Entwicklung einer Formularanwendung mit Kompatibilitätsvalidierung der Einfach- und Mehrfachauswahl-Eingabefelder

Vorgelegt von:

Alexander Johr

Meine Adresse

Erstprüfer:	Prof. Jürgen Singer Ph.D.
Zweitprüfer:	Prof. Daniel Ackermann
Datum:	02.11.2020

Hochschule Harz
Fachbereich Automatisierung und Informatik

Thema und Aufgabenstellung der Masterarbeit

MA AI 29/2021

für Herrn Alexander Johr

Entwicklung einer Formularanwendung mit Kompatibilitätsvalidierung der Einfach- und Mehrfachauswahl-Eingabefelder

Das Thünen-Institut für Ländliche Räume wertet Daten zu Maßnahmen auf landwirtschaftlich genutzten Flächen aus. Dafür müssen entsprechende Maßnahmen bundesweit mit Zeitbezug auswertbar sein und mit Attributen versehen werden. Um die Eingabe für die Wissenschaftler des Instituts zu beschleunigen und um fehlerhafte Eingaben zu minimieren, soll eine spezielle Formularanwendung entwickelt werden. Neben herkömmlichen Freitextfeldern beinhaltet das gewünschte Formular zum Großteil Eingabefelder für Einfach- und Mehrfachauswahl. Je nach Feld kann die Anzahl der Auswahloptionen mitunter zahlreich sein. Dem Nutzer sollen daher nur solche Auswahloptionen angeboten werden, die zusammen mit der zuvor getroffenen Auswahl sinnvoll sind.

Im Wesentlichen ergibt sich die Kompatibilität der Auswahloptionen aus der Bedingung, dass für dasselbe oder ein anderes Eingabefeld eine Auswahlmöglichkeit gewählt bzw. nicht gewählt wurde. Diese Bedingungen müssen durch Konjunktion und Disjunktion verknüpft werden können. In Sonderfällen muss ein Formularfeld jedoch auch die Konfiguration einer vom Standard abweichenden Bedingung ermöglichen. Wird dennoch versucht, eine deaktivierte Option zu selektieren, wäre eine Anzeige der inkompatiblen sowie der stattdessen notwendigen Auswahl ideal.

Die primäre Zielplattform der Anwendung ist das Desktop-Betriebssystem Microsoft Windows 10. Idealerweise ist die Formularanwendung auch auf weiteren Desktop-Plattformen sowie mobilen Endgeräten wie Android- und iOS-Smartphones und -Tablets lauffähig. Die Serialisierung der eingegebenen Daten genügt dem Institut zunächst als Ablage einer lokalen Datei im JSON-Format.

Die Masterarbeit umfasst folgende Teilaufgaben:

- Analyse der Anforderungen an die Formularanwendung
- Evaluation der angemessenen Technologie für die Implementierung
- Entwurf und Umsetzung der Übersichts- und Eingabeoberfläche
- Konzeption und Implementierung der Validierung der Eingabefelder

- Entwicklung von automatisierten Testfällen zur Qualitätskontrolle
- Bewertung der Implementierung und Vergleich mit den Wunschkriterien

Prof. Jürgen Singer Ph.D. 1. Prüfer	Prof. Daniel Ackermann 2. Prüfer

Inhaltsverzeichnis

Abbildungsverzeichnis

Listingsverzeichnis

1 Anforderungen

Dieses Kapitel behandelt die Anforderungen

- Performance: Hohe Anzahl Eingabefelder
- nested formulars

Wunsch

- Alle Komponenten wiederverwendbar wie etwa Selection Caard nicht nur für Choices
-
- Strategy Entwurfsmuster für compute choice von viewModelType weil viewmodel nötig ist für nested formular
-

2 Technologie Auswahl

Dieses Kapitel behandelt die Auswahl der Frontend-Technologie für die Umsetzung der Formular-Anwendung. Dazu werden im ersten Schritt die dafür in Frage kommende Technologien identifiziert. Anschließend wird der Trend der Popularität dieser Technologien miteinander verglichen. Die daraus resultierenden Kandidaten sollen dann detaillierter untersucht werden. In Hinblick auf die Anforderungen an die Formular-Anwendung soll dabei die angemessenste Frontend-Technologie ausgewählt werden.

2.1 Trendanalyse

Zwei Quellen wurden für die Analyse der Technologie-Trends ausgewählt: die Ergebnisse der jährlichen Stack Overflow Umfragen und das Such-Interesse von Google Trends.

Stack Overflow Umfrage Die Internet-Plattform Stack Overflow richtet sich an Softwareentwickler und bietet ihren Nutzern die Möglichkeiten, Fragen zu stellen, Antworten einzustellen und Antworten anderer Nutzer auf- und abzuwerten. Besonders für Fehlermeldungen, die häufig während der Softwareentwicklung auftreten, findet man auf dieser Plattform rasch die Erklärung und den Lösungsvorschlag gleich mit. Dadurch lässt sich auch die Herkunft des Domain-Namens herleiten:

> We named it Stack Overflow, after a common type of bug that causes software to crash – plus, the domain name stackoverflow.com happened to be available. - Joel Spolsky, Mitgründer von Stack Overflow [1]

Aufgrund des Erfolgsrezepts von Stack Overflow ist die Plattform kaum einem Softwareentwickler unbekannt. Dementsprechend nehmen auch jährlich tausende Entwickler an den von Stack Overflow herausgegebenen Umfragen teil. Seit 2013 beinhalten die Umfragen auch die Angabe der aktuell genutzten und in Zukunft gewünschten Frontend-Technologien. Stackoverflow erstellt aus diesen gesammelten Daten Auswertungen und Übersichten. Doch gleichzeitig werden die zugrundeliegenden Daten veröffentlicht. [2]

Um den Trend der Beliebtheit der Frontend-Technologien aufzuzeigen, wurde ein Jupyter Notebook erstellt. Es transformiert die Daten in ein einheitliches Format, da die Umfrageergebnisse von Jahr zu Jahr in einer unterschiedlichen Struktur abgelegt wurden. Anschließend erstellt es Diagramme, die im Folgenden analysiert werden. Das Jupyter Notebook ist im Anhang zu finden.

Google Trends Suchanfragen die an die Suchmaschine Google abgesetzt werden, lassen sich über den Dienst Google Trends als Trenddiagramm Visualisieren. Um

[1] Spolsky, „How Hard Could It Be?: The Unproven Path“

[2] Stack Exchange, Inc., *Stack Overflow Insights - Developer Hiring, Marketing, and User Research*

das relative Such-Interesse abzubilden, werden die Ergebnisse normalisiert, um die Ergebnisse auf einer Skala von 0 bis 100 darstellen zu können. [3]

> Google Trends ist keine wissenschaftliche Umfrage und sollte nicht mit Umfragedaten verwechselt werden. Es spiegelt lediglich das Suchinteresse an bestimmten Themen wider. [4]

Genau aus diesem Grund wird Google Trends im Folgenden lediglich zum Abgleich der Ergebnisse der Stack Overflow Umfrage eingesetzt.

2.1.1 Frameworks mit geringer Relevanz

NativeScript, Sencha (bzw. Sencha Touch) und Appcelerator spielen in den Umfrageergebnisse eine Untergeordnete Rolle. Dies ist in den aufsummierten Stimmen von 2013 bis 2020 für alle in der Umfrage auftauchenden Frontend-Technologien zu sehen (Abb. 1).

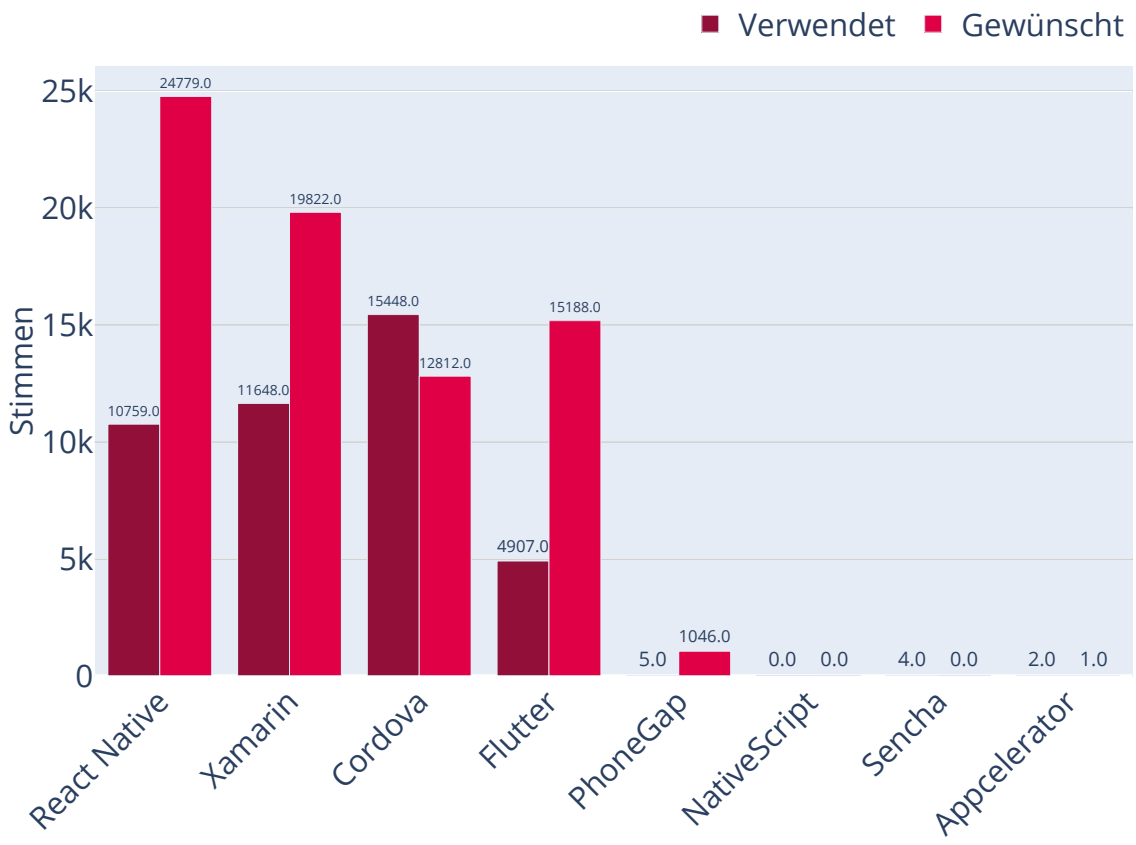

Abbildung 1: Summe der Stimmen der Stack Overflow Umfrage von 2013 bis 2020, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

Auch das Suchinteresse auf Google ist für diese Frameworks äußerst gering. In Abbildung 2 werden NativeScript, Sencha, Appcelerator und auch Adobe PhoneGap mit Apache Cordova für das relative Suchinteresse verglichen.

[3] Vgl. Google LLC, *Häufig gestellte Fragen zu Google Trends-Daten - Google Trends-Hilfe*
[4] Google LLC, *Häufig gestellte Fragen zu Google Trends-Daten - Google Trends-Hilfe*

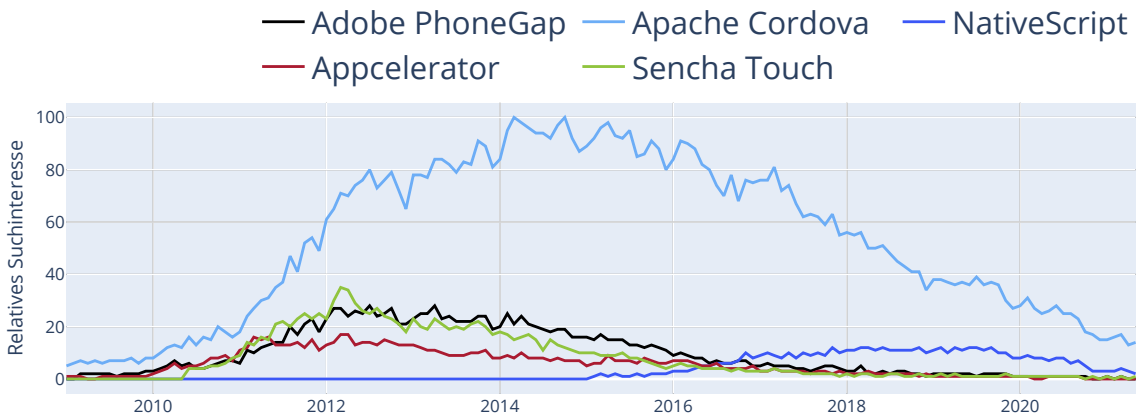

Abbildung 2: , Quelle: Eigene Abbildung, Notebook: Charts/Google Trends/Google Trends.ipynb, Daten-Quelle: Google Trends[5]

Verwandte Technologien zu Apache Cordova Das Ionic Framework taucht in den Ergebnissen der Stack Overflow Umfragen nicht auf. Ein Grund dafür könnte sein, dass es auf Apache Cordova aufbaut[6], welches bereits in den Ergebnissen vorkommt. Adobe PhoneGap taucht zwar in den Ergebnissen von 2013 mit 1043 Stimmen auf (Siehe Abbildung 3), verliert jedoch in den Folgejahren mit weniger als 10 Stimmen abrubt an Relevanz. Das stimmt nicht mit dem Suchinteresse auf Google überein, da es dort ab 2013 sogar steigt, wie in Abbildung 2 zu sehen ist. 2013 existierte PhoneGap noch als extra Mehrfachauswahlfeld in den Daten, während es ab 2014 nur noch in dem Feld für die sonstigen Freitext Angaben auftaucht [7]. Auch Adobe PhoneGap baut auf Apache Cordova auf[8]. Für diese Auswertung spielen diese verwandten Technologien eine untergeordnete Rolle, da sie auch in den Google Trends weit hinter Apache Cordova zurückbleiben (Abb. 2).

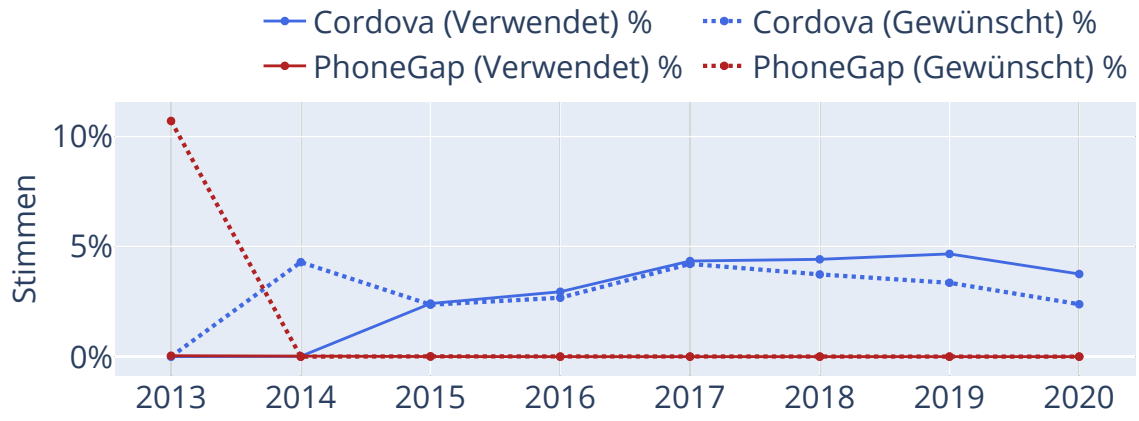

Abbildung 3: Stimmen für Cordova und PhoneGap 2013 bis 2020, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

Am Beispiel von Adobe PhoneGap wird deutlich, wie wichtig es ist, auf eine Technologie zu setzen, die weit verbreitet ist. Im schlimmsten Fall wird die Technologie sogar vom Betreiber aufgrund zu geringer Nutzung komplett eingestellt, wie es bereits bei PhoneGap geschehen ist. Adobe gab am 11. August 2020 bekannt, dass

[6]Lynch, *The Last Word on Cordova and PhoneGap*

[7]Vgl. Stack Exchange, Inc., *Stack Overflow Insights - Developer Hiring, Marketing, and User Research*

[8]Vgl. Adobe Inc., *FAQ | PhoneGap Docs*

die Entwicklung an PhoneGap eingestellt wird und empfiehlt die Migration hin zu Apache Cordova.[9]

2.1.2 Frameworks mit sinkender Relevanz

Die Technologien Xamarin und Cordova zeigen bereits einen abfallenden Trend, wie in Abbildung 4 ersichtlich ist. Im Fall von Xamarin gibt es immerhin mehr Entwickler, die sich wünschen, mit dem Framework zu arbeiten, als Entwickler, die tatsächlich mit Xamarin arbeiten. Cordova scheint in diesem Hinblick dagegen eher unbeliebt: es gibt mehr Entwickler, die mit Cordova arbeiten, als tatsächlich damit arbeiten wollen.

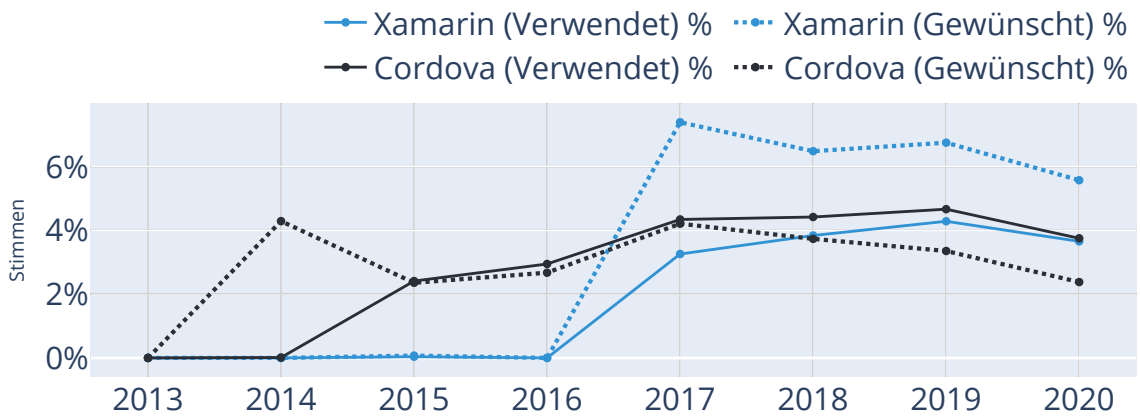

Abbildung 4: Stimmen für Xamarin und Cordova 2013 bis 2020, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

In Abbildung 5 ist noch einmal zu sehen, dass Google Trends die Erkenntnisse aus der Stack Overflow Umfrage reflektiert und es wird auch sichtbar, welche beiden Technologien möglicherweise der Grund für den Rückgang von Xamarin und Cordova sind.

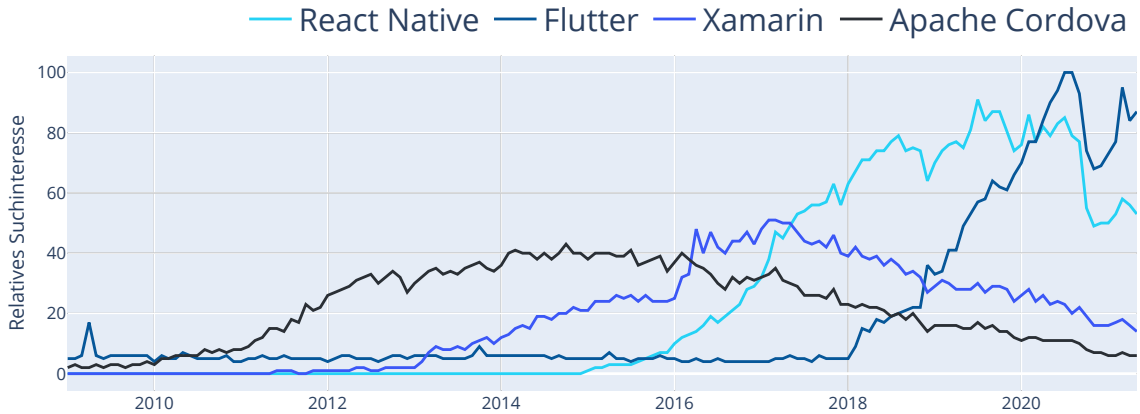

Abbildung 5: Suchinteresse sinkende und steigende Relevanz, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

[9]Vgl. Adobe Inc., *Update for Customers Using PhoneGap and PhoneGap Build*

2.1.3 Frameworks mit steigender Relevanz

Besser ist es, auf Technologien zu setzen, die noch einen steigenden Trend der Verbreitung und Beliebtheit zeigen. In Abbildung 6 wird sichtbar, dass es sich dabei um Flutter und immerhin im Hinblick auf die Verbreitung auch für React Native handelt. Ungünstigerweise wird React Native in der Stack Overflow Umfrage erst seit 2018 als tatsächliches Framework abgefragt. Vorher erschien lediglich das Framework React, welches nicht für den Vergleich der Cross-PlatformFrameworks eignet, da es sich um ein reines Web-Framework handelt. Doch auch die Ergebnisse von Google Trends zeigen einen ähnlichen Verlauf für die Jahre 2019 und 2020 (Abb. 5).

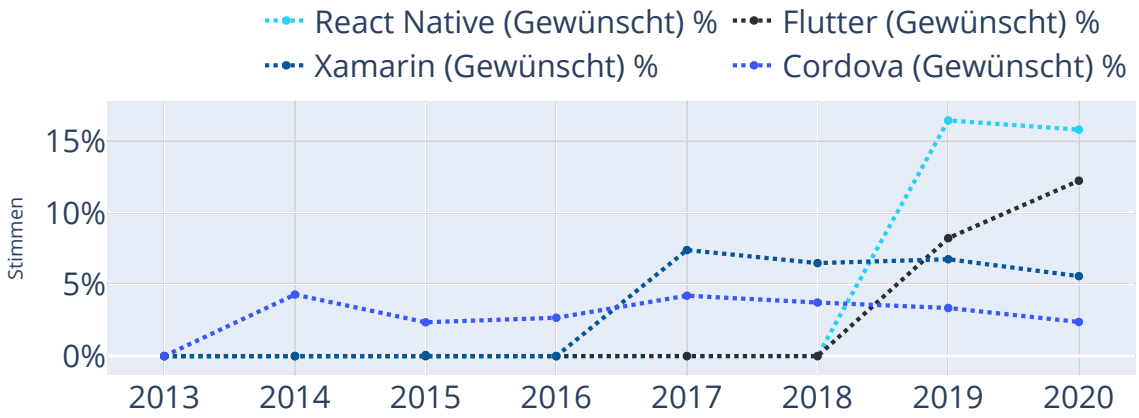

Abbildung 6: Stimmen für React Native und Flutter 2013 bis 2020, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

Im Vergleich von dem Jahr 2019 mit 2020 wird sichbar, dass die Zahl der Entwickler, die sich wünschen, mit React Native zu arbeiten, gesunken ist. Dennoch ist die Anzahl der Entwickler, die mit React Native arbeiten möchten noch weit höher, als die der Entwickler, die tatsächlich mit React Native arbeiten.

Es ist möglich, dass der abfallende Trend daran liegt, dass die Zahl der Entwickler, die mit Flutter arbeiten möchten im selben Jahr gestiegen ist. React Native hat im Vergleich zu Flutter jedoch noch immer mehr aktive Entwickler und die Tendenz ist steigend. Doch die Anzahl der aktiven Flutter Entwickler zeigt einen noch stärker steigenden Trend. So könnte es sein, dass die Zahl der Flutter Entwickler die der React Native Entwickler in einem der nächsten Jahre überholt. Im Such-Interesse hat sich diese Entwicklung bereits vollzogen (Abb. 5).

Nichtsdestotrotz scheinen beide Technologien als Kandidaten für einen detaillierteren Vergleich für dieses Projekt in Frage zu kommen. Im nächsten Kapitel soll evaluiert werden, welches Framework für die Entwicklung der Formular-Anwendung angemessener ist.

2.2 Vergleich React Native und Flutter

2.2.1 Vergleich zweier minimaler Beispiele für Formulare und Validierung

Es soll eine Formularanwendung mit komplexer Validierung im Rahmen dieser These erstellt werden. Es ist durchaus sinnvoll, die beiden Technologien anhand von Beispielanwendungen, welche Formulare und die Validierung dieser beinhalten, zu vergleichen. Deshalb sollen nachfolgend jeweils eine solche Beispielanwendung der jeweiligen Technologie gefunden werden. Die Anwendung werden sich stark voneinander unterscheiden, weshalb sie im nächsten Schritt vereinfacht und aneinander angeglichen werden. Anschließend wird ersichtlich werden, nach welchen Kriterien sich die Technologien im Hinblick auf die Entwicklung der Formularanwendung vergleichen lassen.

React Native React native stellt nur eine vergleichsweise geringe Anzahl von eigenen Komponenten zur Verfügung und zu diesen gehören keine, welche die Validierung von Formularen ermöglichen. Doch die im react.js Raum sehr bekannten Bibliotheken Formic, Redux Forms und React Hook Form sind alle drei kompatibel mit React Native.[10,11,12]

Für die Formular-Anwendung ist die Validierung komplexer Bedingungen nötig. Die Formular-Validierungs-Bibliotheken bieten in der Regel Funktionen an, welche überprüfen, ob ein Feld gefüllt ist oder der Inhalt einem speziellen Muster entspricht – wie etwa einem regulären Ausdruck. Doch solche mitgelieferten Validierungs-Funktionen reichen nicht aus um die Komplexität der Bedingungen abzubilden. Stattdessen müssen benutzerdefinierte Funktionen zum Einsatz kommen.

Keiner der drei oben genannten Validierungs-Bibliotheken ist in dieser Hinsicht limitiert. Sie alle bieten die Möglichkeit, eine JavaScript Funktion für die Validierung zu übergeben. Diese Funktion gibt einen Wahrheitswert zurück – wahr, wenn das Feld oder die Felder valide sind, falsch, falls nicht. In React Hook Form ist es die Funktion register, die ein Parameter-Object namens Register Options erhält, dessen Eigenschaft validate die JavaScript Funktion zugewiesen werden kann.[13] In Redux Form ist es die initialisierungs-Funktion reduxForm, die ein Konfigurations-Objekt mit dem Namen config erhält, in welchem die Eigenschaft ebenfalls validate heißt.[14] Auch in Formic ist der Bezeichner validate, und ist als Attribut in der Formic Komponente zu finden.[15]

Es ist also absehbar, dass die Formular-Anwendung in React Native entwickelt werden kann. Die nötigen Funktionen werden von den Bibliotheken bereitgestellt. Einziger Nachteil hierbei ist, dass es sich um Drittanbieter Bibliotheken handelt, welche im Verlauf der Zeit an Beliebtheit gewinnen und verlieren können. Möglicherweise geht die Beliebtheit einer der Bibliothek mit der Zeit zurück, weshalb es weniger

[10] Vgl. *React Native | Formik Docs*
[11] Vgl. *Does redux-form work with React Native?*
[12] Vgl. *React Native | React Hook Form - Get Started*
[13] Vgl. *register | React Hook Form - API*
[14] Vgl. *reduxForm | Redux Form - API*
[15] Vgl. *<Formik /> | Formik Docs API*

Kontributionen wie etwa neue Funktionalitäten oder Fehlerbehebungen, sowie Fragen und Antworten und Anleitungen zu diesen Bibliotheken geben wird, da die Entwickler sich für andere Bibliotheken entscheiden. Die Wahl der Bibliothek kann also schwerwiegende Folgen wie Mangel an Dokumentation oder Limitationen im Vergleich zu anderen Bibliotheken mit sich bringen. Eine Migration von der einen Bibliothek zu einer anderen könnte in Zukunft notwendig werden, wenn diese Limitationen während der Entwicklung auffallen. Aus dem Grund ist es in der Regel von Vorteil, wenn solche Funktionalitäten bereits im Kern der Frontend-Technologie integriert sind. Der Fall, dass die Kern-Komponenten an Relevanz verlieren und empfohlen wird, auf externe Bibliotheken zuzugreifen, ist zwar nicht ausgeschlossen, geschied aber im wesentlichen seltener.

Flutter Die Flutter Dokumentation stellt in ihrer cookbook Sektion ein Beispiel einer minimalistischen Formularanwendung mit Validierung bereit.[16] Das Rezept ist Teil einer Serie von insgesamt fünf Anleitungen, welche Formulare in Flutter behandeln.[17]

2.2.2 Automatisiertes Testen

Automatisierte Test in React Native Die React Native Dokumentation führt genau eine Seite mit einem Überblick über die unterschiedlichen Testarten. Dabei wird das Konzept von Unit Tests, Mocking, Integrations Tests, Komponenten Tests und Snapshot Tests kurz erläutert, jedoch ohne ein Beispiel zu geben oder zu verlinken. Vier Quellcodeschnipsel sind auf der Seite zu finden: Ein Schnipsel zeigt den minimalen Aufbau eines Tests, zwei weitere Schnipsel veranschaulichen Beispielhaft, wie Nutzerinteraktionen getestet werden können und Letzteres zeigt die textuelle Representation der Ausgabe einer Komponente, die für einen Snapshottest verwendet wird. Weiterhin wird auf die Jest API Dokumentation verwiesen, sowie auf ein Beispiel für einen Snapshot Test in der Jest Dokumentation.[I]

Um die notwendigen Anleitungen für das Erstellen der jeweiligen Tests ausfindig zu machen, ist es notwendig, die Dokumentation von React Native zu verlassen.

Die Dokumentation von Jest enthält mehr Details zum Einsatz der Testbibliothek, welches für mehrere auf Javascript basierende Frontend Framework kompatibel ist[II]. Somit muss zum Erstellen der Unit-Tests immerhin nur dieses Framework studiert werden.

Zum Entwickeln von Test von React Testing Library React Native Komponenten wird unter anderem auf die Bibliothek React Native Testing Library verwiesen. Anders als der Name vermuten lässt, handelt es sich nicht um eine von React Native bereitgestellte Bibliothek. Im Unterschied zur React Testing Library, von der sie inspiriert ist, läuft sie ebenso wie React Native selbst nicht in einer Browser-Umgebung.[18] Herausgegeben wird die react native testing Library vom Drittanbieter

[16] Vgl. Google LLC, *Build a form with validation*
[17] Vgl. Google LLC, *Forms | Flutter Docs Cookbook*
[18] Vgl. Borenkraout, *Native Testing Library Introduction | Testing Library Docs*

[I] https://jestjs.io/docs/snapshot-testing
[II] https://jestjs.io/docs/getting-started

Callstack - ein Partner im React Native Ökosystem.[19]

Sie verwendet im Hintergrund den React Test Renderer[III], welcher wiederum vom React Team angeboten wird und auch zum Testen von react.js Anwendungen geeignet ist. Der React Test Renderer Wird ebenfalls empfohlen, um Komponententest zu kreieren, die keine React Native spezifischen Funktionalitäten nutzen.

Um Integrationstest so entwickeln, welche die Applikation auf einem physischen Gerät oder auf einem Emulator testen, wird auf zwei weitere Drittanbieter Bibliotheken verlinkt: Appium[IV] und Detox[V]. Es wird darauf hingewiesen, dass Detox speziell für die Entwicklung von React Native Integrationstest entwickelt wurde. Appium wird lediglich als ein weiteres bekanntes Werkzeug erwähnt.

Es lässt sich damit zusammenfassen, dass der Aufwand der Einarbeitung für automatisiertes Testen in React Native vergleichsweise hoch ist. Die Dokumentation ist auf die Seiten der jeweiligen Anbieter verteilt. Der Entwickler muss sich den Überblick selbst verschaffen und zusätzlich die für das Framework React Native relevanten Inhalte identifizieren. Notwendig ist auch das Erlernen von mehreren APIs um alle Testarten abzudecken. Für einen Anfänger kommt erschwerend hinzu, dass eine Entscheidung für die eine oder andere Bibliothek notwendig wird. Um diese Entscheidung treffen zu können, ist eine Auseinandersetzung mit den Vor- und Nachteile der Technologien im Vorfeld vom Entwickler zu leisten.

Automatisierte Test in Flutter Die Flutter Dokumentation erklärt sehr umfangreich auf 11 Unterseiten die unterschiedlichen Testarten mit Quellcodebeispielen und verlinkt für jede Testart eine bis mehrere detaillierte Schritt-für-Schritt-Anleitungen, wie ein solcher Test erstellt wird.

Einer Seite erklärt den Unterschied zwischen Unit Test, Widget Test und Integrationstest[VI]. Eine weitere Seite erklärt Integrationstests in mehr Detail[VII].

Ein sogenanntes Codelab führt durch die Erstellung einer minimalistischen App und zwei Unit-, fünf Widget- und zwei Integrationstest für diese App[VIII]

Im sogenannten Kochbuch tauchen folgende Rezepte auf:

- 2 Rezepte für Unit Tests
  - eine grundlegende Anleitung zum Erstellen von Unit-Tests [IX]
  - Eine weitere Anleitung zum Nutzen von mocks in Unit Test mithilfe der Bibliothek mockito [X]

[19] Vgl. Facebook Inc., *The React Native Ecosystem*

[III] https://reactjs.org/docs/test-renderer.html
[IV] http://appium.io/
[V] https://github.com/wix/detox/
[VI] https://flutter.dev/docs/testing
[VII] https://flutter.dev/docs/testing/integration-tests
[VIII] https://codelabs.developers.google.com/codelabs/flutter-app-testing
[IX] https://flutter.dev/docs/cookbook/testing/unit/introduction
[X] https://flutter.dev/docs/cookbook/testing/unit/mocking

- 3 Rezepte für Widget Tests
  - Eine grundlegende Anleitung zum Erstellen von Widget Test [XI]
  - Ein Rezept mit detaillierteren Beispielen zum Finden von widgets zur Laufzeit eines Widget Tests [XII]
  - Ein Rezept zum Testen von Nutzerverhalten wie dem Tab, dem Drag und dem eingeben von Text [XIII]
- 3 Rezepte für Integrationstests
  - Eine grundlegende Anleitung zum Erstellen eines Integrationstest [XIV]
  - eine Anleitung zum simulieren von scrollen in der Anwendung während der Laufzeit eines Integrationstest [XV]
  - eine Anleitung zum Performance Profiling [XVI]

Zusammengefasst: Der Aufwand der Einarbeitung in das Testen in Flutter ist gering. Alle Werkzeuge werden vom Dart- und Flutter-Team bereitgestellt. Die Dokumentation ist umfangreich, folgt jedoch einem roten Faden. Eine Übersichtsseite fasst die Kerninformationen zusammen und verweist auf die jeweiligen Seiten für detailliertere Informationen und Übungen.

[XI] https://flutter.dev/docs/cookbook/testing/widget/introduction
[XII] https://flutter.dev/docs/cookbook/testing/widget/finders
[XIII] https://flutter.dev/docs/cookbook/testing/widget/tap-drag
[XIV] https://flutter.dev/docs/cookbook/testing/integration/introduction
[XV] https://flutter.dev/docs/cookbook/testing/integration/scrolling
[XVI] https://flutter.dev/docs/cookbook/testing/integration/profiling

3 Implementierung

3.1 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. 7).

Abbildung 7: 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. 8).

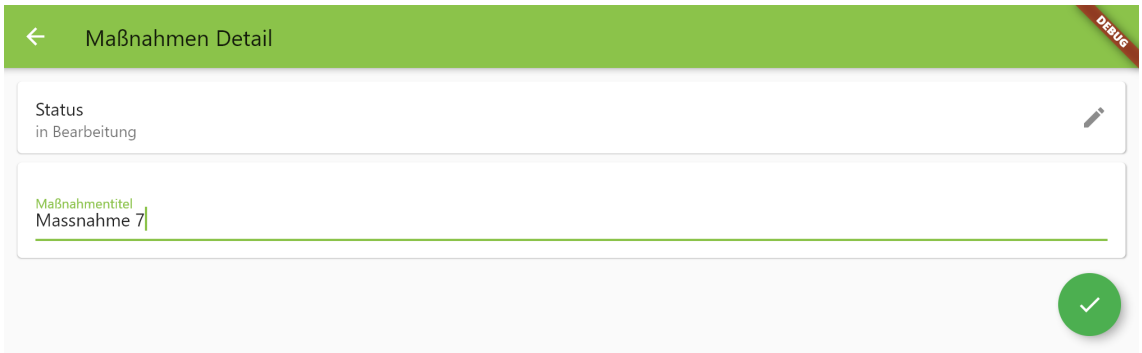

Abbildung 8: Die Eingabemaske zeigt im Schritt 1 eine Karte zum Selektieren des Status und ein Eingabefeld für den Titel. Quelle: Eigene Abbildung

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. 9).

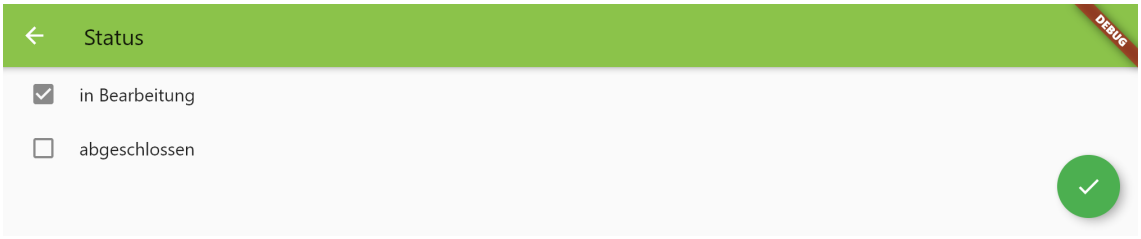

Abbildung 9: Der Selektions-Bildschirm für das Feld Status erlaubt die Auswahl der Optionen „in Bearbeitung“ und „abgeschlossen“. Quelle: Eigene Abbildung

3.1.1 Auswahloptionen hinzufügen

Dart verfügt – anders als beispielsweise Java[20] – nicht über Aufzählungstypen mit zusätzlichen Eigenschaften. Das Schlüsselwort `enum` in Dart erlaubt lediglich die Auflistung konstanter Symbole[21]. Für die Auswahl Optionen ist es jedoch notwendig, dass es zwei Eigenschaften gibt:

- die Abkürzung, die in der resultierendem 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 Applikationeingelesen 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.

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.[22]

Die Klasse `LetzterStatus` erbt von der Basisklasse `Choice` (Z. 5). Der Konstruktor der Klasse (Z. 9) übergibt beide Parameter als Argumente an den Konstruktor der

[20] Vgl. Gosling u. a., *The Java® Language Specification Java SE 16 Edition*, S. 321.
[21] Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 74f.
[22] Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 74f.

```
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

Klasse `Choice`. Genau wie in Java wird mithilfe des Schlüsselwortes `super`(Z. 10) der Konstruktor der Basisklasse aufgerufen. Doch anders als in Java erfolgt der Aufruf des super Konstruktors nicht in der ersten Zeile des Konstruktor-Körpers[23]. Weil das Aufrufen des Konstruktors der Basisklasse 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)[24].

Die Basisklasse `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.[25] 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.

```
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

Dies erübrigt sowohl die Angabe des Parametertypes mittels `(String abbreviation, String description)`, denn der Typ des Parameters kann bereits durch Angabe des Typs in der Instanzvariablen-Deklaration(Z. 4-5) abgeleitet werden. Außerdem entfällt auch die Zuweisung, die man ansonstenin der Form `this.abbreviation = abbreviation` und `this.description = description` in der Initialisierungsliste erreichen würde.[26]

Die Variable `letzterStatusChoices` fasst die beiden statischen Klassenvariablen als eine Kollektion zusammen. Da es sich um eine solche Kollektion handelt, in der jedes Element nur ein einziges Mal vorkommen darf, ist hier von einer Menge zu sprechen. Auffällig hier ist, dass das Schlüsselwort new fehlt. In Dart ist das Schlüsselwort für die Konstruktion von Instanzen optional. Die Klasse, die zur Konstruktion dieser Menge verwendet wird, ist die selbst erstellte Klasse choices. Über das Typargu-

[23] Vgl. Gosling u. a., *The Java® Language Specification Java SE 16 Edition*, S. 310.
[24] Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 42.
[25] Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. S16.
[26] Vgl. Google LLC, *Dart Programming Language Specification 5th edition*, S. 40f.

```
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

ment LetzterStatus wird erreicht, das ausschließlich Variablen dieses Typs in der Menge eingefügt werden dürfen. Wird stattdessen eine Variable eingefügt, die weder vom selben Typ, noch von einem Typ, der von letzter Status erbt, so gibt es einen Compilerfehler. Dies dient einzig und allein dem Zweck, dem Fehler vorzubeugen, dass aus Versehen falsche Optionen in der Menge eingetragen werden. Über den Parameter name ist es möglich dieser Menge die Beschriftung "Status" hinzuzufügen. Es handelt sich hier um einen benamten Parameter.

Listing 4 zeigt die Klasse choices. Sie erbt von UnmodifiableSetView und erlaubt damit die Erstellung einer eigenen Menge - auch Set genannt. Methoden, die man von einem Set erwartet, lassen sich somit direkt auf Instanzen der Klasse Choices aufrufen. Darunter unter anderem die contains Methode, welche erlaubt, das Vorhandensein eines Objektes im Set zu überprüfen. Instanzvariable name - deklariert in Zeile 11 - wird im Konstruktor in Zeile 16 zugewiesen. Auffällig hierbei ist, dass der Parameter in geschweiften Klammern geschrieben steht und das Schlüsselwort required vorangestellt ist. Das macht den Parameter zu einem vorgeschriebenen benannten Parameter.

Vorgeschriebene benamte Parameter Gewöhnlicher benannte Parameter sind optional. Wird ihnen das Schlüsselwort `required` vorangestellt, so müssen sie gesetz werden, denn sonst gibt es einen Compilerfehler. An dieser Stelle ist das `required` Schlüsselwort sinnvoll, denn es handelt sich um den Datentyp String der nicht den wert `null` annehmender. Würde der Parameter aber optional sein, so wäre es möglich, das programm zu kompilieren, auch wenn bei Aufrufen des Konstruktors kein Argument für den Parameter übergeben wurde. Doch in diesem Fall gäbe es keinen Initialwert für Name und somit müsste der Instanzvariable null zugewiesen werden. in der statischen Analyse wird daher sichergestellt, das Instanzvariablen durch benannte Parameter nur dann initialisiert werden dürfen, wenn dieser durch required als vorgeschrieben gekennzeichnet sind und damit unter keinem Umstand ausgelassen werden können. Dürfte name den Wert Null annehmen, So würde es sich um den nullable Datentyp String mit der Notation String? Handeln.

Typen ohne NULL-Zulässigkeit Im Vergleich zu vielen anderen Programmiersprachen wie beispielsweise in Java wird in Dart zwischen gewöhnlichen Typen und nullable Typen unterschieden. In Sprachen wie Beispielsweise Java ist es nur bei atomaren Datentypen wie int und float vorgeschrieben einen Wert anzugeben. Null ist bei diesen primitiven Datentypen nicht als Wert erlaubt. Doch nicht atomare Datentypen erlauben immer die Angabe von Null als wert. Null drückt dabei immer das Nichtvorhandensein von Daten aus. Ab Dart 2.12 kann allen Datentypen standardmäßig keinen neuen Wert zugewiesen werden. das hat den Vorteil, dass der Compiler sich darauf verlassen kann, das eine Variable niemals den Wert 0 haben

kann. Das ist besonders dann nützlich, wenn auf Einem Objekt eine Methode aufgerufen wird. Ist die Referenz das Objekt ist in Wahrheit 0 so gibt es erst zur Laufzeit einen Fehler, dass die Methode auf der Referenz `null`nicht aufgerufen werden kann. Damit ein Laufzeitfehler geworfen werden kann, muss vor jedem Aufruf einer Methode auf einer Referenz überprüft werden, ob die Referenzen nicht `null` sind. Würde diese Überprüfung nicht stattfinden, so könnte kein Laufzeitfehler geworfen werden und das Programm würde ohne Fehlermeldung abstürzen. Handelt es sich allerdings um eine Referenz, die niemals den Wert 0 annehmen kann, so kann der Compiler die Überprüfung auf 0 Wörter für diese Referenzen überspringen. Damit hört zusätzlich die Ausführungsgeschwindigkeit erhöht, da die Überprüfung auf `null`Zeit in Anspruch nimmt. Vor allem aber ist das forderheft für den Wickler, da der Compiler Fehlermeldungen und Warnungen mitteilen kann, wenn Operationen mit Variablen mit potenziellen `null` Werten verwendet werden.

Neben Name wird mit choiceByAbbreviation eine weitere Instanzvariable deklariert 12. es handelt sich um den Datentyp map - eine Kollektion die Daten mittels Schlüssel Wertepaaren ablegen kann. als Schlüssel wird die Abkürzung mit dem Datentyp String verwendet. Als wert ist der generische typ Parameter t angegeben. der typ Parameter ist in Zeile zehn deklariert und muss mindestens von der Klasse Joyce erben. In choiceByAbbreviation werden also die Auswahlmöglichkeiten über ihre Abkürzung abgelegt und können über dieselbe wieder referenziert werden. Da es sich auch hier um eine unveränderliche Instanzvariable handelt, muss sie schon in der Initialisierungsliste initialisiert werden 17 bis 19. Dabei wird zunächst mit der öffnenden geschweiften Klammer 17 ein sogenanntes Map Literal begonnen, welches mit schließenden geschweiften Klammer 19 endet.

Set und Map Literale Dart erlaubt es setz und Maps als sogenannte literale zu deklarieren. es entfällt also die Instanziierung eines Sets oder einer Map über den Klassennamen und der darauffolgenden Zuweisung der einzelnen Werte. Stattdessen startet das literal mit einer öffnenden geschweiften Klammer und endet mit einer schließenden geschweiften Klammer Punkt innerhalb der Klammern werden die Werte im Fall eines Sets mit, getrennt nacheinander aufgeführt. Im Fall einer Map werden der Schlüssel und der Wert durch einen : voneinander getrennt und die Schlüsse Wertepaare wiederum durch, getrennt nacheinander aufgelistet.

Auffällig ist jedoch, dass In Zeile 18 dem Set lateral keine einfache Auflistung von Werten übergeben wird. Stattdessen wird das mit dem sogenannte collection for eine wiederholung verwendet.

Collection for Dart erlaubt es Schleifen innerhalb von Listen-, Set- und Map-Literalen zu verwenden. Dabei darf die Schleife jedoch keinen steifen Körper besitzen. Ledig der Schleifenkopf wird dazu in das literal selbstgeschrieben gefolgt von dem Wert, der bei jedem Schleifendurchlauf dem literal hinzugefügt werden soll. Da wir kann der Wert von der Schleifen variable abgeleitet werden. In Zeile 18 wird beispielsweise durch die Menge aller Auswahloptionen choices iteriert und dabei in jedem Schleifendurchlauf die Ausweise Option in die Variablen choice gespeichert. Während des Schleifendurchlauf wird dann ein Wertepaar gebildet wobei choice.abbreviation der Schlüssel ist und Das Objekt choice selbst der Wert.

Die Map choiceByAbbreviation erlaubt es nach der Initialisierung mit Hilfe der Methode fromAbbreviation 14 über die Abkürzung das dazugehörigen Choice Objekt abzurufen. Beispielsweise gibt der Befehl letzterStatusChoices.fromAbbreviation(“fertig”) das Objekt LetzterStatus(fertig", äbgeschlossen") zurück. Auffällig dabei ist das der Parameter abbreviation Mit dem Typen String? und der generische Rückgabetyp mit T? gekennzeichnet ist. Der Suffix ? macht beide zu Typen mit Null-Zulässigkeit.

Typen mit NULL-Zulässigkeit Wird in Dart hinter einem Typen ein `?` Angegeben, so kann die Variable nicht nur Werte annehmen, die dieser Datentyp zulässt sondern zusätzlich auch noch den Wert 0, der ausdrückt, dass die Variable keinen Wert hat. Methoden auf Objekten mit nur Zulässigkeit aufzurufen ist nicht ohne weiteres möglich. Der wird aus solange nicht durch eine Fallunterscheidung zuvor überprüft wird, dass die Variable tatsächlich nicht `null` ist oder der Aufruf der Methode mit dem Suffix ! nach dem Objekt erzwungen wird.

Die Methode fromAbbreviation Soll für die Deserialisierung genutzt werden. sollten in Formular Auswahlfelder leer gelassen worden sein, so haben entsprechenden variablen den Wert `null`. Wenn nun das Formular abgespeichert wird, so tauchen auch in der abgespeicherten json-datei keine Werte für das fällt auf. Wird die Dateiendung gelesen wird die Methode fromAbbreviation genutzt um aus der in der json-datei gespeicherten Abkürzung wieder die entsprechende Auswahl Option abzurufen. Sollte jedoch kein Wert hinterlegt sein so wird `letzterStatusChoices.fromAbbreviation(null)` aufgerufen werden. Dadurch wird klar, dass der Parameter `null` zulassen muss. Es impliziert auch, dass potenziell `null` zurückgeben werden kann, da für den Schlüssel `null` kein Wert in der Map hinterlegt sein kann. Deshalb erlaubt auch der Rückgabetyp `null`-Werte.

```
10 class Choices<T extends Choice> extends UnmodifiableSetView<T> {
11   final String name;
12   final Map<String, T> choiceByAbbreviation;
13
14   T? fromAbbreviation(String? abbreviation) =>
     ↪  choiceByAbbreviation[abbreviation];
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

4 Ein Test soll verifizieren, dass die Daten korrekt abgelegt werden

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 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. Entwicklungsumgebungen wie Visual Studio Code und Android Studio erlauben solchen Boilerplate Code generieren zu lassen und dabei nur die erforderlichen Platzhalter einzugeben. In Visual Studio Code werden diese Templates „Snippets“ genannt, in Android Studio heißen sie „Live Templates“. Listing 137 zeigt, wie das live Template für das Generieren eines Wertetyps für built_value aussieht. Templates für built_value wie dieses und weitere müssen nicht vom Nutzer eingegeben werden, sondern existieren bereits als Plugin für die beiden Entwicklungsumgebungen[XVII], [XVIII].

```
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

$ClassName$ 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

[XVII] https://plugins.jetbrains.com/plugin/13786-built-value-snippets
[XVIII] https://marketplace.visualstudio.com/items?itemName=GiancarloCode.built-value-snippets

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.

```
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

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

```
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 CC-CCCCCC.

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 7 ist das Datumsformat sichtbar.

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

```
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

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 10 zeig den Unit Test für die Serialisierung des Typs Maßnahme. In Zeile 8 wird ein Objekt der Klasse Massnahme instanzieiert. 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 „built_value“ 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 Referenz `massnahme` zugewiesen, wodurch sie ihren alten Wert verliert. Über die generiertet Methode `serializers.serializeWith` Kann das Objekt in Json übersetzt werden.

Der erste Parameter `Massnahme.serializer` gibt dabei an, wie diese Serialisierung erfolgen soll und auch dieses Objekt wurde von „built_value“ generiert. Der zweite Parameter ist die tatsächliche `massnahme`, die in Json umgewandelt werden soll. Die Zeilen 13 bis 21 erstellen das Json Dokument, mit dem das serialisierte Ergebnis am Ende verglichen werden soll. Dabei werden die gleichen Eigenschaften eingetragen. So etwa die `guid`(Z. 14), welcher bei der Initialisierung der Maßnahme automatisch und zufällig erstellt wurde. Außerdem das letzte Bearbeitungsdatum, welches den Zeitstempel erhält, zudem die Maßnahme generiert wurde. Da built_value bei der Serialisierung die Datumswerte in Mikrosekunden umwandelt, muss für das erwartete Json-Dokument das gleiche passieren (Z. 16-17). Der letzte Status (Z. 18) wird hierbei auf den Standardwert `'bearb'` gesetzt und der Maßnahmen Titel (Z. 20) auf den gleichen Wert, der in Zeile 9 übergeben wurde. Schließlich vergleicht die Methode `expect` das tatsächlich serialisierter Jason Dokument mit dem, welches zuvor zum Vergleich aufgebaut wurde. Der zweite Parameter ist ein sogenannter Matcher und die Variante mit dem Namen equals überprüft auf absolute Gleichheit.

```
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: Serialisierung einer Maßnahme Unittest, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/test/data_model/massnahme_test.dart

Analog zur Serialisierung testet der Unit-Test in Listing 11 auch die Deserialisierung. Das Json Dokument ist dabei sehr ähnlich und unterscheidet sich lediglich in zwei Details. Der `'guid'` wird auf einen festen Wert festgelegt (Z. 38), statt - wie zuvor - durch den in dem Initialisierungs-Prozess der Maßnahme zufällig generiert zu werden. Außerdem wird auch das letzte bearbeitungsdatum festgesetzt, nämlich auf diemicrosekunden 0 (Z. 40).

Zum Vergleich wird in den Zeilen 46 bis 52 eine Maßnahme über das Builder-Entwurfsmuster generiert und die gleichen feste Werte werden für die Eigenschaften übergeben. Dabei ist darauf zu achten, dass für das Datum die Microsekunden zunächst in ein Objekt von DateTime umgewandelt werden muss. Dafür wird der benahmte Konstruktor `DateTime.fromMillisecondsSinceEpoch` (Z. 51) aufgerufen.

```
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.fromMicrosecondsSinceEpoch(0, isUtc: true);
52     }));
53   var actualMassnahme =
54       serializers.deserializeWith(Massnahme.serializer, json);
55
56   expect(actualMassnahme, equals(expectedMassnahme));
```

Listing 11: Deserialisierung einer Maßnahme Unittest, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/test/data_model/massnahme_test.dart

Benanmte Konstruktoren In Programmiersprachen wie beispielsweise Java können Methoden überladen werden, indem ihre Methodensignatur geändert wird. Beim Aufruf der Methode kann über die Anzahl und die Typen der übergebenen Argumente die gewünschte Methode gewählt werden. Das gleiche gilt für Konstruktoren. Wird ein weiterer Konstruktor für eine Klasse in Java benötigt, so besteht einzig und allein die Möglichkeit darin, den Konstruktor zu überladen. Sowohl überladene Methoden als auch überladene Konstruktoren existieren in Dart nicht. Wird also in dartein Alternative constructorgewünscht Komma so muss er einen Namen bekommen. Beim Aufruf des Konstruktors wird dieser Name dann mit einem . nach dem Klassennamen angegeben, um den gewünschten Konstruktor zu benennen.

Ganz ähnlich wie bei der Serialisierung wird nun mit dem Befehl `serializers . deserializeWith` unter Angabe des Objektes, welches die Deserialisierung übernehmen soll - nämlich wiederum `Massnahme.serializer` - das Json-Dokument in ein Objekt das Werte-Typs `Massnahme` deserialisiert (Z. 53-54). Schließlichh wird in Zeile 56 das Ergebnis der Deserialisierung mit dem gewünschten Ergebnis verglichen.

Gibt man in der Kommandozeile den Befehl `flutter test test/data_model/massnahme _test.dart` ein, so werden die Tests in der Testdatei ausgeführt. Die Ausgabe `00:01 +2: All tests passed!` teilt mit, dass beide Tests ausgeführt und beide Ergebnisse mit den verglichenen Werten übereinstimmten.

4.1 Schritt 2

```dart
abstract class Storage implements Built<Storage, StorageBuilder> {
  @BuiltValueField(wireName: 'massnahmen')
  BuiltSet<Massnahme> get massnahmen;

  static void _initializeBuilder(StorageBuilder b) =>
      b..massnahmen = SetBuilder();

  Storage._();

  factory Storage([void Function(StorageBuilder) updates]) = _$Storage;

  static Serializer<Storage> get serializer => _$storageSerializer;
}
```

Listing 12: Der Werte-Typ Storage, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/data_model/storage.dart

5 HIER EINFÜGEN Status Choice

```
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   };
31   expect(actualJson, equals(expectedJson));
```

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));
```

Listing 14: Ein automatisierter Testfall überprüft, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/test/data_model/storage_test.dart

```dart
class MassnahmenJsonFile {
  Future<File> get _localMassnahmenJsonFile async {
    var directory = await getApplicationSupportDirectory();
    return File("${directory.path}/Maßnahmen.json");
  }

  Future<void> saveMassnahmen(Map<String, dynamic> massnahmenAsJson) async {
    var file = await _localMassnahmenJsonFile;
    await file.writeAsString(jsonEncode(massnahmenAsJson));
  }

  Future<Map<String, dynamic>> readMassnahmen() async {
    var file = await _localMassnahmenJsonFile;

    var fileExists = await file.exists();
    if (fileExists) {
      final fileContent = await file.readAsString();
      final jsonObject = jsonDecode(fileContent) as Map<String, dynamic>;

      return jsonObject;
    } else {
      throw MassnahmenFileDoesNotExistException("$file was not found");
    }
  }
}
```

Listing 15: Die Klasse MassnahmenJsonFile, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/persistence/massnahmen_json_file.dart

```dart
class MassnahmenPool {
  final MassnahmenJsonFile jsonFile;

  MassnahmenPool(this.jsonFile) {
    init();
  }

  final storageSubject = BehaviorSubject<Storage>.seeded(
      Storage((b) => b..massnahmen = SetBuilder()));

  Storage get storage => storageSubject.value;

  set storage(Storage m) {
    storageSubject.value = m;
  }

  init() async {
    refresh();
  }

  refresh() async {
    try {
      final massnahmenAsJson = await jsonFile.readMassnahmen();

      final massnahmen =
          serializers.deserializeWith(Storage.serializer, massnahmenAsJson)!;

      storage = massnahmen;
    } on MassnahmenFileDoesNotExistException {
      storage = Storage();
    }
  }

  putMassnahmeIfAbsent(Massnahme massnahme) async {
    var rebuild = storage.rebuild((b) => b.massnahmen
      ..removeWhere((m) => m.guid == massnahme.guid)
      ..add(massnahme));

    var serializedMassnahmen =
        serializers.serializeWith(Storage.serializer, rebuild);

    await jsonFile.saveMassnahmen(serializedMassnahmen as Map<String, dynamic>);

    storage = rebuild;
  }
}
```

Listing 16: Die Klasse MassnahmenPool, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/data_access/massnahmen_pool.dart

```dart
final createNewMassnahmeButtonKey = GlobalKey();

class MassnahmenMasterScreen extends StatelessWidget {
  static const routeName = '/massnahmen_master';

  const MassnahmenMasterScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);

    return Scaffold(
      appBar: AppBar(
        title: const Text('Maßnahmen Master'),
      ),
      body: StreamBuilder<Storage>(
          stream: massnahmenPool.storageSubject,
          builder: (context, _) {
            return SingleChildScrollView(

...

            );
          }),
      floatingActionButton: FloatingActionButton(
          key: createNewMassnahmeButtonKey,
          child: const Icon(
            Icons.post_add_outlined,
            color: Colors.white,
          ),
          onPressed: () {
            final vm =
                Provider.of<MassnahmenFormViewModel>(context, listen: false);

            vm.model = Massnahme();
            Navigator.of(context).pushNamed(MassnahmenDetailScreen.routeName);
          }),
    );
  }
}
```

Listing 17: Die Struktur der Klasse MassnahmenMasterScreen, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/screens/massnahmen_master.dart

```
return SingleChildScrollView(
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      const Padding(
        padding: EdgeInsets.all(16.0),
        child: Text(
          "Abgeschlossen",
          style: TextStyle(fontSize: 20),
        ),
      ),
      SingleChildScrollView(
          scrollDirection: Axis.horizontal,
          child: Padding(
            padding: const EdgeInsets.all(16.0),
            child: MassnahmenTable(
                massnahmenPool.storage.massnahmen
                    .where((m) =>
                        m.letzteBearbeitung.letzterStatus ==
                        LetzterStatus.fertig.abbreviation)
                    .toSet(), onSelect: (selectedMassnahme) {
              final vm = Provider.of<MassnahmenFormViewModel>(
                  context,
                  listen: false);
              vm.model = selectedMassnahme.rebuild((m) => m
                ..letzteBearbeitung.letztesBearbeitungsDatum =
                    DateTime.now().toUtc());
              Navigator.of(context)
                  .pushNamed(MassnahmenDetailScreen.routeName);
            }),
          )),
```

Listing 18: Die Ausgabe der finalen Maßnahmen, Quelle: Eigenes Listing,
Datei: Quellcode/Schritt-1/conditional_form/lib/screens/massnahmen_master.dart

```
.where((m) =>
    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

```dart
typedef OnSelectCallback = void Function(Massnahme selectedMassnahme);

class MassnahmenTable extends StatelessWidget {
  final Set<Massnahme> _massnahmenToDisplay;
  final OnSelectCallback? onSelect;

  const MassnahmenTable(this._massnahmenToDisplay, {this.onSelect, Key? key})
      : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Table(
      border: TableBorder.all(width: 3),
      defaultColumnWidth: const IntrinsicColumnWidth(),
      defaultVerticalAlignment: TableCellVerticalAlignment.middle,
      children: [
        TableRow(children: [
          _buildColumnHeader(const Text("Zuletzt bearbeitet am")),
          _buildColumnHeader(const Text("Maßnahmentitel"))
        ]),
        ..._massnahmenToDisplay.map((m) {
          return TableRow(children: [
            _buildSelectableCell(m, Text(m.letzteBearbeitung.formattedDate)),
            _buildSelectableCell(m, Text(m.identifikatoren.massnahmenTitel)),
          ]);
        }).toList(),
      ],
    );
  }

  Widget _buildColumnHeader(Widget child) => Padding(
        padding: const EdgeInsets.all(8.0),
        child: child,
      );

  Widget _buildSelectableCell(Massnahme m, Widget child,
          {double padding = 8.0}) =>
      TableRowInkWell(
        onTap: () {
          if (onSelect != null) {
            onSelect!(m);
          }
        },
        child: Padding(
          padding: EdgeInsets.all(padding),
          child: child,
        ),
      );
}
```

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

```dart
const saveMassnahmeTooltip = "Validiere und speichere Massnahme";

class MassnahmenDetailScreen extends StatelessWidget {
  static const routeName = '/massnahmen-detail';

  const MassnahmenDetailScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    final vm = Provider.of<MassnahmenFormViewModel>(context, listen: false);
    final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);

    Future<bool> saveRecordAndGoBackToOverviewScreen() {...}

    Widget createMassnahmenTitelTextFormField(MassnahmenFormViewModel vm) {...}

    return Scaffold(
        appBar: AppBar(
          title: const Text('Maßnahmen Detail'),
        ),
        body: WillPopScope(
          onWillPop: () => saveRecordAndGoBackToOverviewScreen(),
          child: Stack(
            children: [
              SingleChildScrollView(
                child: Center(
                  child: Padding(
                    padding: const EdgeInsets.all(8.0),
                    child: Column(...),
                  ),
                ),
              ),
              Align(
                alignment: Alignment.bottomRight,
                child: Padding(
                  padding: const EdgeInsets.all(16.0),
                  child: Column(
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      FloatingActionButton(
                        tooltip: saveMassnahmeTooltip,
                        heroTag: 'save_floating_action_button',
                        child: const Icon(Icons.check, color: Colors.white),
                        onPressed: () => saveRecordAndGoBackToOverviewScreen(),
                      )
                    ],
                  ),
                ),
              )
            ],
          ),
        ));
  }
}
```

Listing 22: Die Struktur des Bildschirms MassnahmenDetailScreen, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
child: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    SelectionCard<LetzterStatus>(
      title: letzterStatusChoices.name,
      allChoices: letzterStatusChoices,
      initialValue: {
        if (vm.letzterStatus.value != null)
          vm.letzterStatus.value!
      },
      onSelect: (selectedChoice) =>
          vm.letzterStatus.value = selectedChoice,
      onDeselect: (selectedChoice) =>
          vm.letzterStatus.value = null,
    ),
    createMassnahmenTitelTextFormField(vm),
    const SizedBox(height: 64)
  ],
),
```

Listing 23: Die Ausgabe der Formularfelder, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
Widget createMassnahmenTitelTextFormField(MassnahmenFormViewModel vm) {
  final focusNode = FocusNode();
  return Card(
    child: Padding(
      padding: const EdgeInsets.all(16.0),
      child: TextFormField(
        focusNode: focusNode,
        initialValue: vm.massnahmenTitel.value,
        decoration: const InputDecoration(
            hintText: 'Maßnahmentitel', labelText: 'Maßnahmentitel'),
        onChanged: (value) {
          vm.massnahmenTitel.value = value;
        },
      ),
    ),
  );
}
```

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

```dart
Widget createMultipleChoiceSelectionScreen(BuildContext context) {
  Future<bool> goBack() {
    Navigator.of(context).pop();
    return Future.value(true);
  }

  return Scaffold(
    appBar: AppBar(
      title: Text(title),
    ),
    body: Builder(builder: (context) {
      return StreamBuilder(
          stream: selectionViewModel,
          builder: (context, snapshot) {
            final selectedChoices = selectionViewModel.value;
            return ListView(children: [
              ...allChoices.map((ChoiceType c) {
                bool isSelected = selectedChoices.contains(c);

                onTileTab() {
                  selectionViewModel.value =
                      selectionViewModel.value.rebuild((b) {
                    b.replace(isSelected ? [] : [c]);
                  });

                  if (isSelected) {
                    onDeselect(c);
                  } else {
                    onSelect(c);
                  }
                }

                return ListTile(
                  key: Key(
                      "valid choice ${allChoices.name} - ${c.abbreviation}"),
                  title: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [Text(c.description)],
                  ),
                  leading: IconButton(
                    icon: Icon(isSelected
                        ? Icons.check_box
                        : Icons.check_box_outline_blank),
                    onPressed: onTileTab,
                  ),
                  onTap: onTileTab,
                );
              }).toList(),
            ]);
          });
    }),
    floatingActionButton: FloatingActionButton(
      onPressed: goBack,
      tooltip: confirmButtonTooltip,
      child: const Icon(Icons.check),
    ),
  );
}
```

Listing 26: Die Funktion createMultipleChoiceSelectionScreen, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/widgets/selection_card.dart

```dart
Widget build(BuildContext context) {
  final focusNode = FocusNode();

  navigateToSelectionScreen() async {
    focusNode.requestFocus();

    Navigator.push<Set<Choice>>(
        context,
        MaterialPageRoute(
            builder: (context) =>
                createMultipleChoiceSelectionScreen(context)));
  }

  return StreamBuilder(
      stream: selectionViewModel,
      builder: (context, snapshot) {
        final selectedChoices = selectionViewModel.value;
        return Card(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              ListTile(
                focusNode: focusNode,
                title: Text(title),
                subtitle:
                    Text(selectedChoices.map((c) => c.description).join(",")),
                trailing: const Icon(Icons.edit),
                onTap: navigateToSelectionScreen,
              )
            ],
          ),
        );
      });
}
```

Listing 27: Die Build Methode der SelectionCard, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/widgets/selection_card.dart

```dart
typedef OnSelect<ChoiceType extends Choice> = void Function(
    ChoiceType selectedChoice);

typedef OnDeselect<ChoiceType extends Choice> = void Function(
    ChoiceType selectedChoice);

const confirmButtonTooltip = 'Auswahl übernehmen';

class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
  final String title;
  final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
  final Choices<ChoiceType> allChoices;
  final OnSelect<ChoiceType> onSelect;
  final OnDeselect<ChoiceType> onDeselect;

  SelectionCard(
      {required this.title,
      required Iterable<ChoiceType> initialValue,
      required this.allChoices,
      required this.onSelect,
      required this.onDeselect,
      Key? key})
      : selectionViewModel = BehaviorSubject<BuiltSet<ChoiceType>>.seeded(
            BuiltSet.from(initialValue)),
        super(key: key);
```

Listing 28: Die Klasse SelectionCard, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/lib/widgets/selection_card.dart

```dart
Future<void> main() => integrationDriver();
```

Listing 29: Der Integration Test Driver, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/integration_test/driver.dart

```dart
const durationAfterEachStep = Duration(milliseconds: 1);

@GenerateMocks([MassnahmenJsonFile])
void main() {
  testWidgets('Can fill the form and save the correct json', (tester) async {
    final binding = IntegrationTestWidgetsFlutterBinding.ensureInitialized()
        as IntegrationTestWidgetsFlutterBinding;
    binding.framePolicy = LiveTestWidgetsFlutterBindingFramePolicy.fullyLive;

    final mockMassnahmenJsonFile = MockMassnahmenJsonFile();
    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

```dart
await tester.pumpWidget(MultiProvider(
  providers: [
    Provider<MassnahmenFormViewModel>(
        create: (_) => MassnahmenFormViewModel()),
    Provider<MassnahmenJsonFile>(create: (_) => mockMassnahmenJsonFile),
    Provider(
        create: (context) => MassnahmenPool(
            Provider.of<MassnahmenJsonFile>(context, listen: false))),
  ],
  builder: (context, child) => MaterialApp(
    title: 'Maßnahmen',
    initialRoute: MassnahmenMasterScreen.routeName,
    routes: {
      MassnahmenMasterScreen.routeName: (context) =>
          const MassnahmenMasterScreen(),
      MassnahmenDetailScreen.routeName: (context) =>
          const MassnahmenDetailScreen()
    },
  ),
));
```

Listing 31: Initialisierung des Widgets für den Integrations Tests, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/integration_test/app_test.dart

```dart
Future<void> tabSelectionCard(Choices choices, {Finder? ancestor}) async {
  final Finder textLabel;
  if (ancestor != null) {
    textLabel =
        find.descendant(of: ancestor, matching: find.text(choices.name));
  } else {
    textLabel = find.text(choices.name);
  }

  expect(textLabel, findsWidgets);

  final card = find.ancestor(of: textLabel, matching: find.byType(Card));
  expect(card, findsOneWidget);

  await tester.ensureVisible(card);
  await tester.tap(card);
  await tester.pumpAndSettle(durationAfterEachStep);
}
```

Listing 32: Die Hilfsmethode tabSelectionCard, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/integration_test/app_test.dart

```dart
Future<Finder> tabOption(Choice choice, {bool tabConfirm = false}) async {
  final choiceLabel = find.text(choice.description);
  expect(choiceLabel, findsOneWidget);

  var listTileKey = tester
      .element(choiceLabel)
      .findAncestorWidgetOfExactType<ListTile>()!
      .key!;
  var listTile = find.byKey(listTileKey);

  expect(listTile, findsOneWidget);

  await tester.ensureVisible(choiceLabel);
  await tester.tap(choiceLabel);
  await tester.pumpAndSettle(durationAfterEachStep);

  if (tabConfirm) {
    await tabConfirmButton();
  }

  return listTile;
}
```

Listing 33: Die Hilfsmethode tabOption, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/integration_test/app_test.dart

```dart
Future<void> fillTextFormField(
    {required String title, required String text}) async {
  final textFormField = find
      .ancestor(of: find.text(title), matching: find.byType(TextFormField))
      .first;
  expect(textFormField, findsOneWidget);

  await tester.ensureVisible(textFormField);
  await tester.enterText(textFormField, text);
  await tester.pumpAndSettle(durationAfterEachStep);
}
```

Listing 34: Die Hilfsmethode fillTextFormField, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-1/conditional_form/integration_test/app_test.dart

```dart
await tester.pumpAndSettle(durationAfterEachStep);

var createNewMassnahmeButton = find.byKey(createNewMassnahmeButtonKey);
final gesture = await tester.press(createNewMassnahmeButton);
await tester.pumpAndSettle(durationAfterEachStep);
await gesture.up();
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 aukm_nur_vns = FoerderklasseChoice("aukm_nur_vns",
10       "Agrarumwelt-(und Klima)Maßnahme: nur Vertragsnaturschutz");
11   static final aukm_ohne_vns = FoerderklasseChoice("aukm_ohne_vns",
12       "Agrarumwelt-(und Klima)Maßnahmen, tw. auch mit Tierwohlaspekten, aber OHNE
         ↪ Vertragsnaturschutz");
13   static final twm_ziel = FoerderklasseChoice(
14       "twm_ziel", "Tierschutz/Tierwohlmaßnahmen mit diesem als Hauptziel");
15   static final contact =
16       FoerderklasseChoice("contact", "bitte um Unterstützung");
```

Listing 42: Die Klasse FoerderklasseChoice, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
final foerderklasseChoices = Choices<FoerderklasseChoice>({
  FoerderklasseChoice.oelb,
  FoerderklasseChoice.azl,
  FoerderklasseChoice.ea,
  FoerderklasseChoice.aukm_nur_vns,
  FoerderklasseChoice.aukm_ohne_vns,
  FoerderklasseChoice.twm_ziel,
  FoerderklasseChoice.contact
}, name: "Förderklasse");
```

Listing 43: Die Menge foerderklasseChoices, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
class KategorieChoice extends Choice {
  static final zf_us =
      KategorieChoice("zf_us", "Anbau Zwischenfrucht/Untersaat");
  static final anlage_pflege =
      KategorieChoice("anlage_pflege", "Anlage/Pflege Struktur");
  static final dungmang = KategorieChoice("dungmang", "Düngemanagement");
  static final extens = KategorieChoice("extens", "Extensivierung");
  static final flst = KategorieChoice("flst", "Flächenstilllegung/Brache");
  static final umwandlg = KategorieChoice("umwandlg", "Nutzungsumwandlung");
  static final bes_kult_rass = KategorieChoice(
      "bes_kult_rass", "Förderung bestimmter Rassen / Sorten / Kulturen");
  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

```dart
final kategorieChoices = Choices<KategorieChoice>({
  KategorieChoice.zf_us,
  KategorieChoice.anlage_pflege,
  KategorieChoice.dungmang,
  KategorieChoice.extens,
  KategorieChoice.flst,
  KategorieChoice.umwandlg,
  KategorieChoice.bes_kult_rass,
  KategorieChoice.contact
}, name: "Kategorie");
```

Listing 45: Die Menge kategorieChoices, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
class ZielflaecheChoice extends Choice {
  static final ka = ZielflaecheChoice("ka", "keine Angabe/Vorgabe");
  static final al = ZielflaecheChoice("al", "AL");
  static final gl = ZielflaecheChoice("gl", "GL");
  static final lf = ZielflaecheChoice("lf", "LF");
  static final dk_sk = ZielflaecheChoice("dk_sk", "DK/SK");
  static final hff = ZielflaecheChoice("hff", "HFF");
  static final biotop_le =
      ZielflaecheChoice("biotop_le", "Landschaftselement/Biotop o.Ä.");
  static final wald = ZielflaecheChoice("wald", "Wald/Forst");
  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

```dart
final zielflaecheChoices = Choices<ZielflaecheChoice>({
  ZielflaecheChoice.ka,
  ZielflaecheChoice.al,
  ZielflaecheChoice.gl,
  ZielflaecheChoice.lf,
  ZielflaecheChoice.dk_sk,
  ZielflaecheChoice.hff,
  ZielflaecheChoice.biotop_le,
  ZielflaecheChoice.wald,
  ZielflaecheChoice.contact
}, name: "Zielfläche");
```

Listing 47: Die Menge zielflaecheChoices, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
class ZieleinheitChoice extends Choice {
  static final ka = ZieleinheitChoice("ka", "keine Angabe/Vorgabe");
  static final m3 = ZieleinheitChoice("m3", "m³ (z.B. Gülle)");
  static final pieces =
      ZieleinheitChoice("pieces", "Kopf/Stück (z.B. Tiere oder Bäume)");
  static final gve = ZieleinheitChoice("gve", "GV/GVE");
  static final rgve = ZieleinheitChoice("rgve", "RGV");
  static final ha = ZieleinheitChoice("ha", "ha");
  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

```dart
final zieleinheitChoices = Choices<ZieleinheitChoice>({
  ZieleinheitChoice.ka,
  ZieleinheitChoice.m3,
  ZieleinheitChoice.pieces,
  ZieleinheitChoice.gve,
  ZieleinheitChoice.rgve,
  ZieleinheitChoice.ha,
  ZieleinheitChoice.contact
}, name: "Zieleinheit");
```

Listing 49: Die Menge zieleinheitChoices, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
class ZielsetzungLandChoice extends Choice {
  static final ka = ZielsetzungLandChoice("ka", "keine Angabe/Vorgabe");
  static final bsch = ZielsetzungLandChoice("bsch", "Bodenschutz");
  static final wsch = ZielsetzungLandChoice("wsch", "Gewässerschutz");
  static final asch = ZielsetzungLandChoice("asch", "Spezieller Artenschutz");
  static final biodiv = ZielsetzungLandChoice("biodiv", "Biodiversität");
  static final strutktviel =
      ZielsetzungLandChoice("strutktviel", "Erhöhung der Strukturvielfalt");
  static final genet_res = ZielsetzungLandChoice("genet_res",
      "Erhaltung genetischer Ressourcen (Pflanzen, z. B. im Grünland, und Tiere,
      ↪  z. B. bedrohte Rassen)");
  static final tsch = ZielsetzungLandChoice(
      "tsch", "Tierschutz/Maßnahmen zum Tierwohl im Betrieb");
  static final klima = ZielsetzungLandChoice("klima", "Klima");
  static final contact =
      ZielsetzungLandChoice("contact", "bitte um Unterstützung");
```

Listing 50: Die Klasse ZielsetzungLandChoice, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-2/conditional_form/lib/choices/choices.dart

```dart
final _zielsetzungLandChoices = {
  ZielsetzungLandChoice.ka,
  ZielsetzungLandChoice.bsch,
  ZielsetzungLandChoice.wsch,
  ZielsetzungLandChoice.asch,
  ZielsetzungLandChoice.biodiv,
  ZielsetzungLandChoice.strutktviel,
  ZielsetzungLandChoice.genet_res,
  ZielsetzungLandChoice.tsch,
  ZielsetzungLandChoice.klima,
  ZielsetzungLandChoice.contact
};

final hauptzielsetzungLandChoices = Choices<ZielsetzungLandChoice>(
    _zielsetzungLandChoices,
    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

```
final massnahmenTitel = BehaviorSubject<String>.seeded("");

final foerderklasse = BehaviorSubject<FoerderklasseChoice?>.seeded(null);
final kategorie = BehaviorSubject<KategorieChoice?>.seeded(null);
final zielflaeche = BehaviorSubject<ZielflaecheChoice?>.seeded(null);
final zieleinheit = BehaviorSubject<ZieleinheitChoice?>.seeded(null);
final hauptzielsetzungLand =
    BehaviorSubject<ZielsetzungLandChoice?>.seeded(null);

set model(Massnahme model) {
  guid.value = model.guid;

  letzterStatus.value = letzterStatusChoices
      .fromAbbreviation(model.letzteBearbeitung.letzterStatus);
  massnahmenTitel.value = model.identifikatoren.massnahmenTitel;

  {
    final mc = model.massnahmenCharakteristika;

    foerderklasse.value =
        foerderklasseChoices.fromAbbreviation(mc.foerderklasse);
    kategorie.value = kategorieChoices.fromAbbreviation(mc.kategorie);

    zielflaeche.value = zielflaecheChoices.fromAbbreviation(mc.zielflaeche);
    zieleinheit.value = zieleinheitChoices.fromAbbreviation(mc.zieleinheit);
    hauptzielsetzungLand.value =
        hauptzielsetzungLandChoices.fromAbbreviation(mc.hauptzielsetzungLand);
  }
}

Massnahme get model => Massnahme((b) => b
  ..guid = guid.value
  ..letzteBearbeitung.letzterStatus = letzterStatus.value?.abbreviation
  ..letzteBearbeitung.letztesBearbeitungsDatum = DateTime.now().toUtc()
  ..identifikatoren.update((b) => b..massnahmenTitel = massnahmenTitel.value)
  ..massnahmenCharakteristika.update((b) => b
    ..foerderklasse = foerderklasse.value?.abbreviation
    ..kategorie = kategorie.value?.abbreviation
    ..zielflaeche = zielflaeche.value?.abbreviation
    ..zieleinheit = zieleinheit.value?.abbreviation
    ..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

```
_buildColumnHeader(const Text("Maßnahmentitel")),
_buildColumnHeader(const Text("Förderklasse")),
_buildColumnHeader(const Text("Kategorie")),
_buildColumnHeader(const Text("Zielfläche")),
_buildColumnHeader(const Text("Zieleinheit")),
_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

```
_buildSelectableCell(m, Text(m.identifikatoren.massnahmenTitel)),
_buildSelectableCell(
    m, Text(m.massnahmenCharakteristika.foerderklasse ?? "")),
_buildSelectableCell(
    m, Text(m.massnahmenCharakteristika.kategorie ?? "")),
_buildSelectableCell(
    m, Text(m.massnahmenCharakteristika.zielflaeche ?? "")),
_buildSelectableCell(
    m, Text(m.massnahmenCharakteristika.zieleinheit ?? "")),
_buildSelectableCell(m,
    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

```
final OnSelect<ChoiceType> onSelect;
final OnDeselect<ChoiceType> onDeselect;
final String? errorText;

SelectionCard(
    {required this.title,
    required Iterable<ChoiceType> initialValue,
    required this.allChoices,
    required this.onSelect,
    required this.onDeselect,
    this.errorText,
    Key? key})
```

Listing 58: errorText wird der SelectionCard hinzugefügt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/widgets/selection_card.dart

5.1 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

```
FloatingActionButton(
  mini: true,
  heroTag: 'save_draft_floating_action_button',
  child: const Icon(Icons.paste, color: Colors.white),
  backgroundColor: Colors.orange,
  onPressed: saveDraft,
),
const SizedBox(
  height: 10,
),
FloatingActionButton(
  tooltip: saveMassnahmeTooltip,
  heroTag: 'save_floating_action_button',
  child: const Icon(Icons.check, color: Colors.white),
  onPressed: validateAndSave,
)
```

Listing 61: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
void saveDraft() {
  ScaffoldMessenger.of(context)
    ..hideCurrentSnackBar()
    ..showSnackBar(
        const SnackBar(content: Text('Entwurf wird gespeichert ...')));

  var draft = vm.model.rebuild((b) =>
      b.letzteBearbeitung.letzterStatus = LetzterStatus.bearb.abbreviation);

  massnahmenPool.putMassnahmeIfAbsent(draft);
  Navigator.of(context).pop();
}
```

Listing 62: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
Widget build(BuildContext context) {
  final vm = Provider.of<MassnahmenFormViewModel>(context, listen: false);
  final massnahmenPool = Provider.of<MassnahmenPool>(context, listen: false);
  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

```
void showValidationError() {
  ScaffoldMessenger.of(context).showSnackBar(SnackBar(
      content: Row(
    children: [
      Text(
          'Fehler im Formular trotz Status
          ↪  "${LetzterStatus.fertig.description}"'),
      const SizedBox(width: 4),
      ElevatedButton(
        onPressed: saveDraft,
        child: Padding(
          padding: const EdgeInsets.fromLTRB(4, 4, 8, 4),
          child: Row(
            children: const [
              Icon(Icons.paste, color: Colors.white),
              SizedBox(width: 4),
              Text(
                "Entwurf speichern?",
                style: TextStyle(fontSize: 18.0, color: Colors.white),
              ),
            ],
          ),
        ),
      ),
    ],
  )));
}
```

Listing 64: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
bool inputsAreValidOrNotMarkedFinal() {
  if (vm.letzterStatus.value != LetzterStatus.fertig) {
    return true;
  }

  if (formKey.currentState!.validate()) {
    return true;
  }

  return false;
}
```

Listing 65: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
Future<bool> validateAndSave() {
  if (inputsAreValidOrNotMarkedFinal()) {
    saveRecordAndGoBackToOverviewScreen();
    return Future.value(true);
  } else {
    showValidationError();
    return Future.value(false);
  }
}
```

Listing 66: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
Widget buildSelectionCard<ChoiceType extends Choice>(
    {required Choices<ChoiceType> allChoices,
    required BehaviorSubject<ChoiceType?> selectionViewModel}) {
  return FormField(
      validator: (_) {
        if (selectionViewModel.value == null) {
          return "Feld ${allChoices.name} enthält keinen Wert!";
        }

        Iterable<Choice> choices = {
          if (selectionViewModel.value != null) selectionViewModel.value!
        };

        if (choices.isEmpty) {
          return "Feld ${allChoices.name} enthält keinen Wert!";
        }

        return null;
      },
      builder: (field) => SelectionCard<ChoiceType>(
            title: allChoices.name,
            allChoices: allChoices,
            initialValue: {
              if (selectionViewModel.value != null)
                selectionViewModel.value!
            },
            onSelect: (selectedChoice) =>
                selectionViewModel.value = selectedChoice,
            onDeselect: (selectedChoice) => selectionViewModel.value = null,
            errorText: field.errorText,
          ));
}
```

Listing 67: Die Maßnahmencharakteristika Selektionskarten werden ergänzt, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-3/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

5.2 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

```
return StreamBuilder(
    stream: selectionViewModel,
    builder: (context, snapshot) {
      final selectedChoices = selectionViewModel.value;
      final bool wrongSelection = selectedChoices
          .any((c) => !c.conditionMatches(priorChoices.value));

      return Card(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            ListTile(
              focusNode: focusNode,
              title: Text(title),
              subtitle:
                  Text(selectedChoices.map((c) => c.description).join(",")),
              trailing: const Icon(Icons.edit),
              onTap: navigateToSelectionScreen,
              tileColor:
                  wrongSelection || errorText != null ? Colors.red : null,
```

Listing 69: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-4/conditional_form/lib/widgets/selection_card.dart

```
body: Builder(builder: (context) {
  return StreamBuilder(
      stream: selectionViewModel,
      builder: (context, snapshot) {
        final selectedChoices = selectionViewModel.value;

        Set<ChoiceType> selectedAndSelectableChoices = {};
        Set<ChoiceType> unselectableChoices = {};

        for (ChoiceType c in allChoices) {
          if (selectedChoices.contains(c) ||
              c.conditionMatches(priorChoices.value)) {
            selectedAndSelectableChoices.add(c);
          } else {
            unselectableChoices.add(c);
          }
        }
```

Listing 70: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-4/conditional_form/lib/widgets/selection_card.dart

```
return ListView(children: [
  ...selectedAndSelectableChoices.map((ChoiceType c) {
    bool isSelected = selectedChoices.contains(c);
    bool selectedButDoesNotMatch =
        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

```dart
class ZielflaecheChoice extends Choice {
  static final ka = ZielflaecheChoice("ka", "keine Angabe/Vorgabe");
  static final al = ZielflaecheChoice("al", "AL",
      condition: (choices) => !choices.contains(KategorieChoice.zf_us));
  static final gl = ZielflaecheChoice("gl", "GL");
  static final lf = ZielflaecheChoice("lf", "LF");
  static final dk_sk = ZielflaecheChoice("dk_sk", "DK/SK",
      condition: (choices) => !choices.contains(FoerderklasseChoice.twm_ziel));
  static final hff = ZielflaecheChoice("hff", "HFF");
  static final biotop_le = ZielflaecheChoice(
      "biotop_le", "Landschaftselement/Biotop o.Ä.",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.azl) ||
              choices.contains(FoerderklasseChoice.ea) ||
              choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
          (!choices.contains(KategorieChoice.zf_us) ||
              !choices.contains(KategorieChoice.bes_kult_rass)));
  static final wald = ZielflaecheChoice("wald", "Wald/Forst",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.ea) ||
              choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
          (!choices.contains(KategorieChoice.zf_us) ||
              !choices.contains(KategorieChoice.bes_kult_rass)));
  static final contact = ZielflaecheChoice("contact", "bitte um Unterstützung");

  ZielflaecheChoice(String abbreviation, String description,
      {bool Function(Set<Choice> choices)? condition})
      : super(abbreviation, description, condition: condition);
}
```

Listing 82: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-4/conditional_form/lib/choices/choices.dart

```dart
class ZieleinheitChoice extends Choice {
  static final ka = ZieleinheitChoice("ka", "keine Angabe/Vorgabe");
  static final m3 = ZieleinheitChoice("m3", "m³ (z.B. Gülle)",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns)) &&
          (choices.contains(KategorieChoice.dungmang) ||
              choices.contains(KategorieChoice.extens)) &&
          (!choices.contains(ZielflaecheChoice.ka) &&
              !choices.contains(ZielflaecheChoice.contact)));
  static final pieces = ZieleinheitChoice(
      "pieces", "Kopf/Stück (z.B. Tiere oder Bäume)",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
              choices.contains(FoerderklasseChoice.twm_ziel)) &&
          (!choices.contains(KategorieChoice.zf_us) ||
              !choices.contains(KategorieChoice.flst) ||
              !choices.contains(KategorieChoice.umwandlg)) &&
          (!choices.contains(ZielflaecheChoice.ka) &&
              !choices.contains(ZielflaecheChoice.contact)));
  static final gve = ZieleinheitChoice("gve", "GV/GVE",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
              choices.contains(FoerderklasseChoice.twm_ziel)) &&
          (!choices.contains(KategorieChoice.zf_us) ||
              !choices.contains(KategorieChoice.anlage_pflege) ||
              !choices.contains(KategorieChoice.flst) ||
              !choices.contains(KategorieChoice.umwandlg)) &&
          (!choices.contains(ZielflaecheChoice.ka) &&
              !choices.contains(ZielflaecheChoice.contact)));
  static final rgve = ZieleinheitChoice("rgve", "RGV",
      condition: (choices) =>
          (choices.contains(FoerderklasseChoice.aukm_nur_vns) ||
              choices.contains(FoerderklasseChoice.aukm_ohne_vns) ||
              choices.contains(FoerderklasseChoice.twm_ziel)) &&
          (!choices.contains(KategorieChoice.zf_us) ||
              !choices.contains(KategorieChoice.anlage_pflege) ||
              !choices.contains(KategorieChoice.flst) ||
              !choices.contains(KategorieChoice.umwandlg)) &&
          (!choices.contains(ZielflaecheChoice.ka) &&
              !choices.contains(ZielflaecheChoice.contact)));
  static final ha = ZieleinheitChoice("ha", "ha",
      condition: (choices) =>
          !choices.contains(ZielflaecheChoice.ka) &&
          !choices.contains(ZielflaecheChoice.contact));
  static final contact = ZieleinheitChoice("contact", "bitte um Unterstützung");

  ZieleinheitChoice(String abbreviation, String description,
      {bool Function(Set<Choice> choices)? condition})
      : super(abbreviation, description, condition: condition);
}
```

Listing 83: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-4/conditional_form/lib/choices/choices.dart

5.3 Schritt 5

```
await tabSelectionCard(nebenzielsetzungLandChoices);
await tabOption(ZielsetzungLandChoice.bsch);
await tabOption(ZielsetzungLandChoice.klima, tabConfirm: true);

var saveMassnahmeButton = find.byTooltip(saveMassnahmeTooltip);
await tester.tap(saveMassnahmeButton);
await tester.pumpAndSettle(durationAfterEachStep);
```

Listing 84: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/integration_test/app_test.dart

```
'massnahmenCharakteristika': {
  'nebenziele': [
    'bsch',
    'klima',
  ],
  'foerderklasse': 'aukm_ohne_vns',
```

Listing 85: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/integration_test/app_test.dart

```
final hauptzielsetzungLandChoices = Choices<ZielsetzungLandChoice>(
    _zielsetzungLandChoices,
    name: "Hauptzielsetzung Land");

final nebenzielsetzungLandChoices =
    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

```
Widget buildSelectionCard<ChoiceType extends Choice>(
    {bool multiSelection = false,
    required Choices<ChoiceType> allChoices,
    required BehaviorSubject<dynamic> selectionViewModel,
    OnSelect<ChoiceType>? onSelect,
    OnDeselect<ChoiceType>? onDeselect}) {
  Iterable<ChoiceType> computeChoices(dynamic viewModel) {
    if (viewModel is BuiltSet) {
      return viewModel.map((e) => e).cast<ChoiceType>();
    } else if (viewModel is ChoiceType?) {
      return {if (viewModel != null) viewModel};
    } else {
      throw ArgumentError.value(viewModel);
    }
  }

  void _defaultOnSingleSelectStrategy(ChoiceType selectedValue) {
    var svm = selectionViewModel as BehaviorSubject<ChoiceType?>;
    svm.value = selectedValue;
  }

  void _defaultOnSingleDeselectStrategy(ChoiceType selectedValue) {
    var svm = selectionViewModel as BehaviorSubject<ChoiceType?>;
    svm.value = null;
  }

  void _defaultOnMultiSelectStrategy(ChoiceType selectedValue) {
    var svm = selectionViewModel as BehaviorSubject<BuiltSet<ChoiceType>>;
    svm.value = svm.value.rebuild((b) => b.add(selectedValue));
  }

  void _defaultOnMultiDeselectStrategy(ChoiceType deselectedValue) {
    var svm = selectionViewModel as BehaviorSubject<BuiltSet<ChoiceType>>;
    svm.value = svm.value.rebuild((b) => b.remove(deselectedValue));
  }

  OnSelect<ChoiceType> onSelectNonNull;
  OnDeselect<ChoiceType> onDeselectNonNull;
  if (multiSelection) {
    onSelectNonNull = onSelect ?? _defaultOnMultiSelectStrategy;
    onDeselectNonNull = onDeselect ?? _defaultOnMultiDeselectStrategy;
  } else {
    onSelectNonNull = onSelect ?? _defaultOnSingleSelectStrategy;
    onDeselectNonNull = onDeselect ?? _defaultOnSingleDeselectStrategy;
  }
```

Listing 88: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
Iterable<Choice> choices = computeChoices(selectionViewModel.value);

if (choices.isEmpty) {
  return "Feld ${allChoices.name} enthält keinen Wert!";
}
```

Listing 89: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
builder: (field) => SelectionCard<ChoiceType>(
      title: allChoices.name,
      multiSelection: multiSelection,
      allChoices: allChoices,
      priorChoices: fvm.priorChoices,
      initialValue: computeChoices(selectionViewModel.value),
      onSelect: onSelectNonNull,
      onDeselect: onDeselectNonNull,
      errorText: field.errorText,
    ));
```

Listing 90: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
buildSelectionCard<ZielsetzungLandChoice>(
    allChoices: hauptzielsetzungLandChoices,
    selectionViewModel: vm.hauptzielsetzungLand),
buildSelectionCard(
    multiSelection: true,
    allChoices: nebenzielsetzungLandChoices,
    selectionViewModel: vm.nebenziele),
```

Listing 91: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
final hauptzielsetzungLand =
    BehaviorSubject<ZielsetzungLandChoice?>.seeded(null);
final nebenziele = BehaviorSubject<BuiltSet<ZielsetzungLandChoice>>.seeded(
    BuiltSet<ZielsetzungLandChoice>());
```

Listing 92: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/screens/massnahmen_detail/massnahmen_form_view_model.dart

```
hauptzielsetzungLand.value =
    hauptzielsetzungLandChoices.fromAbbreviation(mc.hauptzielsetzungLand);
nebenziele.value = BuiltSet(mc.nebenziele
    .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

```
..hauptzielsetzungLand = hauptzielsetzungLand.value?.abbreviation
..nebenziele =
    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

```
_buildColumnHeader(const Text("Hauptzielsetzung Land")),
_buildColumnHeader(const Text("Nebenziele")),
```

Listing 95: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/widgets/massnahmen_table.dart

```
_buildSelectableCell(m,
    Text(m.massnahmenCharakteristika.hauptzielsetzungLand ?? "")),
_buildSelectableCell(
    m,
    Column(
      children: m.massnahmenCharakteristika.nebenziele
          .map((n) => Text(n))
          .toList(),
    )),
```

Listing 96: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-5/conditional_form/lib/widgets/massnahmen_table.dart

```
class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
  final String title;
  final bool multiSelection;
  final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
  final Choices<ChoiceType> allChoices;
  final BehaviorSubject<Set<Choice>> priorChoices;
  final OnSelect<ChoiceType> onSelect;
  final OnDeselect<ChoiceType> onDeselect;
  final String? errorText;

  SelectionCard(
      {required this.title,
      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

5.4 Schritt 6

```
Widget buildSelectionCard<ChoiceType extends Choice>(
    {bool multiSelection = false,
    required Choices<ChoiceType> allChoices,
    required BehaviorSubject<dynamic> selectionViewModel,
    OnSelect<ChoiceType>? onSelect,
    OnDeselect<ChoiceType>? onDeselect,
    ChoiceMatcher<ChoiceType>? customChoiceMatcherStrategy}) {
  final ChoiceMatcher<ChoiceType> matcher =
      customChoiceMatcherStrategy ?? _defaultChoiceMatcherStrategy;
```

Listing 99: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
if (choices.isEmpty) {
  return "Feld ${allChoices.name} enthält keinen Wert!";
}

bool atLeastOneValueInvalid = choices
    .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

```
builder: (field) => SelectionCard<ChoiceType>(
      title: allChoices.name,
      multiSelection: multiSelection,
      allChoices: allChoices,
      priorChoices: fvm.priorChoices,
      initialValue: computeChoices(selectionViewModel.value),
      choiceMatcher: matcher,
      onSelect: onSelectNonNull,
      onDeselect: onDeselectNonNull,
      errorText: field.errorText,
    ));
```

Listing 101: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
  multiSelection: true,
  allChoices: nebenzielsetzungLandChoices,
  selectionViewModel: vm.nebenziele,
  customChoiceMatcherStrategy: (choice, priorChoices) {
    if (vm.hauptzielsetzungLand.value !=
            ZielsetzungLandChoice.ka &&
        vm.hauptzielsetzungLand.value !=
            ZielsetzungLandChoice.contact) {
      return choice != vm.hauptzielsetzungLand.value;
    } else {
      if (choice != ZielsetzungLandChoice.ka &&
          choice != ZielsetzungLandChoice.contact) {
        return false;
      }
      return true;
    }
  },
),
```

Listing 102: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

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

Listing 103: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

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

Listing 104: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

```dart
class SelectionCard<ChoiceType extends Choice> extends StatelessWidget {
  final String title;
  final bool multiSelection;
  final BehaviorSubject<BuiltSet<ChoiceType>> selectionViewModel;
  final Choices<ChoiceType> allChoices;
  final BehaviorSubject<Set<Choice>> priorChoices;
  final OnSelect<ChoiceType> onSelect;
  final OnDeselect<ChoiceType> onDeselect;
  final String? errorText;
  final ChoiceMatcher<ChoiceType> choiceMatcher;

  SelectionCard(
      {required this.title,
      required this.multiSelection,
      required Iterable<ChoiceType> initialValue,
      required this.allChoices,
      required this.priorChoices,
      required this.onSelect,
      required this.onDeselect,
      required this.choiceMatcher,
```

Listing 105: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

```dart
return StreamBuilder(
    stream: selectionViewModel,
    builder: (context, snapshot) {
      final selectedChoices = selectionViewModel.value;
      final bool wrongSelection =
          selectedChoices.any((c) => !choiceMatcher(c, priorChoices.value));
```

Listing 106: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

```dart
for (ChoiceType c in allChoices) {
  if (selectedChoices.contains(c) ||
      choiceMatcher(c, priorChoices.value)) {
    selectedAndSelectableChoices.add(c);
  } else {
    unselectableChoices.add(c);
  }
}
```

Listing 107: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

```dart
return ListView(children: [
  ...selectedAndSelectableChoices.map((ChoiceType c) {
    bool isSelected = selectedChoices.contains(c);
    bool selectedButDoesNotMatch =
        isSelected && !choiceMatcher(c, priorChoices.value);
```

Listing 108: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

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

Listing 109: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-6/conditional_form/lib/widgets/selection_card.dart

5.5 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

```dart
const Choice(this.abbreviation, this.description, {Condition? condition})
    : condition = condition ?? _conditionIsAlwaysMet;

@override
bool operator ==(Object other) =>
    identical(this, other) ||
    other is Choice &&
        description == other.description &&
        abbreviation == other.abbreviation;

@override
int get hashCode => description.hashCode ^ abbreviation.hashCode;
```

Listing 112: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/choices/base/choice.dart

```dart
class DuengungChoice extends Choice {
  static final o = DuengungChoice("o", "keine Angabe / ohne Einschränkung",
      condition: (choices) => !choices.contains(contact));
  static final dueng_keine = DuengungChoice("keine", "keine Düngung",
      condition: (choices) =>
          !choices.contains(o) && !choices.contains(contact));
  static final ausbring_techn = DuengungChoice("ausbring_techn",
      "besondere Ausbringungstechnik (z.B. Schleppschlauch)  mit weiteren Angaben
      ↪  in Technische Anforderungen",
      condition: (choices) =>
          !choices.contains(o) && !choices.contains(contact));
  static final ausbring_t_kurz = DuengungChoice(
      "ausbring_t_kurz", "verkürzte oder vorgegebene Ausbringungszeiten",
      condition: (choices) =>
          !choices.contains(o) && !choices.contains(contact));
  static final dueng_red = DuengungChoice(
      "dueng_red", "reduzierte Düngungsmenge",
      condition: (choices) =>
          !choices.contains(o) && !choices.contains(contact));
  static final dueng_spez_art = DuengungChoice(
      "dueng_spez_art", "Beschchränkung auf spezifische Düngemittel",
      condition: (choices) =>
          !choices.contains(o) && !choices.contains(contact));
  static final contact = DuengungChoice("contact", "bitte um Unterstützung",
      condition: (choices) => !choices.contains(o));

  bool get canSetArt => canSetArtCondition({this});
  final Condition canSetArtCondition =
      (choices) => choices.contains(dueng_spez_art);

  DuengungChoice(String abbreviation, String description,
      {bool Function(Set<Choice> choices)? condition})
      : super(abbreviation, description, condition: condition);
}
```

Listing 113: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/choices/choices.dart

```dart
final duengungChoices = Choices<DuengungChoice>({
  DuengungChoice.o,
  DuengungChoice.dueng_keine,
  DuengungChoice.ausbring_techn,
  DuengungChoice.ausbring_t_kurz,
  DuengungChoice.dueng_red,
  DuengungChoice.dueng_spez_art,
  DuengungChoice.contact,
}, name: "Düngung");
```

Listing 114: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/choices/choices.dart

```dart
class DuengungArtChoice extends Choice {
  static final o = DuengungArtChoice("o", "keine Angabe");
  static final dueng_org_tier_liqu = DuengungArtChoice(
      "dueng_org_tier_liqu", "organisch, tierisch und flüssig: z.B. Gülle");
  static final dueng_org_tier_fest = DuengungArtChoice(
      "dueng_org_tier_fest", "organisch, tierisch, fest: Festmist");
  static final dueng_org_pfl =
      DuengungArtChoice("dueng_org_pfl", "organisch, pflanzlich: Gärrest");
  static final dueng_org = DuengungArtChoice(
      "dueng_org", "organisch: nicht differenziert oder mehreres betreffend");
  static final dueng_miner = DuengungArtChoice("dueng_miner", "mineralisch");
  static final dueng_org_n_miner =
      DuengungArtChoice("dueng_org_n_miner", "organisch und mineralisch");
  static final contact = DuengungArtChoice("contact", "bitte um Unterstützung");

  DuengungArtChoice(String abbreviation, String description,
      {bool Function(Set<Choice> choices)? condition})
      : super(abbreviation, description, condition: condition);
}
```

Listing 115: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/choices/choices.dart

```dart
final duengungArtChoices = Choices<DuengungArtChoice>({
  DuengungArtChoice.o,
  DuengungArtChoice.dueng_org_tier_liqu,
  DuengungArtChoice.dueng_org_tier_fest,
  DuengungArtChoice.dueng_org_pfl,
  DuengungArtChoice.dueng_org,
  DuengungArtChoice.dueng_miner,
  DuengungArtChoice.dueng_org_n_miner,
  DuengungArtChoice.contact
}, 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

```
buildSectionHeadline("Düngung"),
buildSelectionCard<DuengungViewModel>(
    multiSelection: true,
    allChoices: Choices<DuengungViewModel>({
      ...vm.duengung.value,
      for (final c in duengungChoices)
        if (!vm.duengung.value.contains(c))
          DuengungViewModel(c)
    }, name: duengungChoices.name),
    selectionViewModel: vm.duengung,
    listTileTitleBuilder:
        (context, choice, isSelected) {
      return Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(choice.description),
          if (isSelected) ...[
            if (choice.canSetArt)
              buildSelectionCard(
                  allChoices: duengungArtChoices,
                  selectionViewModel: choice.art),
          ]
        ],
      );
    }),
const SizedBox(height: 64)
```

Listing 121: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/screens/massnahmen_detail/massnahmen_detail.dart

```
final nebenziele = BehaviorSubject<BuiltSet<ZielsetzungLandChoice>>.seeded(
    BuiltSet<ZielsetzungLandChoice>());
final duengung = BehaviorSubject<BuiltSet<DuengungViewModel>>.seeded(
    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

```
class DuengungViewModel extends DuengungChoice {
  final art = BehaviorSubject<DuengungArtChoice?>.seeded(null);

  DuengungViewModel(DuengungChoice choice)
      : super(choice.abbreviation, choice.description,
            condition: choice.condition);

  factory DuengungViewModel.fromModel(Duengung model) {
    final vm =
        DuengungViewModel(duengungChoices.fromAbbreviation(model.vorgabe)!);
    vm.model = model;
    return vm;
  }

  set model(Duengung model) {
    if (model.art != null) {
      art.value = duengungArtChoices.fromAbbreviation(model.art);
    }
  }

  Duengung get model => Duengung((b) {
        b.vorgabe = abbreviation;

        if (art.value != null) {
          b.art = art.value!.abbreviation;
        }
      });
}
```

Listing 126: XXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/screens/massnahmen_detail/massnahmen_form_view_model.dart

```
_buildColumnHeader(const Text("Nebenziele")),
_buildColumnHeader(const Text("Düngung"))
```

Listing 127: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/widgets/massnahmen_table.dart

```
_buildSelectableCell(
    m,
    Column(
      children: m.massnahmenCharakteristika.nebenziele
          .map((n) => Text(n))
          .toList(),
    )),
_buildSelectableCell(m, buildDuengungTable(m.duengung), padding: 0)
```

Listing 128: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/widgets/massnahmen_table.dart

```
Widget buildDuengungTable(Iterable<Duengung> duengung) {
  return Table(
    border: TableBorder.all(width: 0.5),
    defaultColumnWidth: const IntrinsicColumnWidth(),
    children: [
      TableRow(children: [
        _buildColumnHeader(const Text("Vorgabe")),
        _buildColumnHeader(const Text("Art")),
      ]),
      ...duengung
          .map((d) => TableRow(
                children: [
                  _buildColumnHeader(Text(d.vorgabe)),
                  if (d.art != null)
                    _buildColumnHeader(Text(d.art ?? ""))
                  else
                    _buildColumnHeader(const Text("")),
                ],
              ))
          .toList()
    ],
  );
}
```

Listing 129: XXXXX, Quelle: Eigenes Listing, Datei: Quellcode/Schritt-7/conditional_form/lib/widgets/massnahmen_table.dart

```
typedef ListTileTitleBuilder<ChoiceType extends Choice> = Widget Function(
    BuildContext context, ChoiceType choice, bool isSelected);

Widget _defaultBuildListTileTitleStrategy(context, Choice c, isSelected) {
  return Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [Text(c.description)],
  );
}
```

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

```
class MassnahmenFormViewModel {
    final letzterStatus = BehaviorSubject<LetzterStatus?>.seeded(null);
    // ...
}
```

Listing 134: Live Template für die Erstellung von built_value Boilerplate-Code in Android Studio, Quelle: Jetbrains Marketplace Built Value Snippets Plugin

```
class ChoiceChangeNotifier extends ChangeNotifier {
    BuiltSet<Choice> _choices = BuiltSet<Choice>();

    BuiltSet<Choice> get choices => _choices;

    set choices(BuiltSet<Choice> choices) {
        _choices = choices;
        notifyListeners();
    }
}
class LetzterStatusViewModel extends ChoiceChangeNotifier {}
```

Listing 135: Live Template für die Erstellung von built_value Boilerplate-Code in Android Studio, Quelle: Jetbrains Marketplace Built Value Snippets Plugin

```
providers: [
  Provider<MassnahmenFormViewModel>(
      create: (_) => MassnahmenFormViewModel()),
  Provider<MassnahmenJsonFile>(create: (_) => MassnahmenJsonFile()),
  Provider(
      create: (context) => MassnahmenPool(
          Provider.of<MassnahmenJsonFile>(context, listen: false))),
  ChangeNotifierProvider(create: (context) => LetzterStatusViewModel()))
],
```

Listing 136: Live Template für die Erstellung von built_value Boilerplate-Code in Android Studio, Quelle: Jetbrains Marketplace Built Value Snippets Plugin

```
Consumer<ConsumerType>(
    builder: (context, choiceChangeNotifier, child) {
      final selectedChoices = choiceChangeNotifier.choices;
      return Card(
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            ListTile(
              focusNode: focusNode,
              title: Text(title),
              subtitle:
              Text(selectedChoices.map((c) => c.description).join(",")),
              trailing: const Icon(Icons.edit),
              onTap: navigateToSelectionScreen,
            )
          ],
        ),
      );
    },
  )
```

Listing 137: Live Template für die Erstellung von built_value Boilerplate-Code in Android Studio, Quelle: Jetbrains Marketplace Built Value Snippets Plugin

Anhang

A Technologiewahl Anhang

A.1 Stimmen verwendeter Frameworks

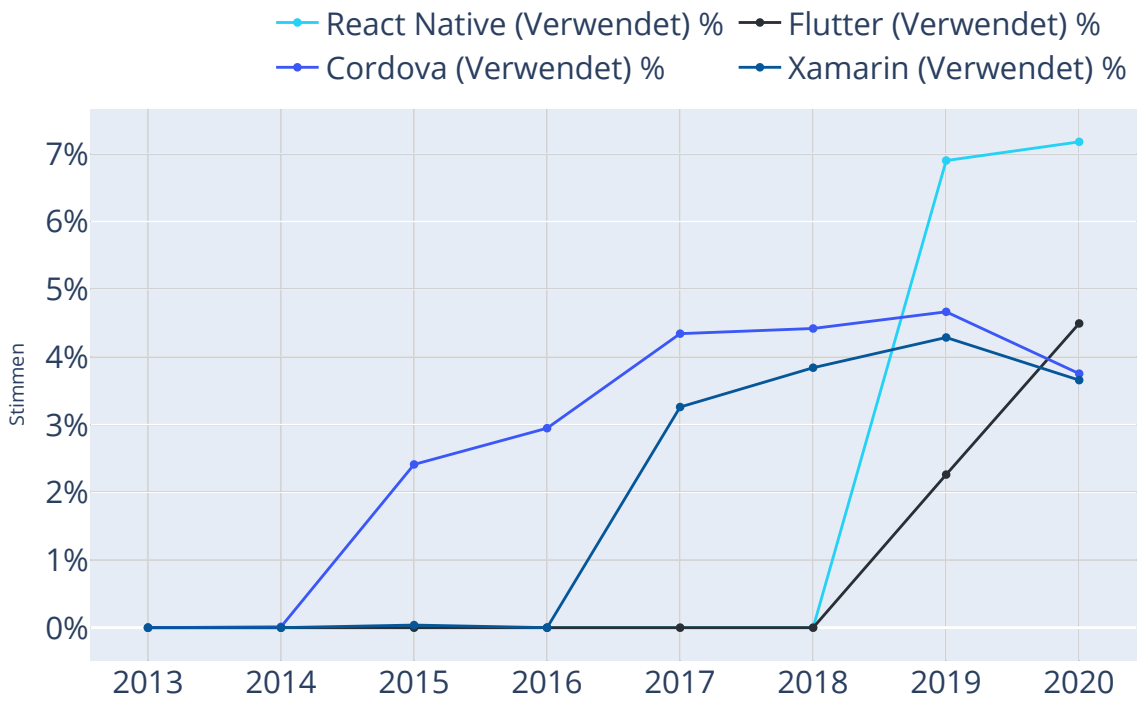

Abbildung 10: Stimmen verwendeter Frameworks, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

A.2 Stimmen gewünschter Frameworks

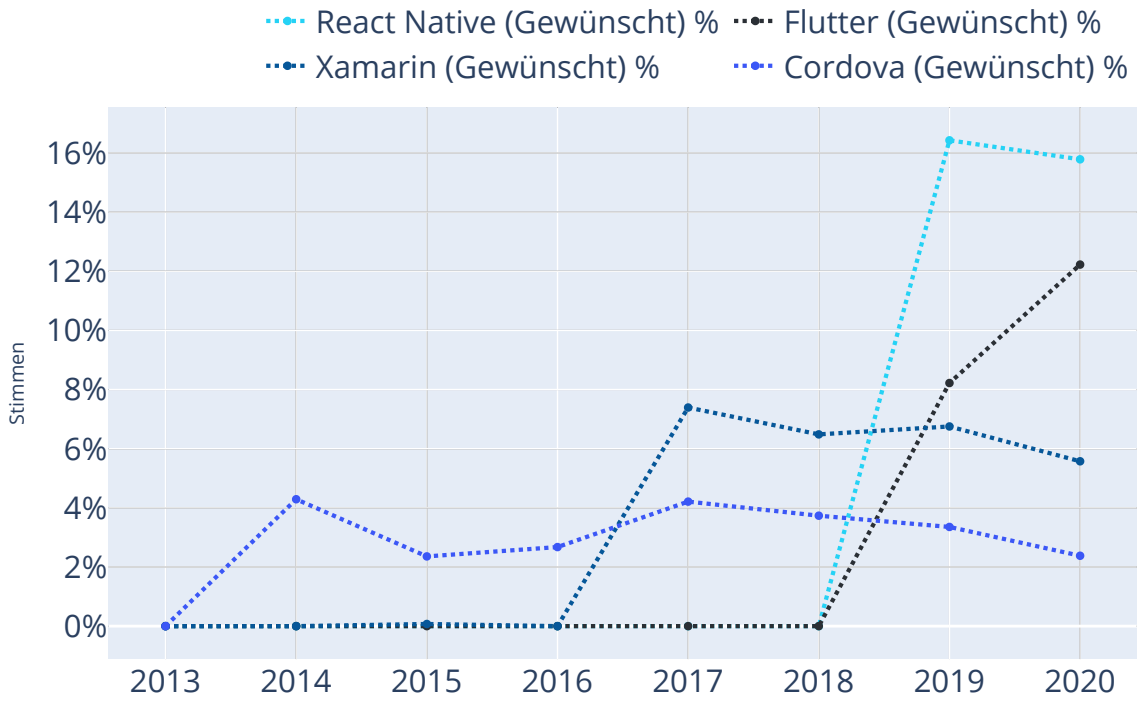

Abbildung 11: Stimmen gewünschter Frameworks, Quelle: Eigene Abbildung, Notebook: Charts/Stack Overflow Umfrage/Stack Overflow Umfrage.ipynb

B Vergleich React Native und Flutter Anhang

```dart
import 'package:flutter/material.dart';
import 'validation.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) => MaterialApp(
        home: Scaffold(
          body: MyCustomForm(),
        ),
      );
}

class MyCustomForm extends StatefulWidget {
  @override
  MyCustomFormState createState() => MyCustomFormState();
}

class MyCustomFormState extends State<MyCustomForm> {
  final _formKey = GlobalKey<FormState>();

  @override
  Widget build(BuildContext context) => Form(
        key: _formKey,
        child: Padding(
          padding: const EdgeInsets.all(8.0),
          child: Column(
            children: [
              TextFormField(
                  decoration: const InputDecoration(labelText: "Name"),
                  validator: (String? value) =>
                      validateNotEmpty("Name", value)),
              TextFormField(
                  decoration: const InputDecoration(labelText: "Email"),
                  validator: (String? value) => validateEmail("Name", value)),
              TextFormField(
                  decoration: const InputDecoration(labelText: "Password"),
                  validator: (String? value) =>
                      validateNotEmpty("Password", value)),
              Padding(
                padding: const EdgeInsets.symmetric(vertical: 16.0),
                child: ElevatedButton(
                  onPressed: () {
                    if (_formKey.currentState!.validate()) {
                      ScaffoldMessenger.of(context).showSnackBar(
                          SnackBar(content: Text('Processing Data')));
                    }
                  },
                  child: Text('Submit'),
                ),
              ),
            ],
          ),
        ),
      );
}
```

Listing 138: , Quelle: Eigenes Listing,
Datei: Quellcode/Vergleich/form-in-flutter/lib/main.dart

```dart
final emailPattern = RegExp(
    r'^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@ _⌋
    ↪  ((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])| _⌋
    ↪  (([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$');

String? validateEmail(String label, String? value) {
  if (value == null || value.isEmpty) {
    return '$label is required';
  } else if (!emailPattern.hasMatch(value)) {
    return 'Invalid Email Format';
  } else {
    return null;
  }
}

String? validateNotEmpty(String label, String? value) {
  if (value == null || value.isEmpty) {
    return '$label is required';
  } else {
    return null;
  }
}
```

Listing 139: , Quelle: Eigenes Listing,
Datei: Quellcode/Vergleich/form-in-flutter/lib/validation.dart

```tsx
import * as React from 'react';
import {
  Text,
  View,
  StyleSheet,
  Button,
  Alert,
  ScrollView,
} from 'react-native';
import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-view';

import Constants from 'expo-constants';
import { useForm } from 'react-hook-form';

// You can import from local files
import Input from './components/Input';
import Form from './components/Form';
import validation from './validation';
import Hero from './Hero';

type FormData = {
  name: string;
  email: string;
  password: string;
};

export default () => {
  const { handleSubmit, register, setValue, errors } = useForm<FormData>();

  const onSubmit = (data: FormData) => {
    Alert.alert('data', JSON.stringify(data));
  };

  return (
    <KeyboardAwareScrollView
      contentContainerStyle={styles.container}
      style={{ backgroundColor: '#181e34' }}>
      <Hero />
      <View style={styles.formContainer}>
        <Form {...{ register, setValue, validation, errors }}>
          <Input name="name" label="Name " />
          <Input name="email" label="Email" />
          <Input name="password" label="Password" secureTextEntry={true} />
          <Button title="Submit" onPress={handleSubmit(onSubmit)} />
        </Form>
      </View>
    </KeyboardAwareScrollView>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    paddingTop: Constants.statusBarHeight,
    backgroundColor: '#181e34',
  },
  formContainer: {
    padding: 8,
    flex: 1,
  },
  button: {
    backgroundColor: 'red',
  },
});
```

Listing 140: , Quelle: XXXXX todo HIGH PRIO: FÜLLEN XXXXX, Datei: Quellcode/Vergleich/form-in-react-native-the-right-way/App.tsx

```
import * as React from 'react';
import { Text, View, StyleSheet, Image } from 'react-native';

export default () => {
    return (
      <View style={styles.container}>
       <Image style={styles.logo} source={require('./assets/hero.jpg')} />
        <Text style={styles.paragraph}>
          Form in React Native, The right Way!
        </Text>
      </View>
    );
  }

const styles = StyleSheet.create({
  container: {
    justifyContent: 'center',
    flex:1,
  },
  paragraph: {
    margin: 24,
    marginTop: 0,
    fontSize: 34,
    fontWeight: 'bold',
    textAlign: 'center',
    color:'#FFF'
  },
  logo: {
   width:'100%',
   height:200
  }
});
```

Listing 141: , Quelle: XXXXX todo HIGH PRIO: FÜLLEN XXXXX,
Datei: Quellcode/Vergleich/form-in-react-native-the-right-way/Hero.tsx

```
export default {
  name: {required: {value: true, message: 'Name is required'}},
  email: {
    required: {value: true, message: 'Email is required'},
    pattern: {
      value: /^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@ _⌋
      ↪  ((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])| _⌋
      ↪  (([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/,
      message: 'Invalid Email Format',
    },
  },
  password: {
    required: {value: true, message: 'Password is required'},
  },
};
```

Listing 142: , Quelle: XXXXX todo HIGH PRIO: FÜLLEN XXXXX,
Datei: Quellcode/Vergleich/form-in-react-native-the-right-way/validation.tsx

```tsx
import * as React from 'react';
import { TextInput, KeyboardAvoidingView, findNodeHandle } from 'react-native';
import { ValidationOptions, FieldError } from 'react-hook-form';

interface ValidationMap {
  [key: string]: ValidationOptions;
}

interface ErrorMap {
  [key: string]: FieldError | undefined;
}

interface Props {
  children: JSX.Element | JSX.Element[];
  register: (
    field: { name: string },
    validation?: ValidationOptions
  ) => void;
  errors: ErrorMap;
  validation: ValidationMap;
  setValue: (name: string, value: string, validate?: boolean) => void;
}

export default ({
  register,
  errors,
  setValue,
  validation,
  children,
}: Props) => {
  const Inputs = React.useRef<Array<TextInput>>([]);

  React.useEffect(() => {
    (Array.isArray(children) ? [...children] : [children]).forEach((child) => {
      if (child.props.name)
        register({ name: child.props.name }, validation[child.props.name]);
    });
  }, [register]);

  return (
    <>
      {(Array.isArray(children) ? [...children] : [children]).map(
        (child, i) => {
          return child.props.name
            ? React.createElement(child.type, {
                ...{
                  ...child.props,
                  ref: (e: TextInput) => {
                    Inputs.current[i] = e;
                  },
                  onChangeText: (v: string) =>
                    setValue(child.props.name, v, true),
                  onSubmitEditing: () => {
                    Inputs.current[i + 1]
                      ? Inputs.current[i + 1].focus()
                      : Inputs.current[i].blur();
                  },
                  //onBlur: () => triggerValidation(child.props.name),
                  blurOnSubmit: false,
                  //name: child.props.name,
                  error: errors[child.props.name],
                },
              })
            : child;
        }
      )}
    </>
  );
};
```

Listing 143: , Quelle: XXXXX todo HIGH PRIO: FÜLLEN XXXXX,
Datei: Quellcode/Vergleich/form-in-react-native-the-right-way/components/Form.tsx

```tsx
import * as React from 'react';
import {
  View,
  TextInput,
  Text,
  StyleSheet,
  ViewStyle,
  TextStyle,
  TextInputProps,
} from 'react-native';
import { FieldError } from 'react-hook-form';
interface Props extends TextInputProps {
  name: string;
  label?: string;
  labelStyle?: TextStyle;
  error?: FieldError | undefined;
}

export default React.forwardRef<any, Props>(
  (props, ref): React.ReactElement => {
    const { label, labelStyle, error, ...inputProps } = props;

    return (
      <View style={styles.container}>
        {label && <Text style={[styles.label, labelStyle]}>{label}</Text>}
        <TextInput
          autoCapitalize="none"
          ref={ref}
          style={[styles.input, { borderColor: error ? '#fc6d47' : '#c0cbd3' }]}
          {...inputProps}
        />
        <Text style={styles.textError}>{error && error.message}</Text>
      </View>
    );
  }
);

const styles = StyleSheet.create({
  container: {
    marginVertical: 8,
  },
  input: {
    borderStyle: 'solid',
    borderWidth: 1,
    borderRadius: 5,
    paddingVertical: 5,
    paddingLeft: 5,
    fontSize: 16,
    height: 40,
    color: '#c0cbd3',
  },
  label: {
    paddingVertical: 5,
    fontSize: 16,
    fontWeight: 'bold',
    color: '#c0cbd3',
  },
  textError: {
    color: '#fc6d47',
    fontSize: 14,
  },
});
```

Listing 144: , Quelle: XXXXX todo HIGH PRIO: FÜLLEN XXXXX,
Datei: Quellcode/Vergleich/form-in-react-native-the-right-way/components/Input.tsx

Literatur

<Formik /> | Formik Docs API. Juni 2021. URL: https://web.archive.org/web/20210409184616if_/https://formik.org/docs/api/formik (Zitiert auf der Seite 17).

Adobe Inc. *FAQ | PhoneGap Docs*. Aug. 2016. URL: https://web.archive.org/web/20200806024626/http://docs.phonegap.com/phonegap-build/faq/ (Zitiert auf der Seite 14).

– *Update for Customers Using PhoneGap and PhoneGap Build*. Aug. 2020. URL: https://web.archive.org/web/20200811121213/https://blog.phonegap.com/update-for-customers-using-phonegap-and-phonegap-build-cc701c77502c?gi=df435eca31bb (Zitiert auf der Seite 15).

Borenkraout, Matan. *Native Testing Library Introduction | Testing Library Docs*. Nov. 2020. URL: https://web.archive.org/web/20210128142719/https://testing-library.com/docs/react-native-testing-library/intro/ (Zitiert auf der Seite 18).

Does redux-form work with React Native? Juni 2021. URL: https://web.archive.org/web/20210602234346/https://redux-form.com/7.3.0/docs/faq/reactnative.md/ (Zitiert auf der Seite 17).

Facebook Inc. *The React Native Ecosystem*. Juni 2021. URL: https://web.archive.org/web/20210602191504/https://github.com/facebook/react-native/blob/d48f7ba748a905818e8c64fe70fe5b24aa098b05/ECOSYSTEM.md (Zitiert auf der Seite 19).

Google LLC. *Build a form with validation*. Juni 2021. URL: https://web.archive.org/web/20210122020924/https://flutter.dev/docs/cookbook/forms/validation (Zitiert auf der Seite 18).

– *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 22, 23).

– *Forms | Flutter Docs Cookbook*. Juni 2021. URL: https://web.archive.org/web/20201102003629/https://flutter.dev/docs/cookbook/forms (Zitiert auf der Seite 18).

– *Häufig gestellte Fragen zu Google Trends-Daten - Google Trends-Hilfe*. Mai 2021. URL: https://support.google.com/trends/answer/4365533 (Zitiert auf der Seite 13).

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 den Seiten 22, 23).

Lynch, Max. *The Last Word on Cordova and PhoneGap*. März 2014. URL: https://web.archive.org/web/20210413012559/https://blog.ionicframework.com/what-is-cordova-phonegap/ (Zitiert auf der Seite 14).

React Native | Formik Docs. Juni 2021. URL: https://web.archive.org/web/20210507005917if_/https://formik.org/docs/guides/react-native (Zitiert auf der Seite 17).

React Native | React Hook Form - Get Started. Juni 2021. URL: https://web.archive.org/web/20210523042601if_/https://react-hook-form.com/get-started/ (Zitiert auf der Seite 17).

reduxForm | Redux Form - API. Juni 2021. URL: https://web.archive.org/web/20210506221401/https://redux-form.com/7.4.2/docs/api/reduxform.md/#-code-validate-values-object-props-object-gt-errors-object-code-optional- (Zitiert auf der Seite 17).

register | React Hook Form - API. Juni 2021. URL: https://web.archive.org/web/20210406032209/https://react-hook-form.com/api/useform/register (Zitiert auf der Seite 17).

Spolsky, Joel. „How Hard Could It Be?: The Unproven Path“. In: *inc.com* (Nov. 2008). URL: http://web.archive.org/web/20081108094045/http://www.inc.com/magazine/20081101/how-hard-could-it-be-the-unproven-path.html (Zitiert auf der Seite 12).

Stack Exchange, Inc. *Stack Overflow Insights - Developer Hiring, Marketing, and User Research*. Mai 2021. URL: https://insights.stackoverflow.com/survey/ (Zitiert auf den Seiten 12, 14).

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