Bearbeitung“ entspricht - mehr dazu im Kapitel CCCCCCCC.

Das Attribut „letztesBearbeitungsDatum“ ist dagegen nicht im Formular änderbar, sondern wird einmalig in Zeile 53 auf den aktuellen Zeitstempel gesetzt. Zugehörig zu diesem Attribut gibt es noch eine abgeleitete Eigenschaft namens „formattedDate“ (Zeilen 45-48). Es ist eine Hilfsmethode, die das letzte Bearbeitungsdatum in ein für Menschen lesbares Datumsformat umwandelt. In dem Übersichts-Bildschirm Abbildung 1 ist das Datumsformat sichtbar.

Da diese Getter-Methode eine Implementierung besitzt, wird für sie von `built_value` kein Quellcode für die Serialisierung generiert.

Wird nun der Befehl `flutter pub run build_runner build` ausgeführt, so wird der Quellcode generiert und die Werte-Typen können für die Serialisierung genutzt werden.

Das Ergebnis der Serialisierung wird im dazugehörigen Unit-Test ersichtlich. Listing ZZZZZZZZ zeigt den Unit Test für den Typ Maßnahme. In Zeile 8 wird ein Objekt der Klasse Massnahme instanziiert. Anders als bei gewöhnlichen Datentypen lassen sich bei diesem unveränderlichen Datentyp keine Attribute nach der Erstellung anpassen. Die einzige Möglichkeit besteht darin, ein neues Objekt mit dem gewünschten Attributwert zu erstellen und die restlichen Werte des alten Objektes zu übernehmen. dies ist in Bild Vaio mithilfe des sogenannten Bilder Entwurfsmuster möglich. In den Zeilen 9 bis 10 wird so ein neues Objekt von der Klasse Maßnahme mit Hilfe der Methode `rebuild` erzeugt und anschließend der



---

```

41 abstract class LetzteBearbeitung
42     implements Built<LetzteBearbeitung, LetzteBearbeitungBuilder> {
43     DateTime get letztesBearbeitungsDatum;
44
45     String get formattedDate {
46         final date = letztesBearbeitungsDatum;
47         return "${date.year}-${date.month}-${date.day} ${date.hour}:${date.minute}";
48     }
49
50     String get letzterStatus;
51
52     static void _initializeBuilder(LetzteBearbeitungBuilder b) => b
53         ..letztesBearbeitungsDatum = DateTime.now().toUtc()
54         ..letzterStatus = LetzterStatus.bearb.abbreviation;

```

---

**Listing 8:** Der Werte-Typ LetzteBearbeitung, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

```

10 @SerializersFor([Massnahme, Storage])
11 final Serializers serializers =
12     (_serializers.toBuilder()..addPlugin(StandardJsonPlugin())).build();

```

---

**Listing 9:** Der Serialisierer für Massnahme und Storage, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_model/serializers.dart](#)

---

Referenz Maßnahme zugewiesen, wodurch sie ihren alten Wert verliert.

---

```

6 test('Massnahme serialises without error', () {
7     var massnahme = Massnahme();
8     massnahme = massnahme
9         .rebuild((b) => b.identifikatoren.massnahmenTitel = "Massnahme 1");
10
11     var actualJson = serializers.serializeWith(Massnahme.serializer, massnahme);
12
13     var expectedJson = {
14         'guid': massnahme.guid,
15         'letzteBearbeitung': {
16             'letztesBearbeitungsDatum': massnahme
17                 .letzteBearbeitung.letztesBearbeitungsDatum.microsecondsSinceEpoch,
18             'letzterStatus': 'bearb'
19         },
20         'identifikatoren': {'massnahmenTitel': 'Massnahme 1'}
21     };
22
23     expect(actualJson, equals(expectedJson));

```

---

**Listing 10:** Ein automatisierter Testfall überprüft, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/test/data\\_model/massnahme\\_test.dart](#)

---

## 2.1 Schritt 2

# 3 HIER EINFÜGEN Status Choice

---

```

36 test('Massnahme deserialises without error', () {
37   var json = {
38     'guid': "test massnahme id",
39     'letzteBearbeitung': {
40       'letztesBearbeitungsDatum': 0,
41       'letzterStatus': 'bearb'
42     },
43     'identifikatoren': {'massnahmenTitel': 'Massnahme 1'}
44   };
45
46   var expectedMassnahme = Massnahme((b) => b
47     ..guid = "test massnahme id"
48     ..identifikatoren.massnahmenTitel = "Massnahme 1"
49     ..letzteBearbeitung.update((b) {
50       b.letztesBearbeitungsDatum =
51         DateTime.fromMillisecondsSinceEpoch(0, isUtc: true);
52     }));
53   var actualMassnahme =
54     serializers.deserializeWith(Massnahme.serializer, json);
55
56   expect(actualMassnahme, equals(expectedMassnahme));

```

---

**Listing 11:** Ein automatisierter Testfall überprüft, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/test/data\\_model/massnahme\\_test.dart](#)

---

```

9 abstract class Storage implements Built<Storage, StorageBuilder> {
10   @BuiltValueField(wireName: 'massnahmen')
11   BuiltSet<Massnahme> get massnahmen;
12
13   static void _initializeBuilder(StorageBuilder b) =>
14     b..massnahmen = SetBuilder();
15
16   Storage._();
17
18   factory Storage([void Function(StorageBuilder) updates]) = _$Storage;
19
20   static Serializer<Storage> get serializer => _$storageSerializer;
21 }

```

---

**Listing 12:** Der Werte-Typ Storage, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_model/storage.dart](#)

---

```

7 test('Storage with one Massnahme serialises without error', () {
8   var storage = Storage();
9   storage = storage.rebuild((b) => b.massnahmen.add(
10     Massnahme((b) => b.identifikatoren.massnahmenTitel = "Massnahme 1")));
11
12   var actualJson = serializers.serializeWith(Storage.serializer, storage);
13
14   var expectedJson = {
15     "massnahmen": [
16       {
17         "guid": storage.massnahmen.first.guid,
18         "letzteBearbeitung": {
19           "letztesBearbeitungsDatum": storage
20             .massnahmen
21             .first
22             .letzteBearbeitung
23             .letztesBearbeitungsDatum
24             .microsecondsSinceEpoch,
25         "letzterStatus": "bearb"
26       },
27       "identifikatoren": {"massnahmenTitel": "Massnahme 1"}
28     ]
29   };
30   expect(actualJson, equals(expectedJson));
31

```

---

**Listing 13:** Ein automatisierter Testfall überprüft, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/test/data\\_model/storage\\_test.dart](#)

---

```

48 test('Storage with one Massnahme deserialises without error', () {
49   var json = {
50     "massnahmen": [
51       {
52         "guid": "test massnahme id",
53         "letzteBearbeitung": {
54           "letztesBearbeitungsDatum": 0,
55           "letzterStatus": "bearb"
56         },
57         "identifikatoren": {"massnahmenTitel": "Massnahme 1"}
58       }
59     ]
60   };
61
62   var expectedStorage = Storage();
63   expectedStorage =
64     expectedStorage.rebuild((b) => b.massnahmen.add(Massnahme((b) => b
65       ..guid = "test massnahme id"
66       ..identifikatoren.massnahmenTitel = "Massnahme 1"
67       ..letzteBearbeitung.update((b) {
68         b.letztesBearbeitungsDatum =
69           DateTime.fromMillisecondsSinceEpoch(0, isUtc: true);
70       })))));
71
72   var actualStorage = serializers.deserializeWith(Storage.serializer, json);
73
74   expect(actualStorage, equals(expectedStorage));
75

```

---

**Listing 14:** Ein automatisierter Testfall überprüft, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/test/data\\_model/storage\\_test.dart](#)

---

```
7 class MassnahmenJsonFile {
8   Future<File> get _localMassnahmenJsonFile async {
9     var directory = await getApplicationSupportDirectory();
10    return File("${directory.path}/Maßnahmen.json");
11  }
12
13  Future<void> saveMassnahmen(Map<String, dynamic> massnahmenAsJson) async {
14    var file = await _localMassnahmenJsonFile;
15    await file.writeAsString(jsonEncode(massnahmenAsJson));
16  }
17
18  Future<Map<String, dynamic>> readMassnahmen() async {
19    var file = await _localMassnahmenJsonFile;
20
21    var fileExists = await file.exists();
22    if (fileExists) {
23      final fileContent = await file.readAsString();
24      final jsonObject = jsonDecode(fileContent) as Map<String, dynamic>;
25
26      return jsonObject;
27    } else {
28      throw MassnahmenFileDoesNotExistException("$file was not found");
29    }
30  }
31 }
```

---

**Listing 15:** Die Klasse MassnahmenJsonFile, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/persistence/massnahmen\\_json\\_file.dart](#)

---

```

8 class MassnahmenPool {
9   final MassnahmenJsonFile jsonFile;
10
11   MassnahmenPool(this.jsonFile) {
12     init();
13   }
14
15   final storageSubject = BehaviorSubject<Storage>.seeded(
16     Storage((b) => b..massnahmen = SetBuilder());
17
18   Storage get storage => storageSubject.value;
19
20   set storage(Storage m) {
21     storageSubject.value = m;
22   }
23
24   init() async {
25     refresh();
26   }
27
28   refresh() async {
29     try {
30       final massnahmenAsJson = await jsonFile.readMassnahmen();
31
32       final massnahmen =
33         serializers.deserializeWith(Storage.serializer, massnahmenAsJson)!;
34
35       storage = massnahmen;
36     } on MassnahmenFileDoesNotExistException {
37       storage = Storage();
38     }
39   }
40
41   putMassnahmeIfAbsent(Massnahme massnahme) async {
42     var rebuild = storage.rebuild((b) => b.massnahmen
43       ..removeWhere((m) => m.guid == massnahme.guid)
44       ..add(massnahme));
45
46     var serializedMassnahmen =
47       serializers.serializeWith(Storage.serializer, rebuild);
48
49     await jsonFile.saveMassnahmen(serializedMassnahmen as Map<String, dynamic>);
50
51     storage = rebuild;
52   }
53 }

```

---

**Listing 16:** Die Klasse MassnahmenPool, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/data\\_access/massnahmen\\_pool.dart](#)

---

```

13 final createNewMassnahmeButtonKey = GlobalKey();
14
15 class MassnahmenMasterScreen extends StatelessWidget {
16   static const routeName = '/massnahmen_master';
17
18   const MassnahmenMasterScreen({Key? key}) : super(key: key);
19
20   @override
21   Widget build(BuildContext context) {
22     final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);
23
24     return Scaffold(
25       appBar: AppBar(
26         title: const Text('Maßnahmen Master'),
27       ),
28       body: StreamBuilder<Storage>(
29         stream: massnahmenPool.storageSubject,
30         builder: (context, _) {
31           return SingleChildScrollView(...);
32         },
33       floatingActionButton: FloatingActionButton(
34         key: createNewMassnahmeButtonKey,
35         child: const Icon(
36           Icons.post_add_outlined,
37           color: Colors.white,
38         ),
39         onPressed: () {
40           final vm =
41             Provider.of<MassnahmenFormViewModel>(context, listen: false);
42
43           vm.model = Massnahme();
44           Navigator.of(context).pushNamed(MassnahmenDetailScreen.routeName);
45         },
46       );
47   }
48 }

```

---

**Listing 17:** Die Struktur der Klasse MassnahmenMasterScreen, Quelle: Eigenes Listing,  
Datei: Quellcode/Schritt-1/conditional\_form/lib/screens/massnahmen\_master.dart

---

```

31 return SingleChildScrollView(
32   child: Column(
33     crossAxisAlignment: CrossAxisAlignment.start,
34     children: [
35       const Padding(
36         padding: EdgeInsets.all(16.0),
37         child: Text(
38           "Abgeschlossen",
39           style: TextStyle(fontSize: 20),
40         ),
41       ),
42       SingleChildScrollView(
43         scrollDirection: Axis.horizontal,
44         child: Padding(
45           padding: const EdgeInsets.all(16.0),
46           child: MassnahmenTable(
47             massnahmenPool.storage.massnahmen
48               .where((m) =>
49                 m.letzteBearbeitung.letzterStatus ==
50                 LetzterStatus.fertig.abbreviation)
51               .toSet(), onSelect: (selectedMassnahme) {
52                 final vm = Provider.of<MassnahmenFormViewModel>(
53                   context,
54                   listen: false);
55                 vm.model = selectedMassnahme.rebuild((m) => m
56                   ..letzteBearbeitung.letztesBearbeitungsDatum =
57                     DateTime.now().toUtc());
58                 Navigator.of(context)
59                   .pushNamed(MassnahmenDetailScreen.routeName);
60               }),
61       ),

```

---

**Listing 18:** Die Ausgabe der finalen Maßnahmen, Quelle: Eigenes Listing,  
 Datei: Quellcode/Schritt-1/conditional\_form/lib/screens/massnahmen\_master.dart

---

```

48 .where((m) =>
49   m.letzteBearbeitung.letzterStatus == LetzterStatus.bearb.abbreviation)

```

---

**Listing 19:** Die Bedingung der Entwurf-Maßnahmen, Quelle: Eigenes Listing,  
 Datei: Quellcode/Schritt-1/conditional\_form/lib/screens/massnahmen\_master.dart

---

```

4 typedef OnSelectCallback = void Function(Massnahme selectedMassnahme);
5
6 class MassnahmenTable extends StatelessWidget {
7   final Set<Massnahme> _massnahmenToDisplay;
8   final OnSelectCallback? onSelect;
9
10  const MassnahmenTable(this._massnahmenToDisplay, {this.onSelect, Key? key})
11    : super(key: key);
12
13  @override
14  Widget build(BuildContext context) {
15    return Table(
16      border: TableBorder.all(width: 3),
17      defaultColumnWidth: const IntrinsicColumnWidth(),
18      defaultVerticalAlignment: TableCellVerticalAlignment.middle,
19      children: [
20        TableRow(children: [
21          _buildColumnHeader(const Text("Zuletzt bearbeitet am")),
22          _buildColumnHeader(const Text("Maßnahmentitel"))
23        ]),
24        ..._massnahmenToDisplay.map((m) {
25          return TableRow(children: [
26            _buildSelectableCell(m, Text(m.letzteBearbeitung.formattedDate)),
27            _buildSelectableCell(m, Text(m.identifikatoren.massnahmenTitel)),
28          ]);
29        }).toList(),
30      ],
31    );
32  }
33
34  Widget _buildColumnHeader(Widget child) => Padding(
35    padding: const EdgeInsets.all(8.0),
36    child: child,
37  );
38
39  Widget _buildSelectableCell(Massnahme m, Widget child,
40    {double padding = 8.0}) =>
41    TableRowInkWell(
42      onTap: () {
43        if (onSelect != null) {
44          onSelect!(m);
45        }
46      },
47      child: Padding(
48        padding: EdgeInsets.all(padding),
49        child: child,
50      ),
51    );
52 }

```

---

**Listing 20:** Die Klasse MassnahmenTable, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)



---

```
5 class MassnahmenFormViewModel {
6   final letzterStatus = BehaviorSubject<LetzterStatus?>.seeded(null);
7
8   final guid = BehaviorSubject<String?>.seeded(null);
9
10  final massnahmenTitel = BehaviorSubject<String>.seeded("");
11
12  set model(Massnahme model) {
13    guid.value = model.guid;
14
15    letzterStatus.value = letzterStatusChoices
16      .fromAbbreviation(model.letzteBearbeitung.letzterStatus);
17    massnahmenTitel.value = model.identifikatoren.massnahmenTitel;
18  }
19
20  Massnahme get model => Massnahme((b) => b
21    ..guid = guid.value
22    ..letzteBearbeitung.letzterStatus = letzterStatus.value?.abbreviation
23    ..letzteBearbeitung.letztesBearbeitungsDatum = DateTime.now().toUtc()
24    ..identifikatoren
25      .update((b) => b..massnahmenTitel = massnahmenTitel.value));
26 }
```

---

**Listing 21:** Die Klasse `MassnahmenFormViewModel`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

10 const saveMassnahmeTooltip = "Validiere und speichere Massnahme";
11
12 class MassnahmenDetailScreen extends StatelessWidget {
13   static const routeName = '/massnahmen-detail';
14
15   const MassnahmenDetailScreen({Key? key}) : super(key: key);
16
17   @override
18   Widget build(BuildContext context) {
19     final vm = Provider.of<MassnahmenFormViewModel>(context, listen: false);
20     final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);
21
22     Future<bool> saveRecordAndGoBackToOverviewScreen() {...}
23
24     Widget createMassnahmenTitelTextFormField(MassnahmenFormViewModel vm) {...}
25
26     return Scaffold(
27       appBar: AppBar(
28         title: const Text('Maßnahmen Detail'),
29       ),
30       body: WillPopScope(
31         onWillPop: () => saveRecordAndGoBackToOverviewScreen(),
32         child: Stack(
33           children: [
34             SingleChildScrollView(
35               child: Center(
36                 child: Padding(
37                   padding: const EdgeInsets.all(8.0),
38                   child: Column(...),
39                 ),
40               ),
41             ),
42             Align(
43               alignment: Alignment.bottomRight,
44               child: Padding(
45                 padding: const EdgeInsets.all(16.0),
46                 child: Column(
47                   mainAxisAlignment: MainAxisAlignment.min,
48                   children: [
49                     FloatingActionButton(
50                       tooltip: saveMassnahmeTooltip,
51                       heroTag: 'save_floating_action_button',
52                       child: const Icon(Icons.check, color: Colors.white),
53                       onPressed: () => saveRecordAndGoBackToOverviewScreen(),
54                     ),
55                   ],
56                 ),
57             ),
58           ],
59         ),
60       ),
61     ));
62 }
63 }

```

---

**Listing 22:** Die Struktur des Bildschirms MassnahmenDetailScreen, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional\_form/lib/screens/massnahmen\_detail/massnahmen\_detail.dart

---

```

64 child: Column(
65   crossAxisAlignment: CrossAxisAlignment.start,
66   children: [
67     SelectionCard<LetzterStatus>(
68       title: letzterStatusChoices.name,
69       allChoices: letzterStatusChoices,
70       initialValue: {
71         if (vm.letzterStatus.value != null)
72           vm.letzterStatus.value!
73       },
74       onSelect: (selectedChoice) =>
75         vm.letzterStatus.value = selectedChoice,
76       onDeselect: (selectedChoice) =>
77         vm.letzterStatus.value = null,
78     ),
79     createMassnahmenTitelTextFormField(vm),
80     const SizedBox(height: 64)
81   ],
82 ),

```

---

**Listing 23:** Die Ausgabe der Formularfelder, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

34 Widget createMassnahmenTitelTextFormField(MassnahmenFormViewModel vm) {
35   final focusNode = FocusNode();
36   return Card(
37     child: Padding(
38       padding: const EdgeInsets.all(16.0),
39       child: TextFormField(
40         focusNode: focusNode,
41         initialValue: vm.massnahmenTitel.value,
42         decoration: const InputDecoration(
43           hintText: 'Maßnahmentitel', labelText: 'Maßnahmentitel'),
44         onChanged: (value) {
45           vm.massnahmenTitel.value = value;
46         },
47       ),
48     ),
49   );
50 }

```

---

**Listing 24:** Die Funktion createMassnahmenTitelTextFormField, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

22 Future<bool> saveRecordAndGoBackToOverviewScreen() {
23   ScaffoldMessenger.of(context)
24     ..hideCurrentSnackBar()
25     ..showSnackBar(
26       const SnackBar(content: Text('Massnahme wird gespeichert ...')));
27
28   massnahmenPool.putMassnahmeIfAbsent(vm.model);
29   Navigator.of(context).pop();
30
31   return Future.value(true);
32 }

```

---

**Listing 25:** Die Funktion saveRecordAndGoBackToOverviewScreen, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

69 Widget createMultipleChoiceSelectionScreen(BuildContext context) {
70   Future<bool> goBack() {
71     Navigator.of(context).pop();
72     return Future.value(true);
73   }
74
75   return Scaffold(
76     appBar: AppBar(
77       title: Text(title),
78     ),
79     body: Builder(builder: (context) {
80       return StreamBuilder(
81         stream: selectionViewModel,
82         builder: (context, snapshot) {
83           final selectedChoices = selectionViewModel.value;
84           return ListView(children: [
85             ...allChoices.map((ChoiceType c) {
86               bool isSelected = selectedChoices.contains(c);
87
88               onTap: () {
89                 selectionViewModel.value =
90                   selectionViewModel.value.rebuild((b) {
91                     b.replace(isSelected ? [] : [c]);
92                   });
93
94               if (isSelected) {
95                 onDeselect(c);
96               } else {
97                 onSelect(c);
98               }
99             })
100           ]
101           return ListTile(
102             key: Key(
103               "valid choice ${allChoices.name} - ${c.abbreviation}"),
104             title: Column(
105               crossAxisAlignment: CrossAxisAlignment.start,
106               children: [Text(c.description)],
107             ),
108             leading: IconButton(
109               icon: Icon(isSelected
110                 ? Icons.check_box
111                 : Icons.check_box_outline_blank),
112               onPressed: onTap,
113             ),
114             onTap: onTap,
115           );
116         }).toList(),
117       ]);
118     });
119   },
120   floatingActionButton: FloatingActionButton(
121     onPressed: goBack,
122     tooltip: confirmButtonText,
123     child: const Icon(Icons.check),
124   ),
125 );
126 }

```

---

**Listing 26:** Die Funktion createMultipleChoiceSelectionScreen, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

34 Widget build(BuildContext context) {
35   final focusNode = FocusNode();
36
37   navigateToSelectionScreen() async {
38     focusNode.requestFocus();
39
40     Navigator.push<Set<Choice>>(
41       context,
42       MaterialPageRoute(
43         builder: (context) =>
44           createMultipleChoiceSelectionScreen(context)));
45   }
46
47   return StreamBuilder(
48     stream: selectionViewModel,
49     builder: (context, snapshot) {
50       final selectedChoices = selectionViewModel.value;
51       return Card(
52         child: Column(
53           crossAxisAlignment: CrossAxisAlignment.start,
54           children: [
55             ListTile(
56               focusNode: focusNode,
57               title: Text(title),
58               subtitle:
59                 Text(selectedChoices.map((c) => c.description).join(", ")),
60               trailing: const Icon(Icons.edit),
61               onTap: navigateToSelectionScreen,
62             )
63           ],
64         ),
65       );
66     });
67 }

```

---

**Listing 27:** Die Build Methode der SelectionCard, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

7 typedef OnSelect<ChoiceType extends Choice> = void Function(
8     ChoiceType selectedChoice);
9
10 typedef OnDeselect<ChoiceType extends Choice> = void Function(
11     ChoiceType selectedChoice);
12
13 const confirmButtonTooltip = 'Auswahl übernehmen';
14
15 class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
16     final String title;
17     final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
18     final Choices<ChoiceType> allChoices;
19     final OnSelect<ChoiceType> onSelect;
20     final OnDeselect<ChoiceType> onDeselect;
21
22     SelectionCard(
23         {required this.title,
24         required Iterable<ChoiceType> initialValue,
25         required this.allChoices,
26         required this.onSelect,
27         required this.onDeselect,
28         Key? key})
29         : selectionViewModel = BehaviorSubject<BuiltSet<ChoiceType>>.seeded(
30             BuiltSet.from(initialValue)),
31           super(key: key);

```

---

**Listing 28:** Die Klasse SelectionCard, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

3 Future<void> main() => integrationDriver();

```

---

**Listing 29:** Der Integration Test Driver, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/driver.dart](#)

---

```

18 const durationAfterEachStep = Duration(milliseconds: 1);
19
20 @GenerateMocks([MassnahmenJsonFile])
21 void main() {
22     testWidgets('Can fill the form and save the correct json', (tester) async {
23         final binding = IntegrationTestWidgetsFlutterBinding.ensureInitialized()
24             as IntegrationTestWidgetsFlutterBinding;
25         binding.framePolicy = LiveTestWidgetsFlutterBindingFramePolicy.fullyLive;
26
27         final mockMassnahmenJsonFile = MockMassnahmenJsonFile();
28         when(mockMassnahmenJsonFile.readMassnahmen()).thenAnswer((_) async => {});

```

---

**Listing 30:** Initialisierung des Integrations Tests, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

30 await tester.pumpWidget(MultiProvider(
31   providers: [
32     Provider<MassnahmenFormViewModel>(
33       create: (_) => MassnahmenFormViewModel(),
34     Provider<MassnahmenJsonFile>(create: (_) => mockMassnahmenJsonFile),
35     Provider(
36       create: (context) => MassnahmenPool(
37         Provider.of<MassnahmenJsonFile>(context, listen: false))),
38   ],
39   builder: (context, child) => MaterialApp(
40     title: 'Maßnahmen',
41     initialRoute: MassnahmenMasterScreen.routeName,
42     routes: {
43       MassnahmenMasterScreen.routeName: (context) =>
44         const MassnahmenMasterScreen(),
45       MassnahmenDetailScreen.routeName: (context) =>
46         const MassnahmenDetailScreen()
47     },
48   ),
49 ));

```

---

**Listing 31:** Initialisierung des Widgets für den Integrations Tests, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

51 Future<void> tabSelectionCard(Choices choices, {Finder? ancestor}) async {
52   final Finder textLabel;
53   if (ancestor != null) {
54     textLabel =
55       find.descendant(of: ancestor, matching: find.text(choices.name));
56   } else {
57     textLabel = find.text(choices.name);
58   }
59
60   expect(textLabel, findsWidgets);
61
62   final card = find.ancestor(of: textLabel, matching: find.byType(Card));
63   expect(card, findsOneWidget);
64
65   await tester.ensureVisible(card);
66   await tester.tap(card);
67   await tester.pumpAndSettle(durationAfterEachStep);
68 }

```

---

**Listing 32:** Die Hilfsmethode tabSelectionCard, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

76 Future<Finder> tabOption(Choice choice, {bool tabConfirm = false}) async {
77   final choiceLabel = find.text(choice.description);
78   expect(choiceLabel, findsOneWidget);
79
80   var listTileKey = tester
81     .element(choiceLabel)
82     .findAncestorWidgetOfExactType<ListTile>()!
83     .key!;
84   var listTile = find.byKey(listTileKey);
85
86   expect(listTile, findsOneWidget);
87
88   await tester.ensureVisible(choiceLabel);
89   await tester.tap(choiceLabel);
90   await tester.pumpAndSettle(durationAfterEachStep);
91
92   if (tabConfirm) {
93     await tabConfirmButton();
94   }
95
96   return listTile;
97 }

```

---

**Listing 33:** Die Hilfsmethode `tabOption`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

99 Future<void> fillTextFormField(
100   {required String title, required String text}) async {
101   final textFormField = find
102     .ancestor(of: find.text(title), matching: find.byType(TextFormField))
103     .first;
104   expect(textFormField, findsOneWidget);
105
106   await tester.ensureVisible(textFormField);
107   await tester.enterText(textFormField, text);
108   await tester.pumpAndSettle(durationAfterEachStep);
109 }

```

---

**Listing 34:** Die Hilfsmethode `fillTextFormField`, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

111 await tester.pumpAndSettle(durationAfterEachStep);
112
113 var createNewMassnahmeButton = find.byKey(createNewMassnahmeButtonKey);
114 final gesture = await tester.press(createNewMassnahmeButton);
115 await tester.pumpAndSettle(durationAfterEachStep);
116 await gesture.up();
117 await tester.pumpAndSettle(durationAfterEachStep);

```

---

**Listing 35:** Der Button zum Kreieren einer Maßnahme wird ausgelöst, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

119 await tabSelectionCard(letzterStatusChoices);
120 await tabOption(LetzterStatus.fertig, tabConfirm: true);

```

---

**Listing 36:** Der letzte Status wird ausgewählt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---



---

```

122 final now = DateTime.now();
123 var massnahmeTitle =
124     "Test Maßnahmen ${now.year}-${now.month}-${now.day} ${now.hour}:${now.minute}";
125 await fillTextFormField(title: "Maßnahmentitel", text: massnahmeTitle);

```

---

**Listing 37:** Der Maßnahmentitel wird eingegeben, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

127 var saveMassnahmeButton = find.byTooltip(saveMassnahmeTooltip);
128 await tester.tap(saveMassnahmeButton);
129 await tester.pumpAndSettle(durationAfterEachStep);

```

---

**Listing 38:** Der Button zum Speichern wird ausgelöst, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

131 var capturedJson =
132     verify(mockMassnahmenJsonFile.saveMassnahmen(captureAny)).captured.last;
133
134 var actualMassnahme = capturedJson['massnahmen'][0] as Map;
135 actualMassnahme.remove("guid");
136 actualMassnahme["letzteBearbeitung"].remove("letztesBearbeitungsDatum");
137
138 var expectedJson = {
139     'letzteBearbeitung': {'letzterStatus': 'fertig'},
140     'identifikatoren': {'massnahmenTitel': massnahmeTitle},
141 };
142
143 expect(actualMassnahme, equals(expectedJson));

```

---

**Listing 39:** Der Button zum Speichern wird ausgelöst, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-1/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

127 await fillTextFormField(title: "Maßnahmentitel", text: massnahmeTitle);
128
129 await tabSelectionCard(foerderklasseChoices);
130 await tabOption(FoerderklasseChoice.aukm_ohne_vns, tabConfirm: true);
131
132 await tabSelectionCard(kategorieChoices);
133 await tabOption(KategorieChoice.extens, tabConfirm: true);
134
135 await tabSelectionCard(zielflaecheChoices);
136 await tabOption(ZielflaecheChoice.al, tabConfirm: true);
137
138 await tabSelectionCard(zieleinheitChoices);
139 await tabOption(ZieleinheitChoice.ha, tabConfirm: true);
140
141 await tabSelectionCard(hauptzielsetzungLandChoices);
142 await tabOption(ZielsetzungLandChoice.biodiv, tabConfirm: true);
143
144 var saveMassnahmeButton = find.byTooltip(saveMassnahmeTooltip);

```

---

**Listing 40:** Der Integrationstest klickt 5 weitere Karten, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

155 var expectedJson = {
156   'letzteBearbeitung': {'letzterStatus': 'fertig'},
157   'identifikatoren': {'massnahmenTitel': massnahmeTitle},
158   'massnahmenCharakteristika': {
159     'foerderklasse': 'aukm_ohne_vns',
160     'kategorie': 'extens',
161     'zielflaeche': 'al',
162     'zieleinheit': 'ha',
163     'hauptzielsetzungLand': 'biodiv'
164   },
165 };

```

---

**Listing 41:** Das erwartete Test-Ergebnis wird erweitert, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```

5 class FoerderklasseChoice extends Choice {
6   static final oelb = FoerderklasseChoice("oelb", "Ökolandbau");
7   static final azl = FoerderklasseChoice("azl", "Ausgleichszulage");
8   static final ea = FoerderklasseChoice("ea", "Erschwernisausgleich");
9   static final aukum_nur_vns = FoerderklasseChoice("aukm_nur_vns",
10     "Agrarumwelt-(und Klima)Maßnahme: nur Vertragsnaturschutz");
11   static final aukum_ohne_vns = FoerderklasseChoice("aukm_ohne_vns",
12     "Agrarumwelt-(und Klima)Maßnahmen, tw. auch mit Tierwohlaspekten, aber OHNE
13     ↳ Vertragsnaturschutz");
14   static final twm_ziel = FoerderklasseChoice(
15     "twm_ziel", "Tierschutz/Tierwohlmaßnahmen mit diesem als Hauptziel");
16   static final contact =
17     FoerderklasseChoice("contact", "bitte um Unterstützung");

```

---

**Listing 42:** Die Klasse FoerderklasseChoice, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

23 final foerderklasseChoices = Choices<FoerderklasseChoice>({
24   FoerderklasseChoice.oelb,
25   FoerderklasseChoice.azl,
26   FoerderklasseChoice.ea,
27   FoerderklasseChoice.aukm_nur_vns,
28   FoerderklasseChoice.aukm_ohne_vns,
29   FoerderklasseChoice.twm_ziel,
30   FoerderklasseChoice.contact
31 }, name: "Förderklasse");

```

---

**Listing 43:** Die Menge foerderklasseChoices, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

33 class KategorieChoice extends Choice {
34   static final zf_us =
35     KategorieChoice("zf_us", "Anbau Zwischenfrucht/Untersaat");
36   static final anlage_pflege =
37     KategorieChoice("anlage_pflege", "Anlage/Pflege Struktur");
38   static final dungmang = KategorieChoice("dungmang", "Düngemanagement");
39   static final extens = KategorieChoice("extens", "Extensivierung");
40   static final flst = KategorieChoice("flst", "Flächenstilllegung/Brache");
41   static final umwandlg = KategorieChoice("umwandlg", "Nutzungsumwandlung");
42   static final bes_kult_rass = KategorieChoice(
43     "bes_kult_rass", "Förderung bestimmter Rassen / Sorten / Kulturen");
44   static final contact = KategorieChoice("contact", "bitte um Unterstützung");

```

---

**Listing 44:** Die Klasse KategorieChoice, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

---

```

50 final kategorieChoices = Choices<KategorieChoice>({
51   KategorieChoice.zf_us,
52   KategorieChoice.anlage_pflege,
53   KategorieChoice.dungmang,
54   KategorieChoice.extens,
55   KategorieChoice.flst,
56   KategorieChoice.umwandlg,
57   KategorieChoice.bes_kult_rass,
58   KategorieChoice.contact
59 }, name: "Kategorie");

```

---

**Listing 45:** Die Menge kategorieChoices, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

61 class ZielflaecheChoice extends Choice {
62   static final ka = ZielflaecheChoice("ka", "keine Angabe/Vorgabe");
63   static final al = ZielflaecheChoice("al", "AL");
64   static final gl = ZielflaecheChoice("gl", "GL");
65   static final lf = ZielflaecheChoice("lf", "LF");
66   static final dk_sk = ZielflaecheChoice("dk_sk", "DK/SK");
67   static final hff = ZielflaecheChoice("hff", "HFF");
68   static final biotop_le =
69     ZielflaecheChoice("biotop_le", "Landschaftselement/Biotop o.Ä.");
70   static final wald = ZielflaecheChoice("wald", "Wald/Forst");
71   static final contact = ZielflaecheChoice("contact", "bitte um Unterstützung");

```

---

**Listing 46:** Die Klasse ZielflaecheChoice, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

77 final zielflaecheChoices = Choices<ZielflaecheChoice>({
78   ZielflaecheChoice.ka,
79   ZielflaecheChoice.al,
80   ZielflaecheChoice.gl,
81   ZielflaecheChoice.lf,
82   ZielflaecheChoice.dk_sk,
83   ZielflaecheChoice.hff,
84   ZielflaecheChoice.biotop_le,
85   ZielflaecheChoice.wald,
86   ZielflaecheChoice.contact
87 }, name: "Zielfläche");

```

---

**Listing 47:** Die Menge zielflaecheChoices, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

89 class ZieleinheitChoice extends Choice {
90   static final ka = ZieleinheitChoice("ka", "keine Angabe/Vorgabe");
91   static final m3 = ZieleinheitChoice("m3", "m³ (z.B. Gülle)");
92   static final pieces =
93     ZieleinheitChoice("pieces", "Kopf/Stück (z.B. Tiere oder Bäume)");
94   static final gve = ZieleinheitChoice("gve", "GV/GVE");
95   static final rgve = ZieleinheitChoice("rgve", "RGV");
96   static final ha = ZieleinheitChoice("ha", "ha");
97   static final contact = ZieleinheitChoice("contact", "bitte um Unterstützung");

```

---

**Listing 48:** Die Klasse ZieleinheitChoice, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

103 final zieleinheitChoices = Choices<ZieleinheitChoice>({
104     ZieleinheitChoice.ka,
105     ZieleinheitChoice.m3,
106     ZieleinheitChoice.pieces,
107     ZieleinheitChoice.gve,
108     ZieleinheitChoice.rgve,
109     ZieleinheitChoice.ha,
110     ZieleinheitChoice.contact
111 }, name: "Zieleinheit");

```

---

**Listing 49:** Die Menge zieleinheitChoices, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

113 class ZielsetzungLandChoice extends Choice {
114     static final ka = ZielsetzungLandChoice("ka", "keine Angabe/Vorgabe");
115     static final bsch = ZielsetzungLandChoice("bsch", "Bodenschutz");
116     static final wsch = ZielsetzungLandChoice("wsch", "Gewässerschutz");
117     static final asch = ZielsetzungLandChoice("asch", "Spezieller Artenschutz");
118     static final biodiv = ZielsetzungLandChoice("biodiv", "Biodiversität");
119     static final struktkviel =
120         ZielsetzungLandChoice("struktkviel", "Erhöhung der Strukturvielfalt");
121     static final genet_res = ZielsetzungLandChoice("genet_res",
122         "Erhaltung genetischer Ressourcen (Pflanzen, z. B. im Grünland, und Tiere, z. B.
123         ↳ bedrohte Rassen)");
124     static final tsch = ZielsetzungLandChoice(
125         "tsch", "Tierschutz/Maßnahmen zum Tierwohl im Betrieb");
126     static final klima = ZielsetzungLandChoice("klima", "Klima");
127     static final contact =
128         ZielsetzungLandChoice("contact", "bitte um Unterstützung");

```

---

**Listing 50:** Die Klasse ZielsetzungLandChoice, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

133 final _zielsetzungLandChoices = {
134     ZielsetzungLandChoice.ka,
135     ZielsetzungLandChoice.bsch,
136     ZielsetzungLandChoice.wsch,
137     ZielsetzungLandChoice.asch,
138     ZielsetzungLandChoice.biodiv,
139     ZielsetzungLandChoice.struktkviel,
140     ZielsetzungLandChoice.genet_res,
141     ZielsetzungLandChoice.tsch,
142     ZielsetzungLandChoice.klima,
143     ZielsetzungLandChoice.contact
144 };
145
146 final hauptzielsetzungLandChoices = Choices<ZielsetzungLandChoice>(
147     _zielsetzungLandChoices,
148     name: "Hauptzielsetzung Land");

```

---

**Listing 51:** Die Menge hauptzielsetzungLandChoices, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/choices/choices.dart](#)

---

```

13 Identifikatoren get identifikatoren;
14
15 Massnahmencharakteristika get massnahmenCharakteristika;

```

---

**Listing 52:** massnahmenCharakteristika wird Massnahme hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

---

```

67 abstract class Massnahmencharakteristika
68     implements
69         Built<Massnahmencharakteristika, MassnahmencharakteristikaBuilder> {
70     String? get foerderklasse;
71     String? get kategorie;
72     String? get zielflaeche;
73     String? get zieleinheit;
74     String? get hauptzielsetzungLand;

```

---

**Listing 53:** Der Werte-Typ Massnahmencharakteristika, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

```

86 buildSectionHeadline("Identifikatoren"),
87 createMassnahmenTitelTextFormField(vm),
88 buildSectionHeadline("Maßnahmencharakteristika"),
89 buildSelectionCard(
90     allChoices: foerderklasseChoices,
91     selectionViewModel: vm.foerderklasse),
92 buildSelectionCard(
93     allChoices: kategorieChoices,
94     selectionViewModel: vm.kategorie),
95 buildSubSectionHeadline("Zielsetzung"),
96 buildSelectionCard(
97     allChoices: zielflaecheChoices,
98     selectionViewModel: vm.zielflaeche),
99 buildSelectionCard(
100    allChoices: zieleinheitChoices,
101    selectionViewModel: vm.zieleinheit,
102 ),
103 buildSelectionCard<ZielsetzungLandChoice>(
104    allChoices: hauptzielsetzungLandChoices,
105    selectionViewModel: vm.hauptzielsetzungLand),

```

---

**Listing 54:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

10 final massnahmenTitel = BehaviorSubject<String>.seeded("");
11
12 final foerderklasse = BehaviorSubject<FoerderklasseChoice?>.seeded(null);
13 final kategorie = BehaviorSubject<KategorieChoice?>.seeded(null);
14 final zielflaeche = BehaviorSubject<ZielflaecheChoice?>.seeded(null);
15 final zieleinheit = BehaviorSubject<ZieleinheitChoice?>.seeded(null);
16 final hauptzielsetzungLand =
17     BehaviorSubject<ZielsetzungLandChoice?>.seeded(null);
18
19 set model(Massnahme model) {
20     guid.value = model.guid;
21
22     letzterStatus.value = letzterStatusChoices
23         .fromAbbreviation(model.letzteBearbeitung.letzterStatus);
24     massnahmenTitel.value = model.identifikatoren.massnahmenTitel;
25
26     {
27         final mc = model.massnahmenCharakteristika;
28
29         foerderklasse.value =
30             foerderklasseChoices.fromAbbreviation(mc.foerderklasse);
31         kategorie.value = kategorieChoices.fromAbbreviation(mc.kategorie);
32
33         zielflaeche.value = zielflaecheChoices.fromAbbreviation(mc.zielflaeche);
34         zieleinheit.value = zieleinheitChoices.fromAbbreviation(mc.zieleinheit);
35         hauptzielsetzungLand.value =
36             hauptzielsetzungLandChoices.fromAbbreviation(mc.hauptzielsetzungLand);
37     }
38 }
39
40 Massnahme get model => Massnahme((b) => b
41     ..guid = guid.value
42     ..letzteBearbeitung.letzterStatus = letzterStatus.value?.abbreviation
43     ..letzteBearbeitung.letztesBearbeitungsDatum = DateTime.now().toUtc()
44     ..identifikatoren.update((b) => b..massnahmenTitel = massnahmenTitel.value)
45     ..massnahmenCharakteristika.update((b) => b
46         ..foerderklasse = foerderklasse.value?.abbreviation
47         ..kategorie = kategorie.value?.abbreviation
48         ..zielflaeche = zielflaeche.value?.abbreviation
49         ..zieleinheit = zieleinheit.value?.abbreviation
50         ..hauptzielsetzungLand = hauptzielsetzungLand.value?.abbreviation));

```

**Listing 55:** Maßnahmencharakteristika werden dem ViewModel hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

```

22 _buildColumnHeader(const Text("Maßnahmentitel")),
23 _buildColumnHeader(const Text("Förderklasse")),
24 _buildColumnHeader(const Text("Kategorie")),
25 _buildColumnHeader(const Text("Zielfläche")),
26 _buildColumnHeader(const Text("Zieleinheit")),
27 _buildColumnHeader(const Text("Hauptzielsetzung Land")),

```

**Listing 56:** Maßnahmencharakteristika werden dem Tabellenkopf hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

32 _buildSelectableCell(m, Text(m.identifikatoren.massnahmenTitel)),
33 _buildSelectableCell(
34   m, Text(m.massnahmenCharakteristika.foerderklasse ?? "")),
35 _buildSelectableCell(
36   m, Text(m.massnahmenCharakteristika.kategorie ?? "")),
37 _buildSelectableCell(
38   m, Text(m.massnahmenCharakteristika.zielflaeche ?? "")),
39 _buildSelectableCell(
40   m, Text(m.massnahmenCharakteristika.zieleinheit ?? "")),
41 _buildSelectableCell(m,
42   Text(m.massnahmenCharakteristika.hauptzielsetzungLand ?? "")),

```

---

**Listing 57:** Maßnahmencharakteristika werden dem Tabellenkörper hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-2/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

19 final OnSelect<ChoiceType> onSelect;
20 final OnDeselect<ChoiceType> onDeselect;
21 final String? errorText;
22
23 SelectionCard(
24   {required this.title,
25     required Iterable<ChoiceType> initialValue,
26     required this.allChoices,
27     required this.onSelect,
28     required this.onDeselect,
29     this.errorText,
30     Key? key})

```

---

**Listing 58:** errorText wird der SelectionCard hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/widgets/selection\\_card.dart](#)

### 3.1 Schritt 3

---

```
19 final OnSelect<ChoiceType> onSelect;
20 final OnDeselect<ChoiceType> onDeselect;
21 final String? errorText;
22
23 SelectionCard(
24   {required this.title,
25   required Iterable<ChoiceType> initialValue,
26   required this.allChoices,
27   required this.onSelect,
28   required this.onDeselect,
29   this.errorText,
30   Key? key})
```

---

**Listing 59:** errorText wird der SelectionCard hinzugefügt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```
63 onTap: navigateToSelectionScreen,
64 ),
65 if (errorText != null)
66   Padding(
67     padding: const EdgeInsets.all(8.0),
68     child: Text(errorText!,
69       style:
70         const TextStyle(fontSize: 12.0, color: Colors.red)),
71   )
```

---

**Listing 60:** errorText wird ausgegeben, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/widgets/selection\\_card.dart](#)



```

218 FloatingActionButton(
219   mini: true,
220   heroTag: 'save_draft_floating_action_button',
221   child: const Icon(Icons.paste, color: Colors.white),
222   backgroundColor: Colors.orange,
223   onPressed: saveDraft,
224 ),
225 const SizedBox(
226   height: 10,
227 ),
228 FloatingActionButton(
229   tooltip: saveMassnahmeTooltip,
230   heroTag: 'save_floating_action_button',
231   child: const Icon(Icons.check, color: Colors.white),
232   onPressed: validateAndSave,
233 )

```

**Listing 61:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

25 void saveDraft() {
26   ScaffoldMessenger.of(context)
27     ..hideCurrentSnackBar()
28     ..showSnackBar(
29       const SnackBar(content: Text('Entwurf wird gespeichert ...')));
30
31   var draft = vm.model.rebuild((b) =>
32     b.letzteBearbeitung.letzterStatus = LetzterStatus.bearb.abbreviation);
33
34   massnahmenPool.putMassnahmeIfAbsent(draft);
35   Navigator.of(context).pop();
36 }

```

**Listing 62:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

20 Widget build(BuildContext context) {
21   final vm = Provider.of<MassnahmenFormViewModel>(context, listen: false);
22   final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);
23   final formKey = GlobalKey<FormState>();

```

**Listing 63:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

48 void showValidationError() {
49   ScaffoldMessenger.of(context).showSnackBar(SnackBar(
50     content: Row(
51       children: [
52         Text(
53           'Fehler im Formular trotz Status "${LetzterStatus.fertig.description}"',
54           const SizedBox(width: 4),
55         ElevatedButton(
56           onPressed: saveDraft,
57           child: Padding(
58             padding: const EdgeInsets.fromLTRB(4, 4, 8, 4),
59             child: Row(
60               children: const [
61                 Icon(Icons.paste, color: Colors.white),
62                 SizedBox(width: 4),
63                 Text(
64                   "Entwurf speichern?",
65                   style: TextStyle(fontSize: 18.0, color: Colors.white),
66                 ),
67               ],
68             ),
69           ),
70         ],
71       ),
72     ));
73 }

```

---

**Listing 64:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

75 bool inputsAreValidOrNotMarkedFinal() {
76   if (vm.letzterStatus.value != LetzterStatus.fertig) {
77     return true;
78   }
79
80   if (formKey.currentState!.validate()) {
81     return true;
82   }
83
84   return false;
85 }

```

---

**Listing 65:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

87 Future<bool> validateAndSave() {
88   if (inputsAreValidOrNotMarkedFinal()) {
89     saveRecordAndGoBackToOverviewScreen();
90     return Future.value(true);
91   } else {
92     showValidationError();
93     return Future.value(false);
94   }
95 }

```

---

**Listing 66:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

132 Widget buildSelectionCard<ChoiceType extends Choice>(
133     {required Choices<ChoiceType> allChoices,
134     required BehaviorSubject<ChoiceType?> selectionViewModel}) {
135     return FormField(
136         validator: (_) {
137             if (selectionViewModel.value == null) {
138                 return "Feld ${allChoices.name} enthält keinen Wert!";
139             }
140
141             Iterable<Choice> choices = {
142                 if (selectionViewModel.value != null) selectionViewModel.value!
143             };
144
145             if (choices.isEmpty) {
146                 return "Feld ${allChoices.name} enthält keinen Wert!";
147             }
148
149             return null;
150         },
151         builder: (field) => SelectionCard<ChoiceType>(
152             title: allChoices.name,
153             allChoices: allChoices,
154             initialValue: {
155                 if (selectionViewModel.value != null)
156                     selectionViewModel.value!
157             },
158             onSelect: (selectedChoice) =>
159                 selectionViewModel.value = selectedChoice,
160             onDeselect: (selectedChoice) => selectionViewModel.value = null,
161             errorText: field.errorText,
162         ));
163 }

```

---

**Listing 67:** Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-3/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

## 3.2 Schritt 4

---

```
15 class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
16   final String title;
17   final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
18   final Choices<ChoiceType> allChoices;
19   final BehaviorSubject<Set<Choice>> priorChoices;
20   final OnSelect<ChoiceType> onSelect;
21   final OnDeselect<ChoiceType> onDeselect;
22   final String? errorText;
23
24   SelectionCard(
25     {required this.title,
26     required Iterable<ChoiceType> initialValue,
27     required this.allChoices,
28     required this.priorChoices,
29     required this.onSelect,
30     required this.onDeselect,
31     this.errorText,
32     Key? key})
```

---

**Listing 68:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```
51 return StreamBuilder(
52   stream: selectionViewModel,
53   builder: (context, snapshot) {
54     final selectedChoices = selectionViewModel.value;
55     final bool wrongSelection = selectedChoices
56       .any((c) => !c.conditionMatches(priorChoices.value));
57
58     return Card(
59       child: Column(
60         crossAxisAlignment: CrossAxisAlignment.start,
61         children: [
62           ListTile(
63             focusNode: focusNode,
64             title: Text(title),
65             subtitle:
66               Text(selectedChoices.map((c) => c.description).join(", ")),
67             trailing: const Icon(Icons.edit),
68             onTap: navigateToSelectionScreen,
69             tileColor:
70               wrongSelection || errorText != null ? Colors.red : null,
```

---

**Listing 69:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

---

```

95 body: Builder(builder: (context) {
96   return StreamBuilder(
97     stream: selectionViewModel,
98     builder: (context, snapshot) {
99       final selectedChoices = selectionViewModel.value;
100
101       Set<ChoiceType> selectedAndSelectableChoices = {};
102       Set<ChoiceType> unselectableChoices = {};
103
104       for (ChoiceType c in allChoices) {
105         if (selectedChoices.contains(c) ||
106             c.conditionMatches(priorChoices.value)) {
107           selectedAndSelectableChoices.add(c);
108         } else {
109           unselectableChoices.add(c);
110         }
111       }

```

---

**Listing 70:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

113 return ListView(children: [
114   ...selectedAndSelectableChoices.map((ChoiceType c) {
115     bool isSelected = selectedChoices.contains(c);
116     bool selectedButDoesNotMatch =
117       isSelected && !c.conditionMatches(priorChoices.value);

```

---

**Listing 71:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

146 tileColor: selectedButDoesNotMatch ? Colors.red : null,

```

---

**Listing 72:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

149 ...unselectableChoices
150   .where((c) => !c.conditionMatches(priorChoices.value))
151   .map((Choice c) {
152     return ListTile(
153       key: Key(
154         "invalid choice ${allChoices.name} - ${c.abbreviation}"),
155       title: Text(c.description),
156       leading: const Icon(Icons.close));
157   }).toList()

```

---

**Listing 73:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

20 BehaviorSubject<Set<Choice>> priorChoices =
21     BehaviorSubject<Set<Choice>>.seeded({});
22
23 MassnahmenFormViewModel() {
24     Stream<Set<Choice>> choicesStream = Rx.combineLatest([
25         foerderklasse,
26         kategorie,
27         zielflaeche,
28         zieleinheit,
29         hauptzielsetzungLand,
30     ], (_) {
31         return {
32             if (foerderklasse.value != null) foerderklasse.value!,
33             if (kategorie.value != null) kategorie.value!,
34             if (zielflaeche.value != null) zielflaeche.value!,
35             if (zieleinheit.value != null) zieleinheit.value!,
36             if (hauptzielsetzungLand.value != null) hauptzielsetzungLand.value!,
37         };
38     });
39
40     choicesStream.listen((event) => priorChoices.add(event));
41 }

```

---

**Listing 74:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

132 Widget buildSelectionCard<ChoiceType extends Choice>(
133     {required Choices<ChoiceType> allChoices,
134     required BehaviorSubject<ChoiceType?> selectionViewModel}) {
135     final fvm = Provider.of<MassnahmenFormViewModel>(context, listen: false);

```

---

**Listing 75:** Die Ausgabe der Formularfelder, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

147 if (choices.isEmpty) {
148     return "Feld ${allChoices.name} enthält keinen Wert!";
149 }
150
151 bool atLeastOneValueInvalid =
152     choices.any((c) => !c.conditionMatches(fvm.priorChoices.value));
153
154 if (atLeastOneValueInvalid) {
155     return "Wenigstens ein Wert im Feld ${allChoices.name} enthält ist fehlerhaft!";
156 }

```

---

**Listing 76:** Die Ausgabe der Formularfelder, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

160 builder: (field) => SelectionCard<ChoiceType>(
161     title: allChoices.name,
162     allChoices: allChoices,
163     priorChoices: fvm.priorChoices,

```

---

**Listing 77:** Die Ausgabe der Formularfelder, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

3 typedef Condition = bool Function(Set<Choice> choices);
4
5 class Choice {
6   final String description;
7   final String abbreviation;
8   final bool Function(Set<Choice> choices) condition;
9
10  bool conditionMatches(Set<Choice> choices) => condition.call(choices);
11
12  bool conditionDoesNotMatch(Set<Choice> choices) => !condition.call(choices);
13
14  const Choice(this.abbreviation, this.description, {Condition? condition})
15    : condition = condition ?? _conditionIsAlwaysMet;
16
17  static bool _conditionIsAlwaysMet(Set<Choice> choices) => true;
18 }

```

---

**Listing 78:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/base/choice.dart](#)

---

```

79 static final al = ZielflaecheChoice("al", "AL",
80   condition: (choices) => !choices.contains(KategorieChoice.zf_us));

```

---

**Listing 79:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/choices.dart](#)

---

```

95 static final wald = ZielflaecheChoice("wald", "Wald/Forst",
96   condition: (choices) =>
97     (choices.contains(FoerderklasseChoice.ea) ||
98       choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
99       choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
100     (!choices.contains(KategorieChoice.zf_us) ||
101       !choices.contains(KategorieChoice.bes_kult_rass)));

```

---

**Listing 80:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/choices.dart](#)

---

```

33 class KategorieChoice extends Choice {
34   static final zf_us = KategorieChoice(
35     "zf_us", "Anbau Zwischenfrucht/Untersaat",
36     condition: (choices) =>
37       choices.contains(FoerderklasseChoice.aukm_ohne_vns));
38   static final anlage_pflege = KategorieChoice(
39     "anlage_pflege", "Anlage/Pflege Struktur",
40     condition: (choices) =>
41       choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
42       choices.contains(FoerderklasseChoice.aukm_ohne_vns));
43   static final dungmang = KategorieChoice("dungmang", "Düngemanagement",
44     condition: (choices) =>
45       choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
46       choices.contains(FoerderklasseChoice.aukm_ohne_vns));
47   static final extens = KategorieChoice("extens", "Extensivierung");
48   static final flst = KategorieChoice("flst", "Flächenstilllegung/Brache",
49     condition: (choices) =>
50       choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
51       choices.contains(FoerderklasseChoice.aukm_ohne_vns));
52   static final umwandlg = KategorieChoice("umwandlg", "Nutzungsumwandlung",
53     condition: (choices) =>
54       choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
55       choices.contains(FoerderklasseChoice.aukm_ohne_vns));
56   static final bes_kult_rass = KategorieChoice(
57     "bes_kult_rass", "Förderung bestimmter Rassen / Sorten / Kulturen",
58     condition: (choices) => !choices.contains(FoerderklasseChoice.ea));
59   static final contact = KategorieChoice("contact", "bitte um Unterstützung");
60
61   KategorieChoice(String abbreviation, String description,
62     {bool Function(Set<Choice> choices)? condition})
63     : super(abbreviation, description, condition: condition);
64 }

```

---

**Listing 81:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/choices.dart](#)



---

```

77 class ZielflaecheChoice extends Choice {
78   static final ka = ZielflaecheChoice("ka", "keine Angabe/Vorgabe");
79   static final al = ZielflaecheChoice("al", "AL",
80     condition: (choices) => !choices.contains(KategorieChoice.zf_us));
81   static final gl = ZielflaecheChoice("gl", "GL");
82   static final lf = ZielflaecheChoice("lf", "LF");
83   static final dk_sk = ZielflaecheChoice("dk_sk", "DK/SK",
84     condition: (choices) => !choices.contains(FoerderklasseChoice.twm_ziel));
85   static final hff = ZielflaecheChoice("hff", "HFF");
86   static final biotop_le = ZielflaecheChoice(
87     "biotop_le", "Landschaftselement/Biotop o.Ä.",
88     condition: (choices) =>
89       (choices.contains(FoerderklasseChoice.azl) ||
90         choices.contains(FoerderklasseChoice.ea) ||
91         choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
92         choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
93       (!choices.contains(KategorieChoice.zf_us) ||
94         !choices.contains(KategorieChoice.bes_kult_rass)));
95   static final wald = ZielflaecheChoice("wald", "Wald/Forst",
96     condition: (choices) =>
97       (choices.contains(FoerderklasseChoice.ea) ||
98         choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
99         choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
100       (!choices.contains(KategorieChoice.zf_us) ||
101         !choices.contains(KategorieChoice.bes_kult_rass)));
102   static final contact = ZielflaecheChoice("contact", "bitte um Unterstützung");
103
104   ZielflaecheChoice(String abbreviation, String description,
105     {bool Function(Set<Choice> choices)? condition})
106     : super(abbreviation, description, condition: condition);
107 }

```

---

**Listing 82:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/choices.dart](#)

---

```

121 class ZieleinheitChoice extends Choice {
122   static final ka = ZieleinheitChoice("ka", "keine Angabe/Vorgabe");
123   static final m3 = ZieleinheitChoice("m3", "m³ (z.B. Gülle)",
124     condition: (choices) =>
125       (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
126         choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
127       (choices.contains(KategorieChoice.dungmang) ||
128         choices.contains(KategorieChoice.extens)) &&
129       (!choices.contains(ZielflaecheChoice.ka) &&
130         !choices.contains(ZielflaecheChoice.contact)));
131   static final pieces = ZieleinheitChoice(
132     "pieces", "Kopf/Stück (z.B. Tiere oder Bäume)",
133     condition: (choices) =>
134       (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
135         choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
136         choices.contains(FoerderklasseChoice.twm_ziel)) &&
137       (!choices.contains(KategorieChoice.zf_us) ||
138         !choices.contains(KategorieChoice.flst) ||
139         !choices.contains(KategorieChoice.umwandlg)) &&
140       (!choices.contains(ZielflaecheChoice.ka) &&
141         !choices.contains(ZielflaecheChoice.contact)));
142   static final gve = ZieleinheitChoice("gve", "GV/GVE",
143     condition: (choices) =>
144       (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
145         choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
146         choices.contains(FoerderklasseChoice.twm_ziel)) &&
147       (!choices.contains(KategorieChoice.zf_us) ||
148         !choices.contains(KategorieChoice.anlage_pflege) ||
149         !choices.contains(KategorieChoice.flst) ||
150         !choices.contains(KategorieChoice.umwandlg)) &&
151       (!choices.contains(ZielflaecheChoice.ka) &&
152         !choices.contains(ZielflaecheChoice.contact)));
153   static final rgve = ZieleinheitChoice("rgve", "RGV",
154     condition: (choices) =>
155       (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
156         choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
157         choices.contains(FoerderklasseChoice.twm_ziel)) &&
158       (!choices.contains(KategorieChoice.zf_us) ||
159         !choices.contains(KategorieChoice.anlage_pflege) ||
160         !choices.contains(KategorieChoice.flst) ||
161         !choices.contains(KategorieChoice.umwandlg)) &&
162       (!choices.contains(ZielflaecheChoice.ka) &&
163         !choices.contains(ZielflaecheChoice.contact)));
164   static final ha = ZieleinheitChoice("ha", "ha",
165     condition: (choices) =>
166       !choices.contains(ZielflaecheChoice.ka) &&
167       !choices.contains(ZielflaecheChoice.contact));
168   static final contact = ZieleinheitChoice("contact", "bitte um Unterstützung");
169
170   ZieleinheitChoice(String abbreviation, String description,
171     {bool Function(Set<Choice> choices)? condition})
172     : super(abbreviation, description, condition: condition);
173 }

```

---

**Listing 83:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-4/conditional\\_form/lib/choices/choices.dart](#)

### 3.3 Schritt 5

---

```
144 await tabSelectionCard(nebenzielsetzungLandChoices);
145 await tabOption(ZielsetzungLandChoice.bsch);
146 await tabOption(ZielsetzungLandChoice.klima, tabConfirm: true);
147
148 var saveMassnahmeButton = find.byTooltip(saveMassnahmeTooltip);
149 await tester.tap(saveMassnahmeButton);
150 await tester.pumpAndSettle(durationAfterEachStep);
```

---

**Listing 84:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```
162 'massnahmenCharakteristika': {
163   'nebenziele': [
164     'bsch',
165     'klima',
166   ],
167   'foerderklasse': 'aukm_ohne_vns',
```

---

**Listing 85:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/integration\\_test/app\\_test.dart](#)

---

```
219 final hauptzielsetzungLandChoices = Choices<ZielsetzungLandChoice>(
220   _zielsetzungLandChoices,
221   name: "Hauptzielsetzung Land");
222
223 final nebenzielsetzungLandChoices =
224   Choices<ZielsetzungLandChoice>(_zielsetzungLandChoices, name: "Nebenziele");
```

---

**Listing 86:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/choices/choices.dart](#)

---

```

68 abstract class Massnahmencharakteristika
69     implements
70         Built<Massnahmencharakteristika, MassnahmencharakteristikaBuilder> {
71     String? get foerderklasse;
72     String? get kategorie;
73     String? get zielflaeche;
74     String? get zieleinheit;
75     String? get hauptzielsetzungLand;
76
77     BuiltSet<String> get nebenziele;

```

---

**Listing 87:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

```

133 Widget buildSelectionCard<ChoiceType extends Choice>(
134     {bool multiSelection = false,
135     required Choices<ChoiceType> allChoices,
136     required BehaviorSubject<dynamic> selectionViewModel,
137     OnSelect<ChoiceType>? onSelect,
138     OnDeselect<ChoiceType>? onDeselect}) {
139     Iterable<ChoiceType> computeChoices(dynamic viewModel) {
140         if (viewModel is BuiltSet) {
141             return viewModel.map((e) => e).cast<ChoiceType>();
142         } else if (viewModel is ChoiceType?) {
143             return {if (viewModel != null) viewModel};
144         } else {
145             throw ArgumentError.value(viewModel);
146         }
147     }
148
149     void _defaultOnSingleSelectStrategy(ChoiceType selectedValue) {
150         var svm = selectionViewModel as BehaviorSubject<ChoiceType?>;
151         svm.value = selectedValue;
152     }
153
154     void _defaultOnSingleDeselectStrategy(ChoiceType selectedValue) {
155         var svm = selectionViewModel as BehaviorSubject<ChoiceType?>;
156         svm.value = null;
157     }
158
159     void _defaultOnMultiSelectStrategy(ChoiceType selectedValue) {
160         var svm = selectionViewModel as BehaviorSubject<BuiltSet<ChoiceType>>;
161         svm.value = svm.value.rebuild((b) => b.add(selectedValue));
162     }
163
164     void _defaultOnMultiDeselectStrategy(ChoiceType deselectedValue) {
165         var svm = selectionViewModel as BehaviorSubject<BuiltSet<ChoiceType>>;
166         svm.value = svm.value.rebuild((b) => b.remove(deselectedValue));
167     }
168
169     OnSelect<ChoiceType> onSelectNonNull;
170     OnDeselect<ChoiceType> onDeselectNonNull;
171     if (multiSelection) {
172         onSelectNonNull = onSelect ?? _defaultOnMultiSelectStrategy;
173         onDeselectNonNull = onDeselect ?? _defaultOnMultiDeselectStrategy;
174     } else {
175         onSelectNonNull = onSelect ?? _defaultOnSingleSelectStrategy;
176         onDeselectNonNull = onDeselect ?? _defaultOnSingleDeselectStrategy;
177     }

```

---

**Listing 88:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

187 Iterable<Choice> choices = computeChoices(selectionViewModel.value);
188
189 if (choices.isEmpty) {
190   return "Feld ${allChoices.name} enthält keinen Wert!";
191 }

```

**Listing 89:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

202 builder: (field) => SelectionCard<ChoiceType>(
203   title: allChoices.name,
204   multiSelection: multiSelection,
205   allChoices: allChoices,
206   priorChoices: fvm.priorChoices,
207   initialValue: computeChoices(selectionViewModel.value),
208   onSelect: onSelectNonNull,
209   onDeselect: onDeselectNonNull,
210   errorText: field.errorText,
211 ));

```

**Listing 90:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

251 buildSelectionCard<ZielsetzungLandChoice>(
252   allChoices: hauptzielsetzungLandChoices,
253   selectionViewModel: vm.hauptzielsetzungLand),
254 buildSelectionCard(
255   multiSelection: true,
256   allChoices: nebenzielsetzungLandChoices,
257   selectionViewModel: vm.nebenziele),

```

**Listing 91:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

18 final hauptzielsetzungLand =
19   BehaviorSubject<ZielsetzungLandChoice?>.seeded(null);
20 final nebenziele = BehaviorSubject<BuiltSet<ZielsetzungLandChoice>>.seeded(
21   BuiltSet<ZielsetzungLandChoice>());

```

**Listing 92:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

```

62 hauptzielsetzungLand.value =
63   hauptzielsetzungLandChoices.fromAbbreviation(mc.hauptzielsetzungLand);
64 nebenziele.value = BuiltSet(mc.nebenziele
65   .map((n) => hauptzielsetzungLandChoices.fromAbbreviation(n)));

```

**Listing 93:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

```

79 ..hauptzielsetzungLand = hauptzielsetzungLand.value?.abbreviation
80 ..nebenziele =
81   SetBuilder(nebenziele.value.map((n) => n.abbreviation).toList()));

```

**Listing 94:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

```

27 _buildColumnHeader(const Text("Hauptzielsetzung Land")),
28 _buildColumnHeader(const Text("Nebenziele")),

```

**Listing 95:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

42 _buildSelectableCell(m,
43   Text(m.massnahmenCharakteristika.hauptzielsetzungLand ?? "")),
44 _buildSelectableCell(
45   m,
46   Column(
47     children: m.massnahmenCharakteristika.nebenziele
48       .map((n) => Text(n))
49       .toList(),
50   )),

```

---

**Listing 96:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

15 class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
16   final String title;
17   final bool multiSelection;
18   final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
19   final Choices<ChoiceType> allChoices;
20   final BehaviorSubject<Set<Choice>> priorChoices;
21   final OnSelect<ChoiceType> onSelect;
22   final OnDeselect<ChoiceType> onDeselect;
23   final String? errorText;
24
25   SelectionCard(
26     {required this.title,
27     required this.multiSelection,

```

---

**Listing 97:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

121 onTileTab() {
122   if (multiSelection) {
123     selectionViewModel.value =
124       selectionViewModel.value.rebuild((b) {
125         if (selectionViewModel.value.contains(c)) {
126           return b.remove(c);
127         } else {
128           b.add(c);
129         }
130       });
131   } else {
132     selectionViewModel.value =
133       selectionViewModel.value.rebuild((b) {
134         b.replace(isSelected ? [] : [c]);
135       });
136   }
137   if (isSelected) {
138     onDeselect(c);
139   } else {
140     onSelect(c);
141   }
142 }

```

---

**Listing 98:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-5/conditional\\_form/lib/widgets/selection\\_card.dart](#)

### 3.4 Schritt 6

---

```
133 Widget buildSelectionCard<ChoiceType extends Choice>(
134     {bool multiSelection = false,
135     required Choices<ChoiceType> allChoices,
136     required BehaviorSubject<dynamic> selectionViewModel,
137     OnSelect<ChoiceType>? onSelect,
138     OnDeselect<ChoiceType>? onDeselect,
139     ChoiceMatcher<ChoiceType>? customChoiceMatcherStrategy}) {
140     final ChoiceMatcher<ChoiceType> matcher =
141         customChoiceMatcherStrategy ?? _defaultChoiceMatcherStrategy;
```

---

**Listing 99:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```
193 if (choices.isEmpty) {
194     return "Feld ${allChoices.name} enthält keinen Wert!";
195 }
196
197 bool atLeastOneValueInvalid = choices
198     .any((c) => !matcher(c as ChoiceType, fvm.priorChoices.value));
```

---

**Listing 100:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```
206 builder: (field) => SelectionCard<ChoiceType>(
207     title: allChoices.name,
208     multiSelection: multiSelection,
209     allChoices: allChoices,
210     priorChoices: fvm.priorChoices,
211     initialValue: computeChoices(selectionViewModel.value),
212     choiceMatcher: matcher,
213     onSelect: onSelectNonNull,
214     onDeselect: onDeselectNonNull,
215     errorText: field.errorText,
216 ));
```

---

**Listing 101:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

260 multiSelection: true,
261 allChoices: nebenzielsetzungLandChoices,
262 selectionViewModel: vm.nebenziele,
263 customChoiceMatcherStrategy: (choice, priorChoices) {
264   if (vm.hauptzielsetzungLand.value !=
265       ZielsetzungLandChoice.ka &&
266       vm.hauptzielsetzungLand.value !=
267       ZielsetzungLandChoice.contact) {
268     return choice != vm.hauptzielsetzungLand.value;
269   } else {
270     if (choice != ZielsetzungLandChoice.ka &&
271         choice != ZielsetzungLandChoice.contact) {
272       return false;
273     }
274     return true;
275   }
276 },
277 ),

```

**Listing 102:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

329 bool _defaultChoiceMatcherStrategy(Choice choice, Set<Choice> priorChoices) {
330   return choice.conditionMatches(priorChoices);
331 }

```

**Listing 103:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

```

13 typedef ChoiceMatcher<ChoiceType extends Choice> = bool Function(
14     ChoiceType choice, Set<Choice> priorChoices);

```

**Listing 104:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)

```

18 class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
19   final String title;
20   final bool multiSelection;
21   final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
22   final Choices<ChoiceType> allChoices;
23   final BehaviorSubject<Set<Choice>> priorChoices;
24   final OnSelect<ChoiceType> onSelect;
25   final OnDeselect<ChoiceType> onDeselect;
26   final String? errorText;
27   final ChoiceMatcher<ChoiceType> choiceMatcher;
28
29   SelectionCard(
30     {required this.title,
31     required this.multiSelection,
32     required Iterable<ChoiceType> initialValue,
33     required this.allChoices,
34     required this.priorChoices,
35     required this.onSelect,
36     required this.onDeselect,
37     required this.choiceMatcher,

```

**Listing 105:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)



---

```

58 return StreamBuilder(
59   stream: selectionViewModel,
60   builder: (context, snapshot) {
61     final selectedChoices = selectionViewModel.value;
62     final bool wrongSelection =
63       selectedChoices.any((c) => !choiceMatcher(c, priorChoices.value));

```

---

**Listing 106:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

111 for (ChoiceType c in allChoices) {
112   if (selectedChoices.contains(c) ||
113     choiceMatcher(c, priorChoices.value)) {
114     selectedAndSelectableChoices.add(c);
115   } else {
116     unselectableChoices.add(c);
117   }
118 }

```

---

**Listing 107:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

120 return ListView(children: [
121   ...selectedAndSelectableChoices.map((ChoiceType c) {
122     bool isSelected = selectedChoices.contains(c);
123     bool selectedButDoesNotMatch =
124       isSelected && !choiceMatcher(c, priorChoices.value);

```

---

**Listing 108:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

166 ...unselectableChoices
167   .where((c) => !choiceMatcher(c, priorChoices.value))
168   .map((Choice c) {
169     return ListTile(
170       key: Key(
171         "invalid choice ${allChoices.name} - ${c.abbreviation}"),
172       title: Text(c.description),
173       leading: const Icon(Icons.close));

```

---

**Listing 109:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-6/conditional\\_form/lib/widgets/selection\\_card.dart](#)

### 3.5 Schritt 7

```
148 await tabSelectionCard(duengungChoices);
149 var listTile = await tabOption(DuengungChoice.ausbring_t_kurz);
150 listTile = await tabOption(DuengungChoice.dueng_spez_art);
151 await tabSelectionCard(duengungArtChoices, ancestor: listTile);
152 await tabOption(DuengungArtChoice.dueng_org_n_miner, tabConfirm: true);
153 await tabConfirmButton();
154
155 var saveMassnahmeButton = find.byTooltip(saveMassnahmeTooltip);
156 await tester.tap(saveMassnahmeButton);
157 await tester.pumpAndSettle(durationAfterEachStep);
```

**Listing 110:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/integration\\_test/app\\_test.dart](#)

```
178   'hauptzielsetzungLand': 'biodiv'
179 },
180   'duengung': [
181     {'vorgabe': 'ausbring_t_kurz'},
182     {'vorgabe': 'dueng_spez_art', 'art': 'dueng_org_n_miner'}
183   ],
184 };
```

**Listing 111:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/integration\\_test/app\\_test.dart](#)

```
14 const Choice(this.abbreviation, this.description, {Condition? condition})
15   : condition = condition ?? _conditionIsAlwaysMet;
16
17 @override
18 bool operator ==(Object other) =>
19   identical(this, other) ||
20   other is Choice &&
21     description == other.description &&
22     abbreviation == other.abbreviation;
23
24 @override
25 int get hashCode => description.hashCode ^ abbreviation.hashCode;
```

**Listing 112:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/choices/base/choice.dart](#)

---

```

226 class DuengungChoice extends Choice {
227   static final o = DuengungChoice("o", "keine Angabe / ohne Einschränkung",
228     condition: (choices) => !choices.contains(contact));
229   static final dueng_keine = DuengungChoice("keine", "keine Düngung",
230     condition: (choices) =>
231       !choices.contains(o) && !choices.contains(contact));
232   static final ausbring_techn = DuengungChoice("ausbring_techn",
233     "besondere Ausbringungstechnik (z.B. Schleppschlauch) mit weiteren Angaben in
234     ↳ Technische Anforderungen",
235     condition: (choices) =>
236       !choices.contains(o) && !choices.contains(contact));
237   static final ausbring_t_kurz = DuengungChoice(
238     "ausbring_t_kurz", "verkürzte oder vorgegebene Ausbringungszeiten",
239     condition: (choices) =>
240       !choices.contains(o) && !choices.contains(contact));
241   static final dueng_red = DuengungChoice(
242     "dueng_red", "reduzierte Düngungsmenge",
243     condition: (choices) =>
244       !choices.contains(o) && !choices.contains(contact));
245   static final dueng_spez_art = DuengungChoice(
246     "dueng_spez_art", "Beschränkung auf spezifische Düngemittel",
247     condition: (choices) =>
248       !choices.contains(o) && !choices.contains(contact));
249   static final contact = DuengungChoice("contact", "bitte um Unterstützung",
250     condition: (choices) => !choices.contains(o));
251
252   bool get canSetArt => canSetArtCondition({this});
253   final Condition canSetArtCondition =
254     (choices) => choices.contains(dueng_spez_art);
255
256   DuengungChoice(String abbreviation, String description,
257     {bool Function(Set<Choice> choices)? condition})
258     : super(abbreviation, description, condition: condition);
259 }

```

---

**Listing 113:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/choices/choices.dart](#)

---

```

260 final duengungChoices = Choices<DuengungChoice>({
261   DuengungChoice.o,
262   DuengungChoice.dueng_keine,
263   DuengungChoice.ausbring_techn,
264   DuengungChoice.ausbring_t_kurz,
265   DuengungChoice.dueng_red,
266   DuengungChoice.dueng_spez_art,
267   DuengungChoice.contact,
268 }, name: "Düngung");

```

---

**Listing 114:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/choices/choices.dart](#)

---

```

270 class DuengungArtChoice extends Choice {
271   static final o = DuengungArtChoice("o", "keine Angabe");
272   static final dueng_org_tier_liqu = DuengungArtChoice(
273     "dueng_org_tier_liqu", "organisch, tierisch und flüssig: z.B. Gülle");
274   static final dueng_org_tier_fest = DuengungArtChoice(
275     "dueng_org_tier_fest", "organisch, tierisch, fest: Festmist");
276   static final dueng_org_pfl =
277     DuengungArtChoice("dueng_org_pfl", "organisch, pflanzlich: Gärrest");
278   static final dueng_org = DuengungArtChoice(
279     "dueng_org", "organisch: nicht differenziert oder mehreres betreffend");
280   static final dueng_miner = DuengungArtChoice("dueng_miner", "mineralisch");
281   static final dueng_org_n_miner =
282     DuengungArtChoice("dueng_org_n_miner", "organisch und mineralisch");
283   static final contact = DuengungArtChoice("contact", "bitte um Unterstützung");
284
285   DuengungArtChoice(String abbreviation, String description,
286     {bool Function(Set<Choice> choices)? condition})
287     : super(abbreviation, description, condition: condition);
288 }

```

---

**Listing 115:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/choices/choices.dart](#)

---

```

290 final duengungArtChoices = Choices<DuengungArtChoice>({
291   DuengungArtChoice.o,
292   DuengungArtChoice.dueng_org_tier_liqu,
293   DuengungArtChoice.dueng_org_tier_fest,
294   DuengungArtChoice.dueng_org_pfl,
295   DuengungArtChoice.dueng_org,
296   DuengungArtChoice.dueng_miner,
297   DuengungArtChoice.dueng_org_n_miner,
298   DuengungArtChoice.contact
299 }, name: "Düngung Art");

```

---

**Listing 116:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/choices/choices.dart](#)

---

```

16 Massnahmencharakteristika get massnahmenCharakteristika;
17
18 BuiltSet<Duengung> get duengung;

```

---

**Listing 117:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

```

91 abstract class Duengung implements Built<Duengung, DuengungBuilder> {
92   String get vorgabe;
93
94   String? get art;
95
96   Duengung._();
97
98   factory Duengung([void Function(DuengungBuilder) updates]) = _$Duengung;
99
100   static Serializer<Duengung> get serializer => _$duengungSerializer;
101 }

```

---

**Listing 118:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/data\\_model/massnahme.dart](#)

---

---

```

133 Widget buildSelectionCard<ChoiceType extends Choice>(
134   {bool multiSelection = false,
135     required Choices<ChoiceType> allChoices,
136     required BehaviorSubject<dynamic> selectionViewModel,
137     ListTileTitleBuilder<ChoiceType>? listTileTitleBuilder,
138     OnSelect<ChoiceType>? onSelect,
139     OnDeselect<ChoiceType>? onDeselect,
140     ChoiceMatcher<ChoiceType>? customChoiceMatcherStrategy}) {

```

---

**Listing 119:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

207 builder: (field) => SelectionCard<ChoiceType>(
208   title: allChoices.name,
209   multiSelection: multiSelection,
210   allChoices: allChoices,
211   priorChoices: fvm.priorChoices,
212   initialValue: computeChoices(selectionViewModel.value),
213   choiceMatcher: matcher,
214   ListTileTitleBuilder: listTileTitleBuilder,
215   onSelect: onSelectNonNull,
216   onDeselect: onDeselectNonNull,
217   errorText: field.errorText,
218 ));

```

---

**Listing 120:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

280 buildSectionHeadline("Düngung"),
281 buildSelectionCard<DuengungViewModel>(
282   multiSelection: true,
283   allChoices: Choices<DuengungViewModel>({
284     ...vm.duengung.value,
285     for (final c in duengungChoices)
286       if (!vm.duengung.value.contains(c))
287         DuengungViewModel(c)
288   }, name: duengungChoices.name),
289   selectionViewModel: vm.duengung,
290   ListTileTitleBuilder:
291     (context, choice, isSelected) {
292       return Column(
293         crossAxisAlignment: CrossAxisAlignment.start,
294         children: [
295           Text(choice.description),
296           if (isSelected) ...[
297             if (choice.canSetArt)
298               buildSelectionCard(
299                 allChoices: duengungArtChoices,
300                 selectionViewModel: choice.art),
301           ]
302         ],
303       );
304     },
305 const SizedBox(height: 64)

```

---

**Listing 121:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_detail.dart](#)

---

```

20 final nebenziele = BehaviorSubject<BuiltSet<ZielsetzungLandChoice>>.seeded(
21     BuiltSet<ZielsetzungLandChoice>());
22 final duengung = BehaviorSubject<BuiltSet<DuengungViewModel>>.seeded(
23     BuiltSet<DuengungViewModel>());

```

---

**Listing 122:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

28 MassnahmenFormViewModel() {
29     Stream<Set<Choice>> choicesStream = Rx.combineLatest([
30         foerderklasse,
31         kategorie,
32         zielflaeche,
33         zieleinheit,
34         hauptzielsetzungLand,
35         duengung,
36     ], (_) {
37         return {
38             if (foerderklasse.value != null) foerderklasse.value!,
39             if (kategorie.value != null) kategorie.value!,
40             if (zielflaeche.value != null) zielflaeche.value!,
41             if (zieleinheit.value != null) zieleinheit.value!,
42             if (hauptzielsetzungLand.value != null) hauptzielsetzungLand.value!,
43             ...duengung.value,
44         };
45     });

```

---

**Listing 123:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

68 nebenziele.value = BuiltSet(mc.nebenziele
69     .map((n) => hauptzielsetzungLandChoices.fromAbbreviation(n)));
70 }
71
72 duengung.value = BuiltSet(
73     {for (var d in model.duengung) DuengungViewModel.fromModel(d)});

```

---

**Listing 124:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

87 ..nebenziele =
88     SetBuilder(nebenziele.value.map((n) => n.abbreviation).toList())
89     ..duengung.replace(duengung.value.map((e) => e.model));
90 }

```

---

**Listing 125:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

92 class DuengungViewModel extends DuengungChoice {
93   final art = BehaviorSubject<DuengungArtChoice?>.seeded(null);
94
95   DuengungViewModel(DuengungChoice choice)
96     : super(choice.abbreviation, choice.description,
97       condition: choice.condition);
98
99   factory DuengungViewModel.fromModel(Duengung model) {
100     final vm =
101       DuengungViewModel(duengungChoices.fromAbbreviation(model.vorgabe!));
102     vm.model = model;
103     return vm;
104   }
105
106   set model(Duengung model) {
107     if (model.art != null) {
108       art.value = duengungArtChoices.fromAbbreviation(model.art);
109     }
110   }
111
112   Duengung get model => Duengung((b) {
113     b.vorgabe = abbreviation;
114
115     if (art.value != null) {
116       b.art = art.value!.abbreviation;
117     }
118   });
119 }

```

---

**Listing 126:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/screens/massnahmen\\_detail/massnahmen\\_form\\_view\\_model.dart](#)

---

```

28 _buildColumnHeader(const Text("Nebenziele")),
29 _buildColumnHeader(const Text("Düngung"))

```

---

**Listing 127:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

45 _buildSelectableCell(
46   m,
47   Column(
48     children: m.massnahmenCharakteristika.nebenziele
49       .map((n) => Text(n))
50       .toList(),
51   )),
52 _buildSelectableCell(m, buildDuengungTable(m.duengung), padding: 0)

```

---

**Listing 128:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

59 Widget buildDuengungTable(Iterable<Duengung> duengung) {
60   return Table(
61     border: TableBorder.all(width: 0.5),
62     defaultColumnWidth: const IntrinsicColumnWidth(),
63     children: [
64       TableRow(children: [
65         _buildColumnHeader(const Text("Vorgabe")),
66         _buildColumnHeader(const Text("Art")),
67       ]),
68       ...duengung
69         .map((d) => TableRow(
70           children: [
71             _buildColumnHeader(Text(d.vorgabe)),
72             if (d.art != null)
73               _buildColumnHeader(Text(d.art ?? ""))
74             else
75               _buildColumnHeader(const Text("")),
76           ],
77         ))
78         .toList()
79     ],
80   );
81 }

```

---

**Listing 129:** XXXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/massnahmen\\_table.dart](#)

---

```

7 typedef ListTileTitleBuilder<ChoiceType extends Choice> = Widget Function(
8   BuildContext context, ChoiceType choice, bool isSelected);
9
10 Widget _defaultBuildListTileTitleStrategy(context, Choice c, isSelected) {
11   return Column(
12     crossAxisAlignment: CrossAxisAlignment.start,
13     children: [Text(c.description)],
14   );
15 }

```

---

**Listing 130:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/selection\\_card.dart](#)



---

```

28 class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
29   final String title;
30   final bool multiSelection;
31   final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
32   final Choices<ChoiceType> allChoices;
33   final BehaviorSubject<Set<Choice>> priorChoices;
34   final OnSelect<ChoiceType> onSelect;
35   final OnDeselect<ChoiceType> onDeselect;
36   final String? errorText;
37   final ListTileTitleBuilder<ChoiceType> listTileTitleBuilder;
38   final ChoiceMatcher<ChoiceType> choiceMatcher;
39
40   SelectionCard(
41     {required this.title,
42     required this.multiSelection,
43     required Iterable<ChoiceType> initialValue,
44     required this.allChoices,
45     required this.priorChoices,
46     ListTileTitleBuilder<ChoiceType>? listTileTitleBuilder,
47     required this.onSelect,
48     required this.onDeselect,
49     required this.choiceMatcher,
50     this.errorText,
51     Key? key})
52     : selectionViewModel = BehaviorSubject<BuiltSet<ChoiceType>>.seeded(
53       BuiltSet.from(initialValue)),
54     listTileTitleBuilder =
55       listTileTitleBuilder ?? _defaultBuildListTileTitleStrategy,
56     super(key: key);
57

```

---

**Listing 131:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```

106 Widget createMultipleChoiceSelectionScreen(BuildContext context) {
107   final formKey = GlobalKey<FormState>();
108
109   Future<bool> goBackOnValidInput() {
110     if (formKey.currentState!.validate()) {
111       Navigator.of(context).pop();
112       return Future.value(true);
113     } else {
114       ScaffoldMessenger.of(context)
115         .showSnackBar(const SnackBar(content: Text('Fehler im Formular.')));
116       return Future.value(false);
117     }
118   }

```

---

**Listing 132:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/selection\\_card.dart](#)

---

```
120 return Scaffold(  
121   appBar: AppBar(  
122     title: Text(title),  
123   ),  
124   body: Builder(builder: (context) {  
125     return WillPopScope(  
126       onWillPop: goBackOnValidInput,  
127       child: Form(  
128         key: formKey,  
129         child: StreamBuilder(  
130           stream: selectionViewModel,  
131           builder: (context, snapshot) {  
132             final selectedChoices = selectionViewModel.value;  
133  
134             Set<ChoiceType> selectedAndSelectableChoices = {};  
135             Set<ChoiceType> unselectableChoices = {};
```

---

**Listing 133:** XXXX, Quelle: Eigenes Listing, Datei: [Quellcode/Schritt-7/conditional\\_form/lib/widgets/selection\\_card.dart](#)

# Anhang

## Literatur

Google LLC. *Dart Programming Language Specification 5th edition*. Apr. 2021. URL: <https://web.archive.org/web/20210702071617/https://dart.dev/guides/language/specifications/DartLangSpec-v2.10.pdf> (Zitiert auf den Seiten 5, 6).

Gosling, James u. a. *The Java® Language Specification Java SE 16 Edition*. Feb. 2021. URL: <https://web.archive.org/web/20210514051033/https://docs.oracle.com/javase/specs/jls/se16/jls16.pdf> (Zitiert auf der Seite 5).

## Eidesstattliche Erklärung

Ich versichere, dass ich die vorstehende Arbeit selbständig angefertigt und mich fremder Hilfe nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß veröffentlichtem oder nicht veröffentlichtem Schrifttum entnommen sind, habe ich als solche kenntlich gemacht.

Wernigerode, den 16.11.2020

---

Alexander Johr