Java-Low-Code-Plattform

René Jahn, SIB Visions GmbH

Die Begriffe Low-Code und No-Code sind zwei neue Sterne am Himmel. Im Jahre 2014 wurde Low-Code von Forrester Research benannt und erst in den letzten Monaten wurde richtig Fahrt aufgenommen. Fast wöchentlich tauchen neue Tools auf. Eine Empfehlung auszusprechen, ist nahezu unmöglich. Auch im Java-Umfeld gibt es viele Tools, die den Anforderungen von Low- und No-Code gerecht werden. Eines dieser Tools wird in diesem Artikel vorgestellt: VisionX [1].

Es gibt ausführliche Definitionen für Low-Code [2] und No-Code [3]. Der Autor versteht darunter Folgendes:

Eine Person, die nicht programmieren kann, soll mit einem No-Code-Tool eine lauffähige Applikation erstellen können, die den eigenen Anforderungen entspricht. Bei Low-Code-Tools sind die Applikationen mit eigenem Code erweiterbar. Dazu sollte der Anwender jedoch ein paar wenige Programmierkenntnisse mitbringen.

Als Applikation zählt sowohl eine mobile App für das Scannen eines Barcodes, als auch eine komplexe Verwaltungsanwendung, die im Browser läuft. Weder von Low-Code noch von No-Code wird vorgegeben, welche Art von Applikation erstellt werden kann. Das ist einzig und allein abhängig vom eingesetzten Tool.

Aus Sicht der Software-Entwickler/innen wären sowohl Low-Code als auch No-Code unnötig, denn genau das ist *ihre* Aufgabe. Doch das sehen die Fachabteilungen und IT-Entscheider ganz anders. Sie sind bemüht, Prozesse und Anforderungen so schnell wie möglich in Software abzubilden, um der Konkurrenz einen Schritt voraus zu sein. Nach Möglichkeit auch so günstig wie möglich.

Das ist aber häufig gar nicht realisierbar, weil die Entwicklungsabteilung keine oder zu wenige Ressourcen frei hat oder bereits mit anderen Aufgaben beschäftigt ist. Es ist auch nicht einfach möglich, das Entwicklungsteam aufzustocken. Zum einen ist es schwierig, eine geeignete Person zu finden, und zum anderen bedarf es einer gewissen Einarbeitungszeit. Ganz abgesehen von den Zusatzkosten.

Diese Probleme sollen mit Low-Code- und No-Code-Tools gelöst werden. Warum denn eigentlich nicht?

Aus Sicht des Autors spricht auch nichts dagegen, bestimmte Aufgaben von Laien oder technisch versierten Personen umsetzen zu lassen. Mit Excel funktioniert das bisher doch auch ganz gut und die Anwender erstellen komplexe Formeln und Eingabemöglichkeiten relativ schnell und problemlos. Erst wenn es um verstrickte Abläufe oder die Anbindung von Fremdsystemen geht, könnte es schnell vorbei sein mit Low-Code. Doch auch das ist zum Teil lösbar, wenn vorgefertigte Bausteine oder Vorlagen vom Tool angeboten werden.

Wenn der Laie wirklich nicht mehr weiterkommt und Hilfe benötigt, dann sollten Softwareentwickler/innen einspringen. Doch hier beginnt es wieder knifflig zu werden, denn ein Laie hat keine technischen Ansprüche. Die Lösung muss lediglich das tun, was gefordert wird, egal mit welcher Technologie oder mit welchem Framework das umgesetzt ist.

Ein/e Softwareentwickler/in hat hingegen konkrete Vorstellungen und arbeitet lieber mit der gewohnten IDE und bekannten Frameworks. Wenn das nicht möglich ist, kommt es häufig zu Konflikten und einer Abwehrhaltung gegenüber den Tools.

Um den/die Softwareentwickler/in anzusprechen, braucht es also ein Low-Code-Tool, das sich mit einer oder allen IDEs im Einklang befindet. Es sollte keine Probleme mit beliebigen Frameworks haben und auch einfach zu bedienen sein. Der Laie soll nicht abgeschreckt werden und die Anforderungen auch ohne Programmierkenntnisse lösen können. Doch auch im Idealfall braucht es eine gewisse Offenheit für Neues, denn Low-Code-Tools vereinfachen die Arbeitsweise und bringen bereits Lösungen für Alltagsprobleme mit. Diese können sich von den Lösungsansätzen der Entwickler/innen unterscheiden. Das bedeutet eine Anpassung der Arbeitsweise.

VisionX

Ein Tool, das den Ansprüchen von Laien und Entwickler/innen gerecht wird, ist VisionX. Dabei handelt es sich um ein mit Java entwickeltes Low-Code- und auch No-Code-Tool. Es verwendet ausschließlich Open-Source-Frameworks und setzt auf etablierte Standards. Die damit erstellten Applikationen verwenden ebenfalls ausschließlich Open-Source-Frameworks und können, sowohl im Tool selbst als auch mit beliebigen Java IDEs (Eclipse, IntelliJ, NetBeans), weiterentwickelt werden. Es funktioniert auch beides gleichzeitig. Als Entwickler/in wäre es somit möglich, eine Eingabemaske zusammenzuklicken, anschließend die Logik in der IDE umzusetzen und zugleich im Tool zu testen. Der Laie kann dann direkt damit arbeiten und gegebenenfalls das Layout an seine Wünsche anpassen und beispielsweise um Berichte erweitern.

Durch die Kombination von Low-Code und IDE können sowohl Software-Entwickler/in als auch Laie an einer Applikation arbeiten, ohne auf die jeweiligen Vorteile verzichten zu müssen. Da Java als Programmiersprache zum Einsatz kommt, gibt es keinerlei Einschränkungen hinsichtlich Frameworks. Die fertige Lösung steht also im Vordergrund und nicht die Technologie.

Was macht VisionX aber nun so besonders? Das ist eine berechtigte Frage, denn schließlich gibt es unzählige Plattformen und Low-Code-Tools.

Wie bereits erwähnt, setzt VisionX auf Open-Source-Frameworks und bietet die Möglichkeit, beliebige Java IDEs und Frameworks einzusetzen. Dadurch ist auch eine Unabhängigkeit zum Hersteller gegeben, denn die Entwicklung der Applikation ist auch ohne VisionX möglich. Mit VisionX können sowohl (native) mobile als auch Browser-basierte und selbst klassische Desktop-Applikationen erstellt werden. Jede Applikation läuft auf allen Plattformen. Es ist nicht nötig, eine Applikation für mobile Geräte und eine eigene Applikation für Browser zu erstellen. Es handelt sich immer um ein und dieselbe Applikation. Durch den Einsatz des Java-Applikations-Frameworks JVx [4] wird dies ermöglicht.

Mit VisionX werden datengetriebene Verwaltungsapplikationen erstellt, die immer auf relationale Datenbanken zugreifen. Von VisionX werden gängige Anbieter wie Oracle, MySQL, MSSQL, PostgreSQL, DB2 und mehr unterstützt. Auch darin liegt ein großer Vorteil, denn die Datenbank ist frei wählbar. Es entsteht auch hier keine Abhängigkeit zum Tool. Doch noch viel wichtiger ist, dass die Installation von Applikationen auf beliebigen Java-Applikationsservern möglich ist und keine Kosten für den Betrieb anfallen. Man zahlt für die Verwendung des Tools und nicht für die Anzahl der Benutzer oder verwendete Ressourcen.

Neben der Flexibilität und Unabhängigkeit gibt es auch auf der Feature-Seite einige relevante Punkte, die ein Alleinstellungsmerkmal darstellen können. So ist beispielsweise ein Applikationsrahmen vorhanden, der eine Authentifizierung bietet, ein Benutzer- und Rollen-

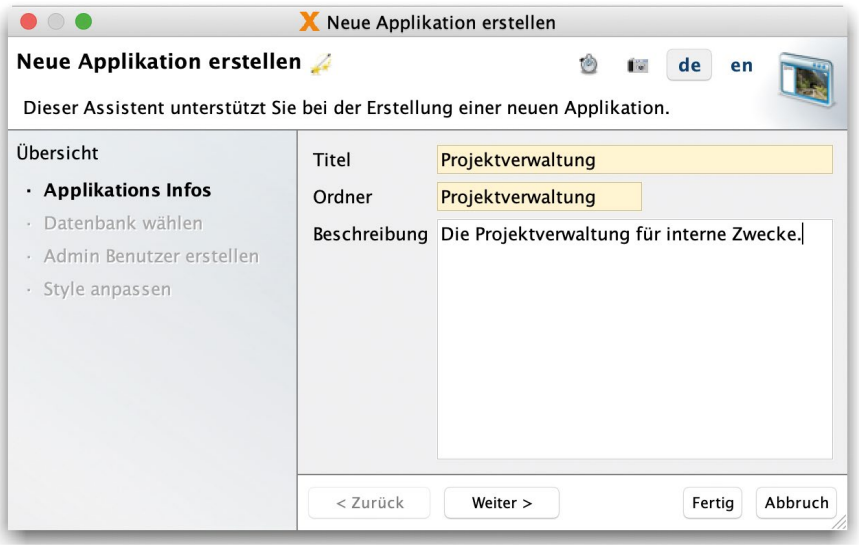

Abbildung 1: Neue Applikation erstellen (Quelle: René Jahn)

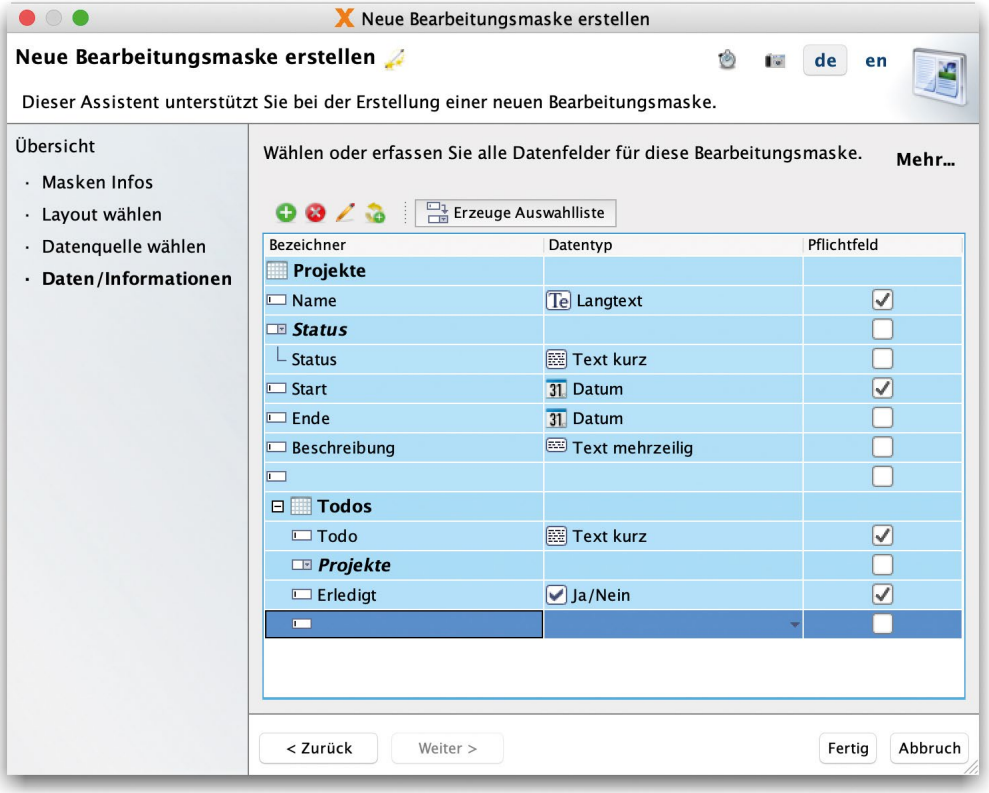

Abbildung 2: Neue Bearbeitungsmaske erstellen (Quelle: René Jahn)

konzept vollständig umsetzt und Mehrsprachigkeit ermöglicht. Der Rahmen ist komplett responsive und passt sich automatisch an die Ausgabegröße an. Ein richtig praktisches Feature ist die Live-Voransicht einer Applikation. Ob im Browser, mobil oder am Desktop. Das spart wertvolle Zeit und der Anwender sieht sofort das Endergebnis. Auf die Live-Preview auf mobilen Geräten werden wir etwas später noch zurückkommen.

Um einen Eindruck von der Arbeit mit VisionX zu bekommen, werden wir eine triviale Projekteverwaltung erstellen. Dazu wird lediglich eine einzige Bearbeitungsmaske benötigt, in der eine Liste von Projekten verwaltet werden kann. Jedes Projekt hat einen Namen, eine Beschreibung, einen Status, Start-/End-Datum und eine To-do-Liste mit To-do-Name und Erledigt-Option.

Zu Beginn erstellen wir eine neue Applikation mit dem Namen „Projektverwaltung", wie in *Abbildung 1* zu sehen ist. Die Vergabe eines Titels ist vollkommen ausreichend.

Nachdem die Applikation erstellt wurde, erscheint ein Assistent, mit dem eine neue Bearbeitungsmaske erstellt werden kann *(siehe Abbildung 2)*.

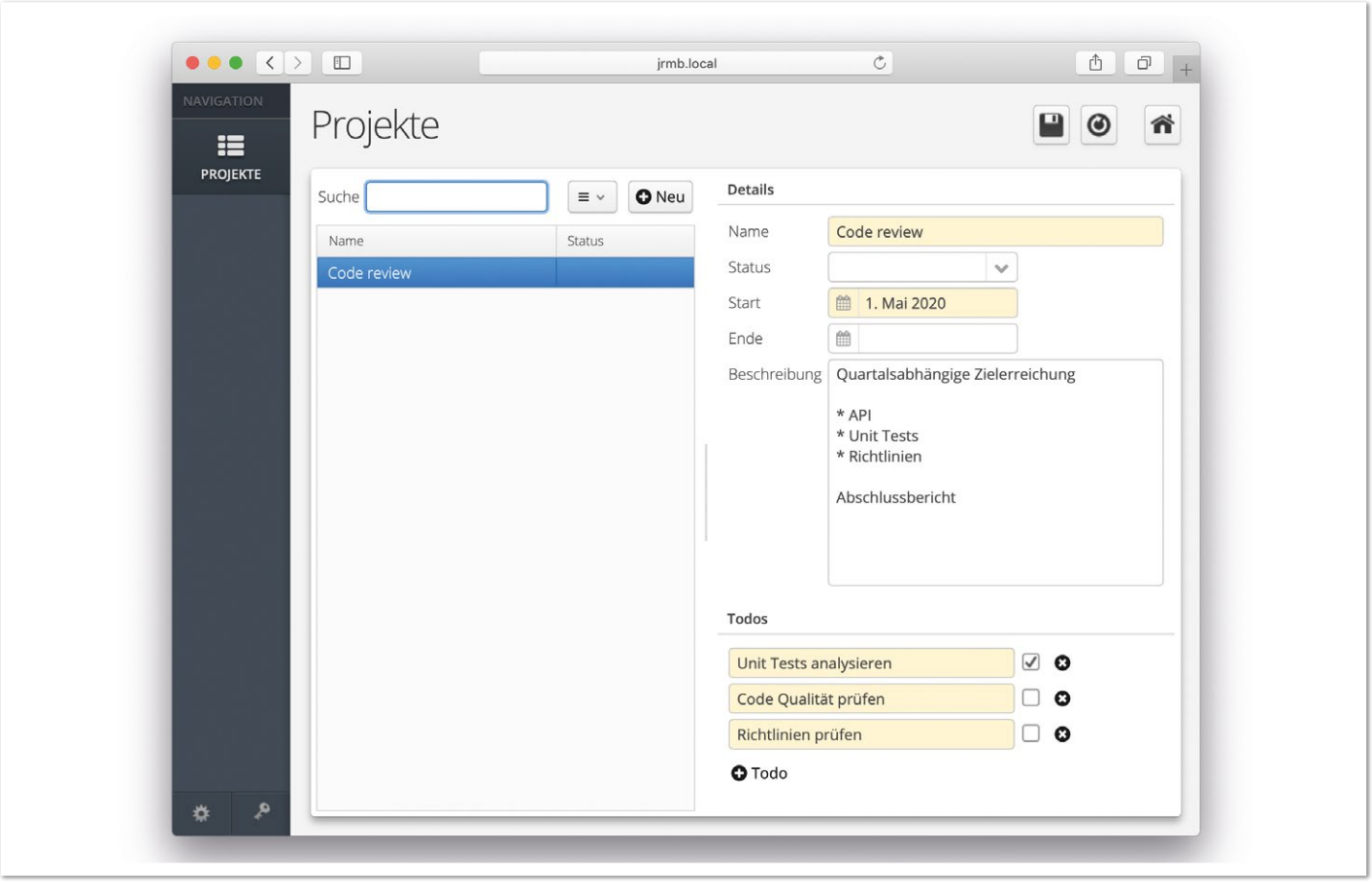

Abbildung 3: Live-Voransicht (Quelle: René Jahn)

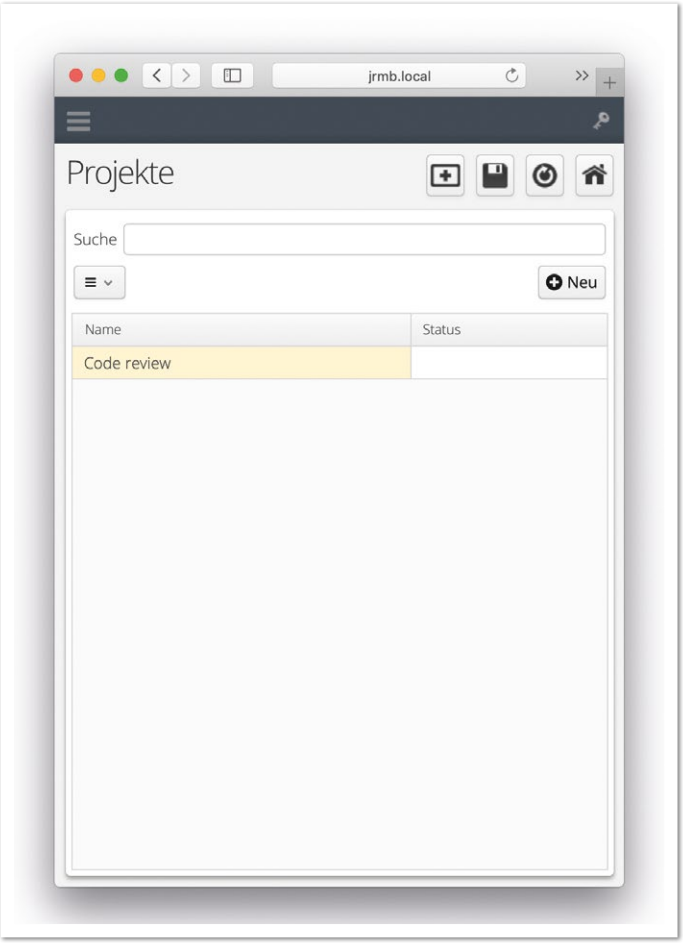

Abbildung 4: Minimale Anzeige, Projektliste (Quelle: René Jahn)

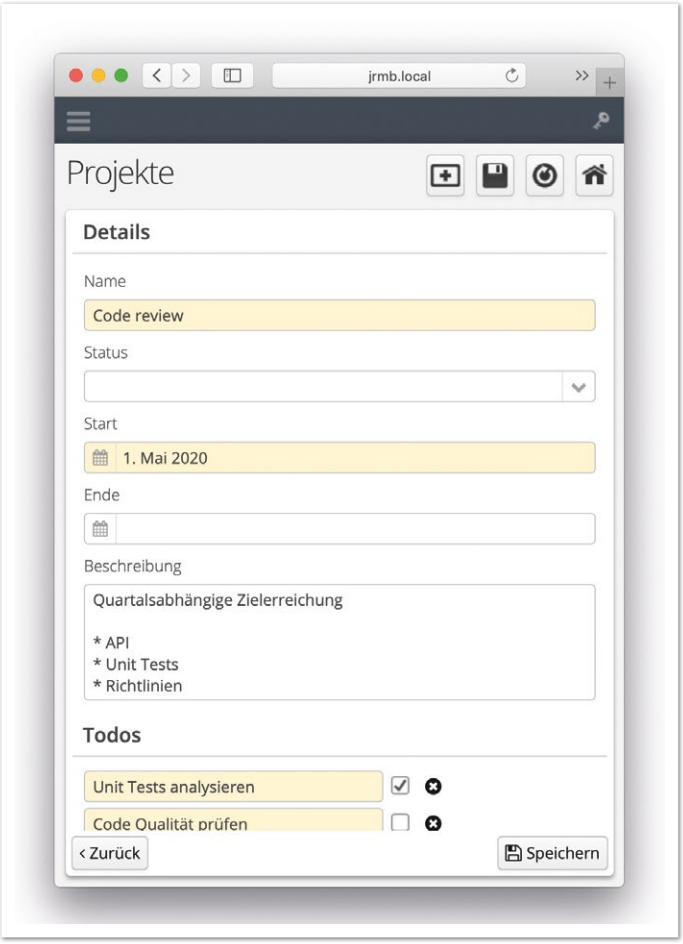

Abbildung 5: Minimale Anzeige, Detail (Quelle: René Jahn)

In diesem Assistenten werden ganz einfach die benötigten Felder eingegeben. Das Status-Feld wird als Auswahlliste und die To-dos werden als Sub-Tabelle definiert. Gleich im Anschluss wird die Maske erstellt; in der Voransicht sieht diese dann wie in *Abbildung 3* gezeigt aus.

In knapp einer Minute wurde in wenigen Schritten eine vollständige Applikation mit Authentifizierung und Bearbeitungsmaske erstellt. Wenn das Ausgabegerät etwas kleiner wäre, beispielsweise ein Smartphone, dann würde die Applikation wie in *Abbildung 4* angezeigt werden. Bei Auswahl eines Datensatzes wird dann das Detail angezeigt, siehe *Abbildung 5*.

Eigener Code

Für Entwickler/innen wäre die Erstellung dieser Applikation inklusive Bearbeitungsmaske ohne einen Code-Generator – in dieser Zeit – nicht möglich gewesen. Auch nicht mit Copy & Paste von vorhandenem Code. Hier spielt ein Low-Code-Tool natürlich seine ganze Stärke aus.

Doch wie ist das nun mit eigenem Code? Die Maske enthält aktuell keinerlei Logik. Nehmen wir mal an, dass es im Meeting-Raum unseres Büros einen großen Monitor gibt, der die aktuellen Projekte des Unternehmens, grafisch aufbereitet, anzeigt. Dieser Monitor soll bei Button-Klick in unserer Maske all unsere Projekte anzeigen. Die Schnittstelle zum Monitor ist nicht standardisiert und somit wird eine spezielle Anbindung benötigt.

Diese Aufgabe kann nur ganz schwer von einem Laien gelöst werden. Er kann aber einen Button integrieren und die Aufgabe von dem/der Softwareentwickler/in lösen lassen. Die Bearbeitungsmaske wird also um einen Button erweitert, wie in *Abbildung 6* unterhalb der Projektliste zu sehen ist.

Der/die Softwareentwickler/in bevorzugt eine IDE, deshalb wird das Projekt ganz einfach importiert. In unserem Fall kommt Eclipse zum Einsatz *(siehe Abbildung 7)*.

Für die IDE ist das Projekt bereits vorkonfiguriert; einzig und allein die Funktion beim Ausführen des Buttons *(siehe Listing 1)* muss implementiert werden.

Als Entwickler/in kann man nun seiner Fantasie freien Lauf lassen. Jeglicher Code, der eingefügt wird, kann von VisionX ausgelesen werden. Der Laie sieht dann lediglich, dass benutzerdefinierter Code verwendet wurde *(siehe Abbildung 8)*. Es kann aber ohne Einschränkungen gearbeitet werden. Es ist auch möglich, strukturelle Änderungen oder Namensänderungen am Code durchzuführen. Es gibt keine speziellen Bereiche, die nicht geändert werden dürfen. Von VisionX wird immer der Sourcecode gelesen und interpretiert.

Somit ist die Applikation auch schon fertig. Durch die Zusammenarbeit von Laie und Entwickler/in kann ein perfektes Ergebnis erzielt werden – ohne technologische Einschränkungen. Die aufgewendete Zeit ist mit manueller Codierung nicht vergleichbar und um ein Vielfaches effizienter. Der Testaufwand kann ebenfalls auf ein Minimum reduziert werden, denn die Basisfunktionalität wird bereits durch die Plattform gewährleistet. Im konkreten Beispiel wäre nur die Funktion zum Übertragen der Daten an den Monitor zu testen.

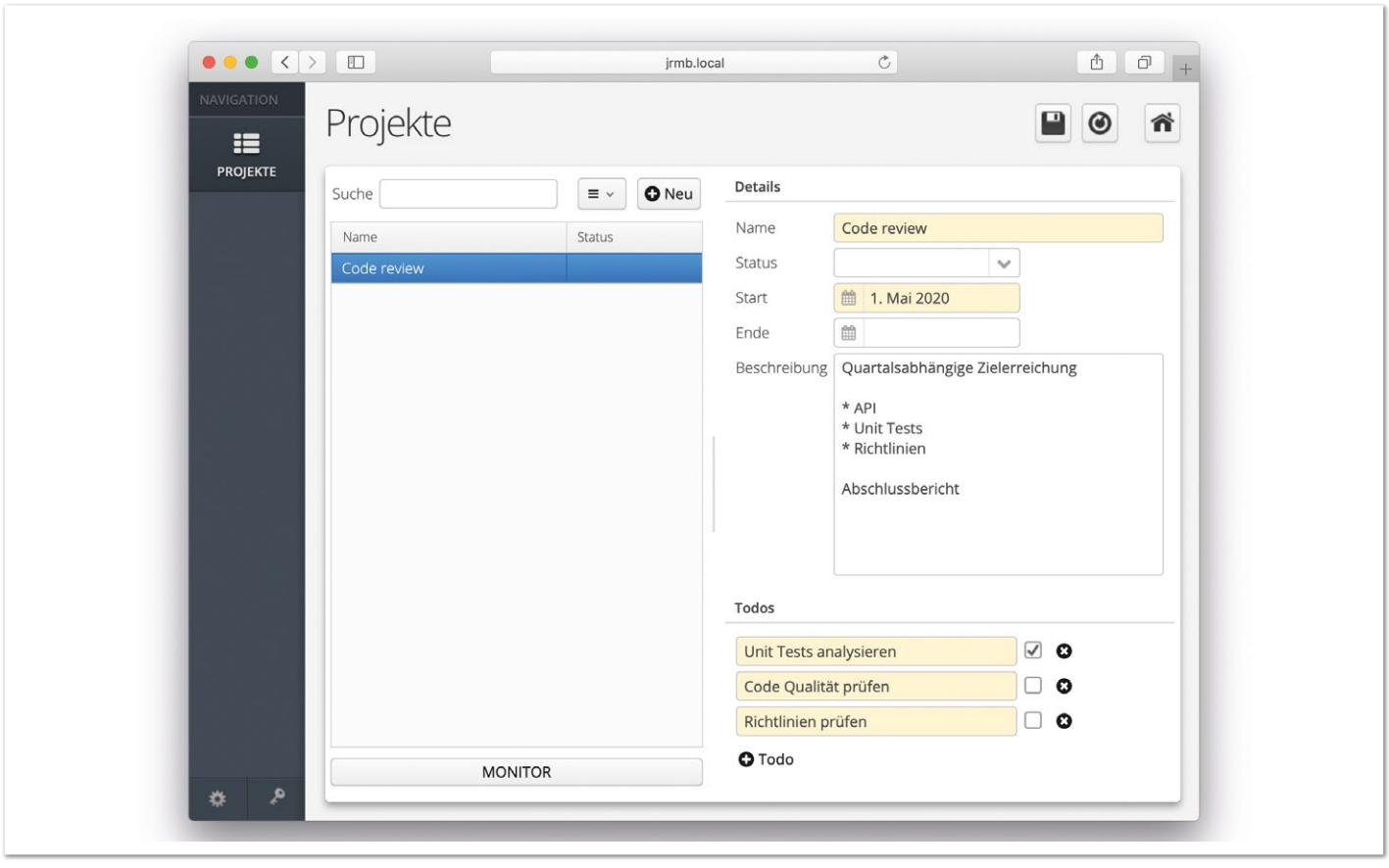

Abbildung 6: Button (Quelle: René Jahn)

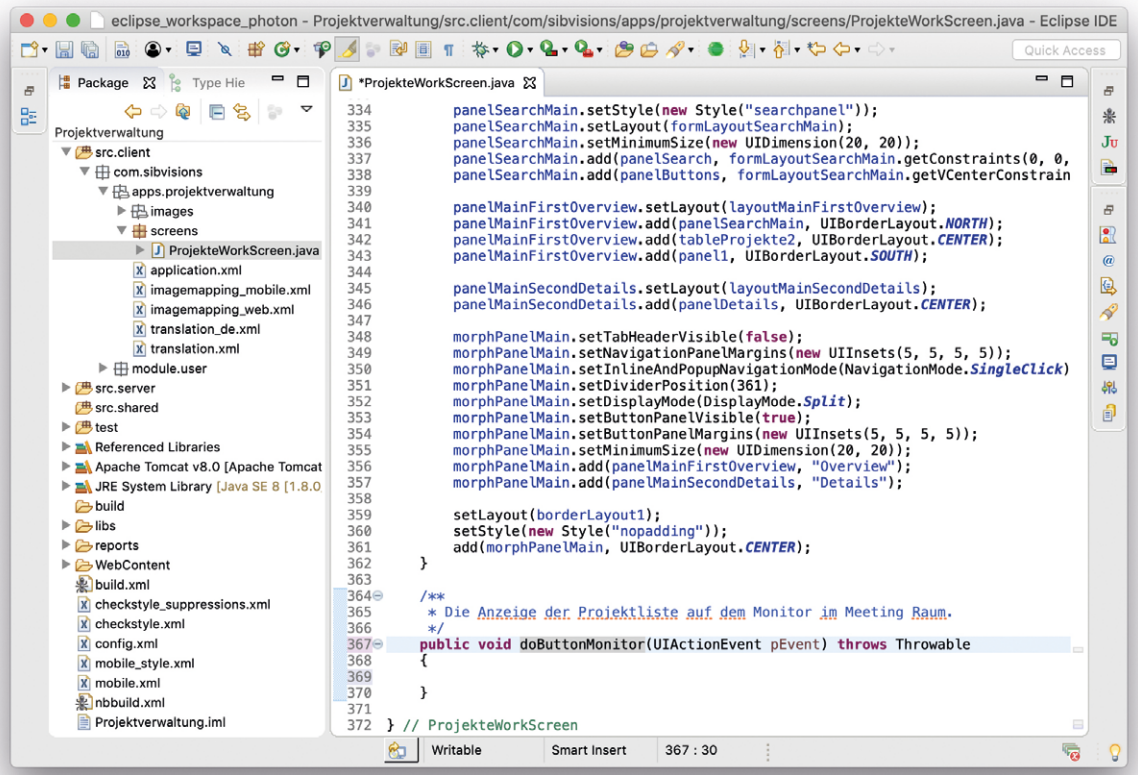

Abbildung 7: Projektverwaltung in Eclipse (Quelle: René Jahn)

```
/**
 * Die Anzeige der Projektliste auf dem Monitor im Meeting Raum.
 */
public void doButtonMonitor(UIActionEvent pEvent) throws Throwable
{
    System.out.println("Mein Code");
}
```

Listing 1: Button Action

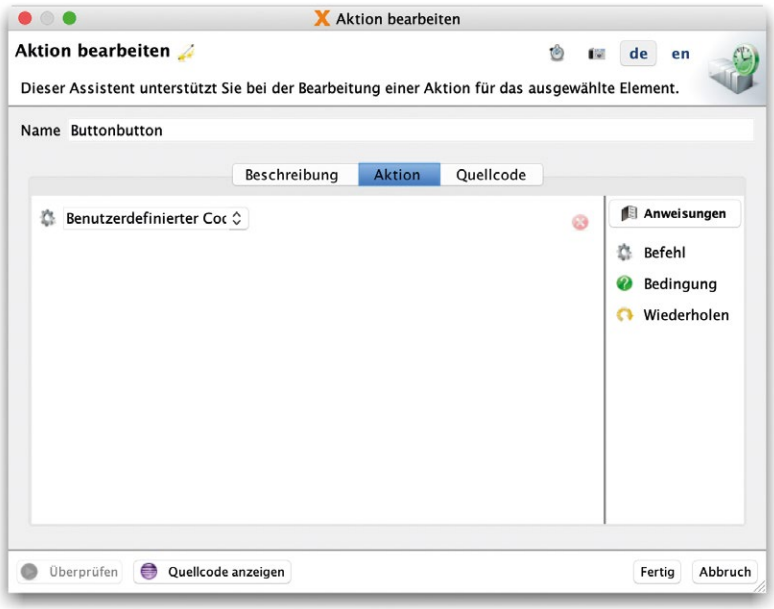

Abbildung 8: Benutzerdefinierter Code (Quelle: René Jahn)

Mobile Live Preview

Wie bereits erwähnt, kann eine Applikation auch als (native) mobile Applikation gestartet werden. Das kann mit einem echten Smartphone erfolgen, wie in *Abbildung 9* zu sehen ist.

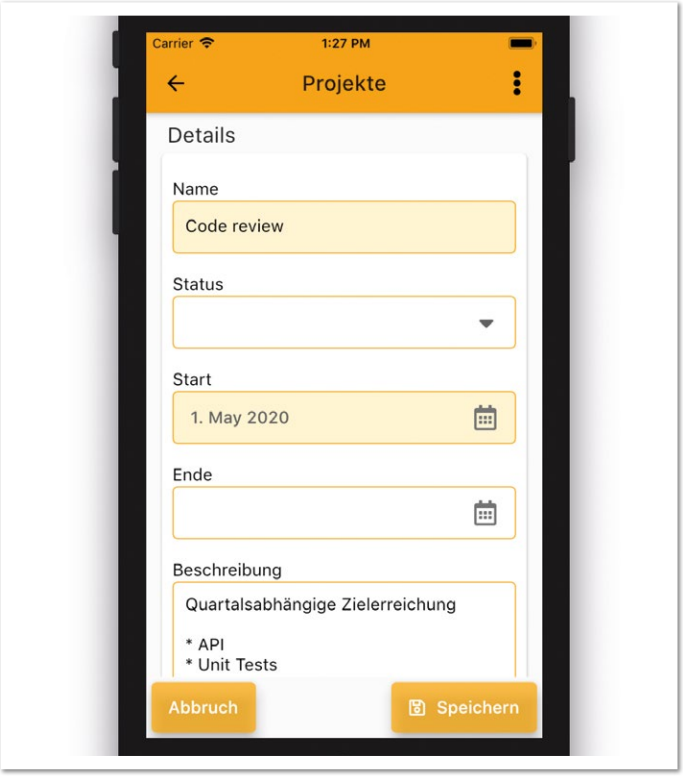

Abbildung 9: Live-Voransicht, Smartphone (Quelle: René Jahn)

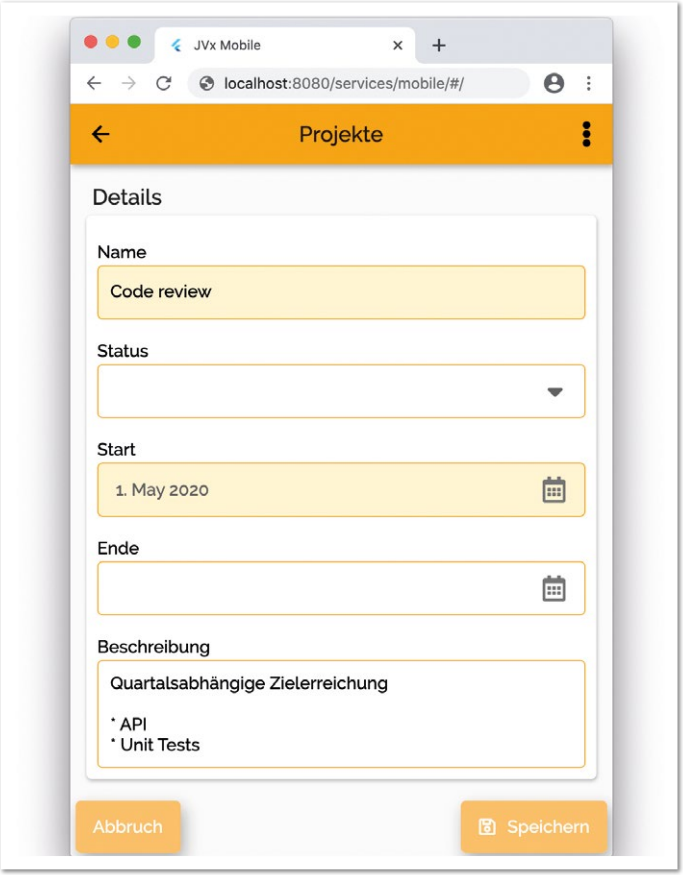

Abbildung 10: Live-Voransicht, Browser (Quelle: René Jahn)

Wer den Artikel „Es fluttert gewaltig" in der Java aktuell 3/20 *[5]* gelesen hat, wird feststellen, dass hier eine Flutter-Applikation *[6]* zum Einsatz kommt. Für diejenigen, die den Artikel nicht gelesen haben, sei kurz erwähnt, dass die in *Abbildung 9* dargestellte Applikation keine Java-Anwendung ist, sondern eine Flutter-Applikation, die direkt am mobilen Gerät läuft. In Ausgabe 3/20 wurde darauf hingewiesen, dass für Flutter auch die Möglichkeit besteht, eine HTML5-Applikation zu erstellen, mit derselben Source-Basis. Doch zum damaligen Zeitpunkt war der Entwicklungsstand noch nicht weit genug.

Darum möchte der Autor in dieser Ausgabe die HTML5-Applikation *(siehe Abbildung 10)* zeigen. Diese wird in VisionX für die Live-Preview verwendet, falls kein Smartphone zur Verfügung steht, oder einfach nur, weil es bequemer ist. Die Darstellung unterscheidet sich im Browser nicht von der Darstellung am Smartphone. Die Lösung ist einzigartig und kommt nur in VisionX zum Einsatz.

Fazit

Der Einsatz von Low-Code-Plattformen wäre für viele Bereiche der Softwareentwicklung ein enormer Boost. Wenn Fachabteilung und Entwicklung Hand in Hand arbeiten, dann entstehen Lösungen, die genauso sind, wie sie in der Praxis benötigt werden. Die Entwicklungszeit reduziert sich und Anpassungen können zeitnah umgesetzt werde, auch ohne Entwickler/in. Aber es ist auch Vorsicht geboten, denn wenn Entwickler/innen nicht mit den Fachabteilungen zusammenarbeiten, entstehen Insellösungen, die aus der Reihe tanzen. Auf die Offenheit von Low-Code-Plattformen sollte geachtet werden, denn Unabhängigkeit ist ein wichtiger Faktor für die Zukunft.

Quellen

[1] *https://visionx.sibvisions.com/*
[2] *https://de.wikipedia.org/wiki/Low-Code-Plattform*
[3] *https://en.wikipedia.org/wiki/No-code_development_platform*
[4] *https://de.wikipedia.org/wiki/JVx*
[5] *https://mydoag.doag.org/formes/pubfiles/12236053/docs/Publikationen/Java-Aktuell/2020/03-2020/03-2020-Java_aktuell-Magazin-WEB[1].pdf*
[6] *https://flutter.dev/*

René Jahn
SIB Visions GmbH
rene.jahn@sibvisions.com

René Jahn ist Mitbegründer der SIB Visions GmbH und Head of Research & Development. Er verfügt über langjährige Erfahrung im Bereich der Framework- und API-Entwicklung. Neben der Open-Source-Sparte bringt er seine Expertise auch in der Produktentwicklung ein. Seine Leidenschaft ist die Evaluierung neuer Technologien und Integration in klassische Business-Applikationen.