Aufgabe

Programmiere eine App, die deine ToDo Listen verwaltet.

In diesem Tutorial erfährst du, wie du in deiner App die folgenden Funktionen umsetzt:

- Anlegen einer ToDo Liste mit einem selbstgewählten Namen
- Übersicht der angelegten Listen
- Bearbeiten einer Liste
- Hinzufügen eines Eintrags zu einer ToDo Liste
- Einträge löschen und bearbeiten
- Löschen einer ToDo Liste

Design der App

Erstelle zunächst das Design der App. Die Programmierung wird danach ergänzt.

Achte im Design auf die Benennung der Elemente entsprechend der Vorgabe. Das ist wichtig, damit du diese in der Programmierung wiederfindest.

1. Screen: Screen1

Die To-Do Listen App besteht aus verschiedenen Screens. Der erste Screen nennt sich *Screen1* und begrüßt den Benutzer in deiner App.

Diesen Screen darfst du gestalten, wie du magst. Wichtig ist, dass es einen Button gibt mit den Namen: *Button_Los*

Oben links im App Inventor kannst du Screens hinzufügen mit "Add Screen …"
Per Klick auf den Screen Namen kannst du zwischen allen erstellten Screens wechseln.

2. Screen: Verwaltung

Dieser Screen nennt sich *Verwaltung*. Dort gibt es eine Übersicht über die angelegten Listen und du kannst neue Listen anlegen.

Auf der rechten Seite siehst du, welche Komponenten du auf diesem Screen benötigst.
Wundere dich nicht, die *TinyDB1* kann man nicht sehen, deshalb ist sie links nicht auf dem Bild.

Design der App

3. Screen: Liste

Dieser Screen zeigt die jeweils geöffnete Liste mit ihren Einträgen. Nenne ihn deshalb *Liste*.

Hier kannst du später die Einträge hinzufügen, löschen und bearbeiten. Und du kannst die gesamte Liste löschen.

Klickst du einen einzelnen Eintrag in der *ListView1* an, erscheint ein sogenannter Notifier. Dies ist ein kleines Pop-Up-Fenster, das dich fragt, was du tun möchtest.

Hinweis: Für jede Frage / Entscheidung ist ein eigener Notifier nötig.

← *Notifier_Bearb_Loe*

Tipp: Das Element *Notifier* findest du in der Palette unter *User Interface* ziemlich in der Mitte.

Programmierung der App

1: Begrüßung in der App

Der Benutzer landet beim Öffnen der App immer auf dem Screen, den du als erstes erstellt hast. Dieser nennt sich immer *Screen1*. Diesen Namen kannst du auch nicht ändern. Nutze den Screen, um den Benutzer zu begrüßen und um ihm zu erklären, was er tun kann.

Mit dem *Button_Los* wird der Benutzer auf den nächsten Screen weitergeleitet. Das soll der Screen mit der Übersicht über die angelegten Listen sein, also *Verwaltung*.

So sieht der **Code** dafür aus:

Das ist auch der gesamte Code, den du für den *Screen1* benötigst. Mehr passiert dort nicht.

Programmierung der App

2: Datenbank - Einführung

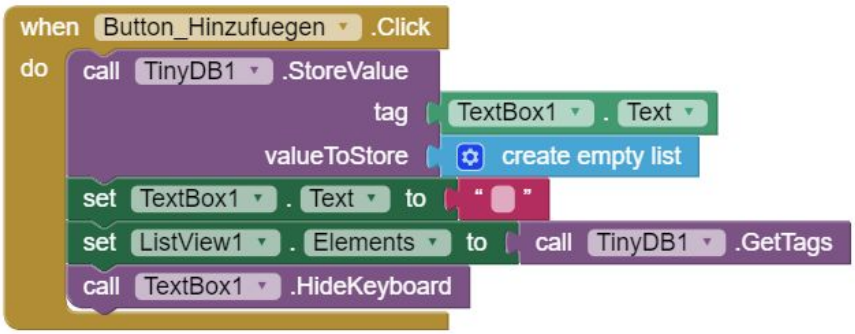
Die Listen werden in der Datenbank *TinyDB1* gespeichert. So bleiben die gespeicherten Daten auch erhalten, wenn du deine App schließt. Einfache Variablen können dies nicht. Die TinyDB1 wird von allen Screens gemeinsam genutzt. Du musst also keine Daten zwischen Screens übertragen.

In einer TinyDB gibt es *Tags* und *Values*. Ein Tag ist so etwas wie ein Kennzeichen, unter dem ein Value - auf deutsch: Wert - abgelegt wird. Um auf den Wert wieder zuzugreifen, sucht man dann einfach nach dem Kennzeichen.
Hier ein Bespiel: Tag = “Einkaufsliste” & Value = “Leere Liste” (die später befüllt wird)

3: Liste anlegen

Gehe zum Screen *Verwaltung* und klicke dort auf *Blocks*.

- Um eine neue Liste in der Datenbank zu speichern, lese den Listentitel aus der Textbox aus (als Tag) und eine leere Liste (als Value).
- Beim Klick auf den *Button_Hinzufuegen* speicherst du das, was in die Textbox eingegeben wurde, als Tag in der Datenbank.
- Danach löschst du den Inhalt der *Textbox1* wieder, indem du ihr einen leeren Text zuweist. So ist sie wieder bereit, sodass etwas neues eingegeben werden kann.
- Außerdem aktualisiere die *ListView1*, um auch die neue Liste in der Übersicht anzuzeigen. Das machst du, indem du aus der Datenbank alle Tags abfragst.
- Damit nach dem Hinzufügen der Liste die Tastatur nicht mehr den halben Bildschirm verdeckt, kannst du sie zum Schluss verstecken.



4: Übersicht anzeigen

Wenn die App geöffnet wird, sollen als erstes schon alle gespeicherten Listentitel angezeigt werden. Frage dazu bei der Initialisierung des Screens *Verwaltung* wieder alle Tags aus der Datenbank ab.

Programmierung der App

5: Liste bearbeiten

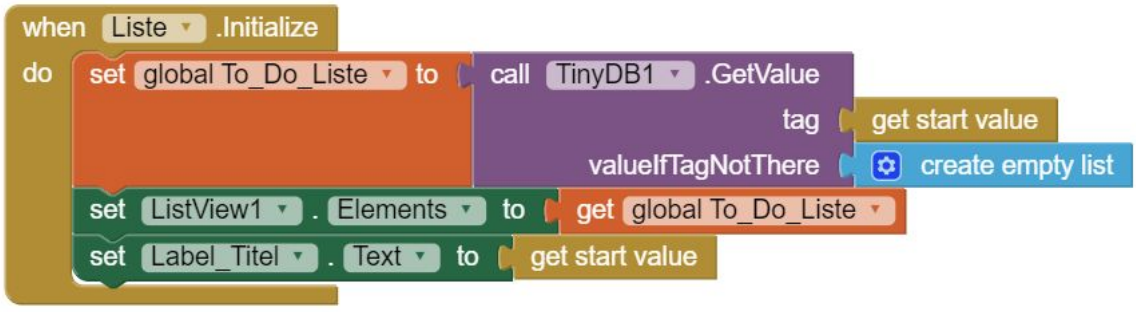
Wenn eine Liste in der Übersicht ausgewählt wird, sollen ihre Einträge angezeigt werden. Dazu öffne den anderen Screen *Liste*. Wichtig ist, dass als Startwert der ausgewählte Listentitel übergegeben wird, damit der andere Screen auch weiß, welche List angezeigt werden soll.

Wechsle zum Screen *Liste* in der Blockansicht.

6: Liste anzeigen

Um möglichst einfach mit der Liste arbeiten zu können, erstelle eine Variable *To_Do_Liste*, in der du die Liste zwischenspeicherst. Bei der Erstellung der Variablen ist die Liste natürlich noch leer.

Der Screen *Liste* kann alle Listen anzeigen. Welche es jeweils sein soll, wurde als startValue (siehe Code oben) übergeben. Hole anhand von diesem Startwert aus der Datenbank die Liste mit den bisher eingetragenen Elementen und speichere sie in der Variablen *To_Do_Liste*. Diese wird dann an die ListView1 übergeben, damit die Listenelemente auch angezeigt werden.



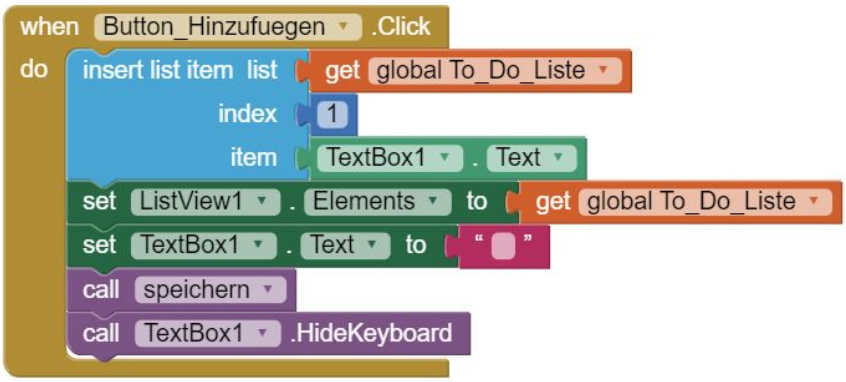
7: Liste speichern

Erstelle eine Prozedur *speichern*. Diese brauchst du gleich, um Änderungen wieder in der Datenbank zu speichern. Der zu speichernde Wert ist der Inhalt der Variablen *To_Do_Liste* und das Tag ist der Listentitel, der noch im Startwert zu finden ist.

Programmierung der App

8: Eintrag hinzufügen

Um einen neuen Eintrag in der Liste einzufügen, ergänze ihn in der Variablen *To_Do_Liste*. Füge ihn am besten vorn ein (index = 1), damit er in der Liste ganz oben angezeigt wird. Ähnlich wie beim Anlegen einer neuen Liste unter Punkt 3 musst du die *ListView1* aktualisieren und die *TextBox1* wieder leeren. Rufe außerdem die eben erstellte Funktion *speichern* auf, um Änderung an der Liste in der Datenbank zu sichern. Verstecke zuletzt noch die Tastatur, damit sie nicht im Weg ist, nachdem der Eintrag hinzugefügt wurde.

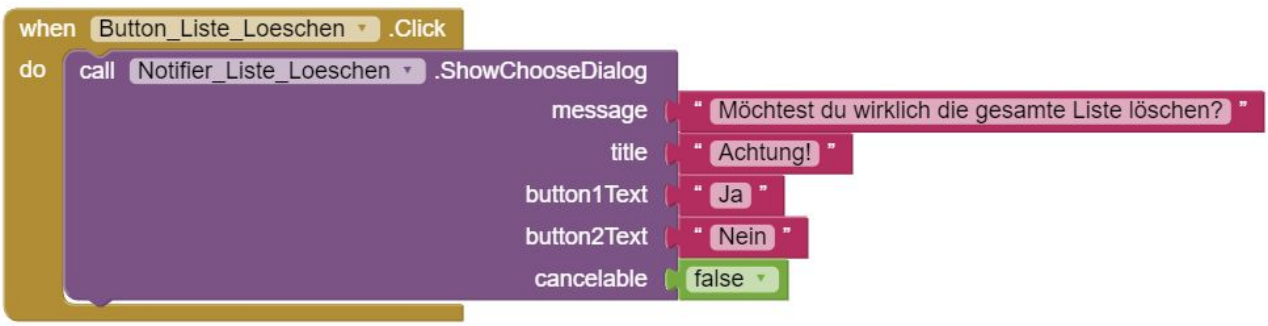


9: Zurück zur Übersicht

Um aus der Ansicht einer einzelnen Liste zurück zur Übersicht aller Listen zu gelangen, schließe einfach den aktuellen Screen *Liste* mit dem folgenden Block.

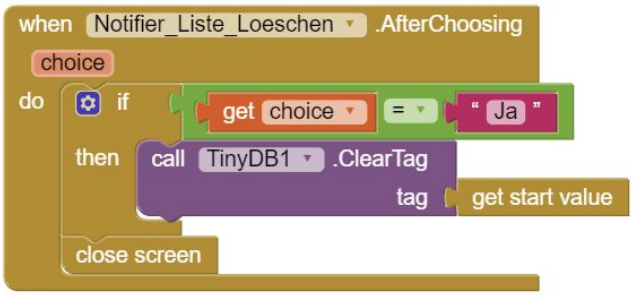
10: Liste löschen

Bevor etwas gelöscht wird, solltest du den Benutzer immer fragen, ob er sich sicher ist, dass diese Aktion durchgeführt werden soll, denn sie kann nicht rückgängig gemacht werden. Nutze dafür den *Notifier_Liste_Loeschen*. Diesem gibst du einen Text mit, der dem Benutzer angezeigt werden soll und zwei mögliche Antworten, von denen er eine auswählen kann. Je nachdem, welcher Antwortbutton ausgewählt wird, soll die App dann reagieren. Um nicht zusätzlich zur Antwortmöglichkeit “Nein” einen Button mit “Cancel” (Abbrechen) zu haben, setzt du *cancelable* auf “false”.



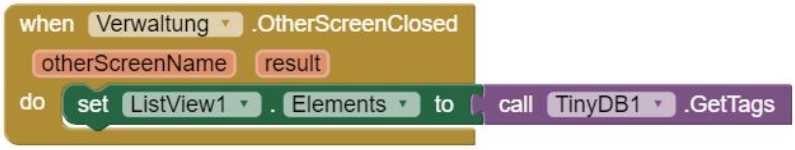
Programmierung der App

Der *Notifier_Liste_Loeschen* stellt die Variable *choice* zur Verfügung. Diese kannst du verwenden, um zu ermitteln, welche Antwort der Benutzer ausgewählt hat.
Wurde die Antwort “Ja” ausgewählt, lösche den Tag mit dem aktuellen Listentitel aus der Datenbank. Dieser ist noch immer als Startwert gespeichert. Schließe danach den Screen, denn die Liste, die angezeigt wird, existiert nun nicht mehr.
Lautet die Antwort “Nein” soll gar nichts gemacht werden, also wird der Fall ignoriert.



Wechsle zum Screen *Verwaltung*.

Da auf dem Screen *Liste* eventuell Änderungen an der Datenbank geschehen sind, bevor der Screen geschlossen wurde, aktualisiere hier die Anzeige der Listentitel wie folgt.



App testen und weitere Features entwickeln

Wow! Das waren jetzt schon einige ziemlich komplexe Programmierschritte.

- **Deine App sollte nun funktionieren. Teste sie bitte.**
- **Dir wird auffallen, dass es an einigen Stellen noch Verbesserungsbedarf gibt.**

Hier findest du ein paar Ideen, wie du die App weiterentwickeln kannst. Versuche die Aufgaben zunächst selbst zu lösen. Die Lösungen findest du auf den nächsten Seiten.

a) Füge eine Löschen-Funktion durch Anklicken eines einzelnen Eintrags einer To-Do Liste hinzu. Bevor endgültig gelöscht wird, soll gefragt werden, ob man sich sicher ist.
Tipp: Nutze einen Notifier.

b) Man soll auch einen Eintrag bearbeiten können, falls man sich verschrieben hat. Ergänze diese Funktion mit einem weiteren Notifier, der wählen lässt zwischen Bearbeiten und Löschen.

c) Ergänze die Funktion, dass in der Übersicht auf dem Screen *Verwaltung* hinter jedem Listentitel in Klammern die Anzahl der enthaltenen Elemente notiert wird.

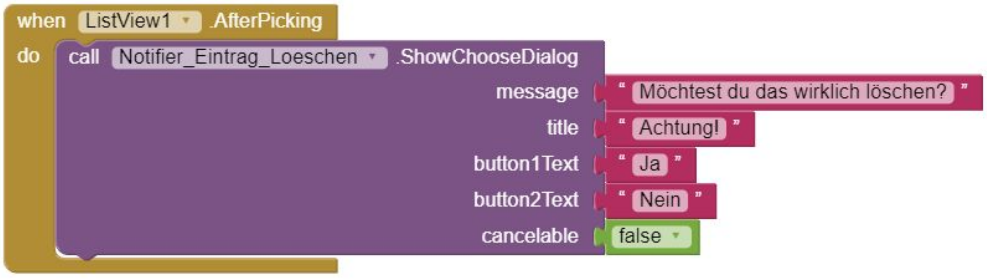
d) Überlege dir weitere eigene Features für deine To-Do Listen App, auf die du Lust hast!

Aufgabe a)

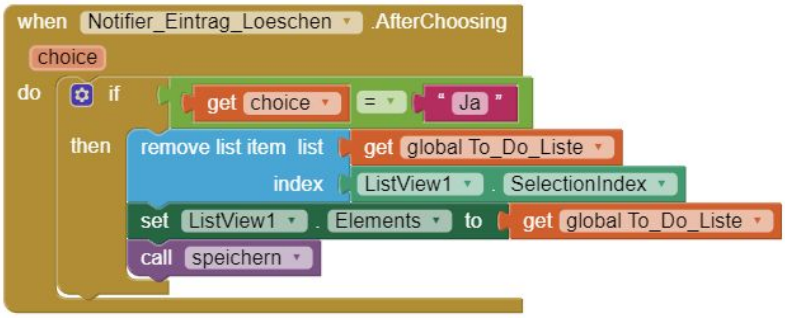
Gehe zum Screen *Liste* und klicke auf *Blocks*.

Zeige zuerst den gleichen Dialog an, wie wenn die ganze Liste gelöscht wird. Nur diesmal im *Notifier_Eintrag_Loeschen*. Es gibt also wieder eine Warnung und zwei Antwortmöglichkeiten.

Tipp: Du kannst einfach den Block vom *Notifier_Liste_Loeschen* duplizieren und den Notifier anpassen.



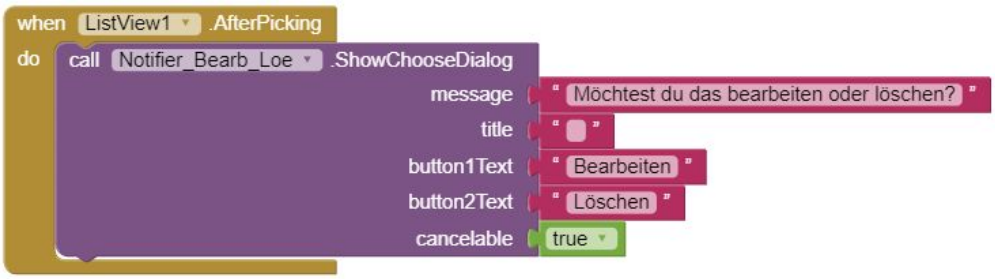
Bei der Antwort “Ja” lösche das ausgewählte Element aus der Variablen *To_Do_Liste* und speichere die Änderung hinterher.
Um das richtige Element zu löschen, ermittle den Index des ausgewählten Elements in der *ListView1*. Dieser Index entspricht genau dem Index des Elements in der gespeicherten Liste.



Aufgabe b)

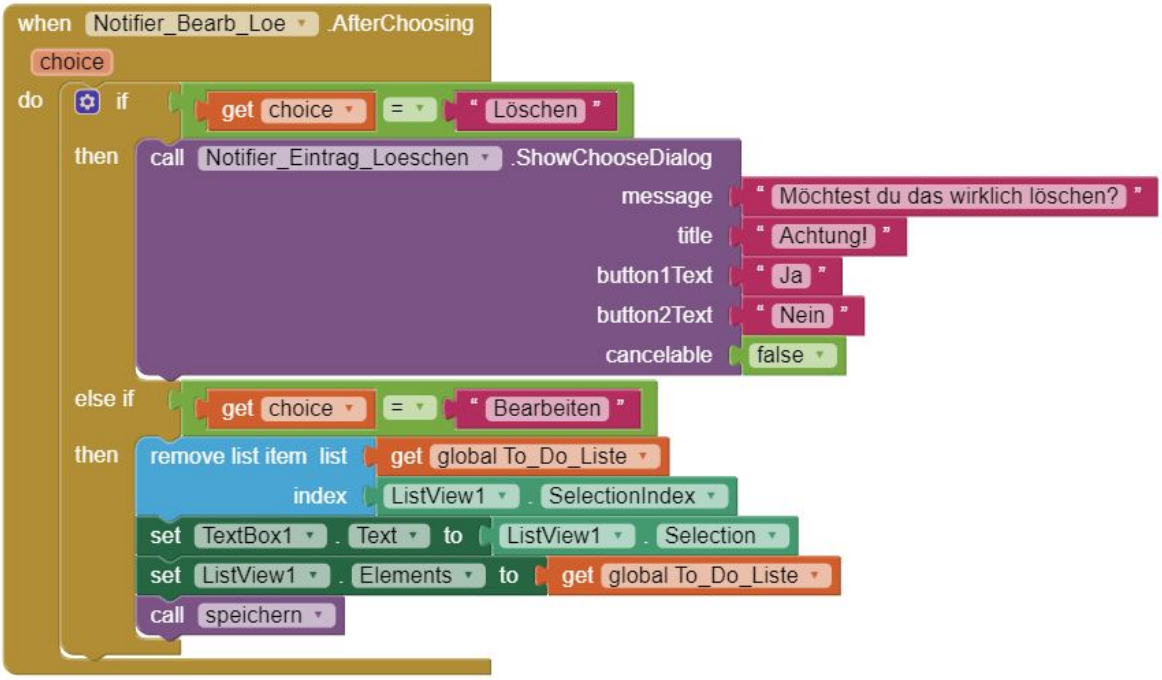
Statt mit dem *Notifier_Eintrag_Loeschen* sofort zu fragen, ob der Eintrag wirklich gelöscht werden soll, rufe zuerst den *Notifier_Bearb_Loe* auf und stelle die Optionen “Bearbeiten” und “Löschen” zur Wahl. Dieser Dialog sollte auch abzubrechen sein (*cancelable* = *true*), falls nichts von beidem getan werden soll.

Wichtig! Es darf nur einen Block mit *ListView1.AfterPicking* geben. Den Block, der vorher darin war, brauchst du gleich noch.



Fortsetzung Aufgabe b)

Wenn also die Option “Löschen” gewählt wurde, verwende wie vorher den *Notifier_Eintrag_Loeschen*. Hat der Benutzer die Option “Bearbeiten” ausgewählt, entferne diesen Eintrag zunächst aus der *To_Do_Liste*. Schreibe ihn dann in die *TextBox1*. So kann der Benutzer den Eintrag einfach verändern und neu hinzufügen. Das erneute Hinzufügen ist schon programmiert über den *Button_Hinzufuegen*.
Damit der unbearbeitete Eintrag aus der *ListView1* verschwindet, musst du diese noch aktualisieren und die *To_Do_Liste* in der Datenbank speichern.

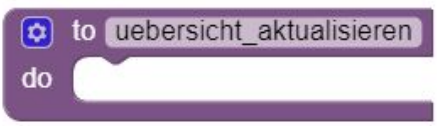


Aufgabe c)

Gehe zum Screen *Verwaltung* und klicke auf *Blocks*.

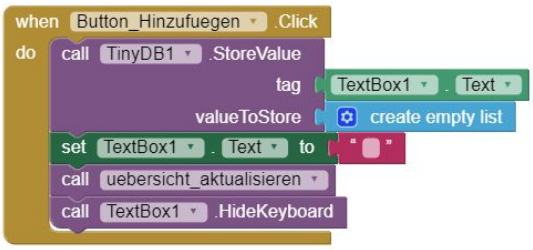
Um die Funktion einzubauen, dass die Anzahl der Elemente in Klammern hinter dem Titel angezeigt wird, muss einiges verändert werden in der Programmierung dieses Screens.

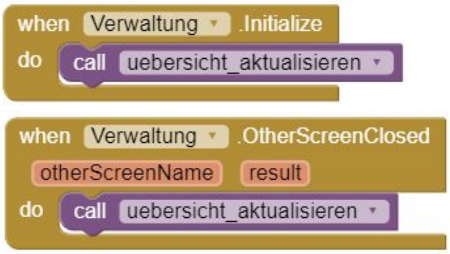
Da nicht mehr nur der Titel selbst angezeigt werden soll, reicht es nicht, die Listentitel mit *TinyDB1.GetTags* abzufragen und anzuzeigen. Deshalb erstelle eine (zunächst leere) neue Funktion *uebersicht_aktualisieren*. Erstelle außerdem eine neue Variable mit einer leeren Liste, in der die Listentitel später zwischengespeichert werden können.



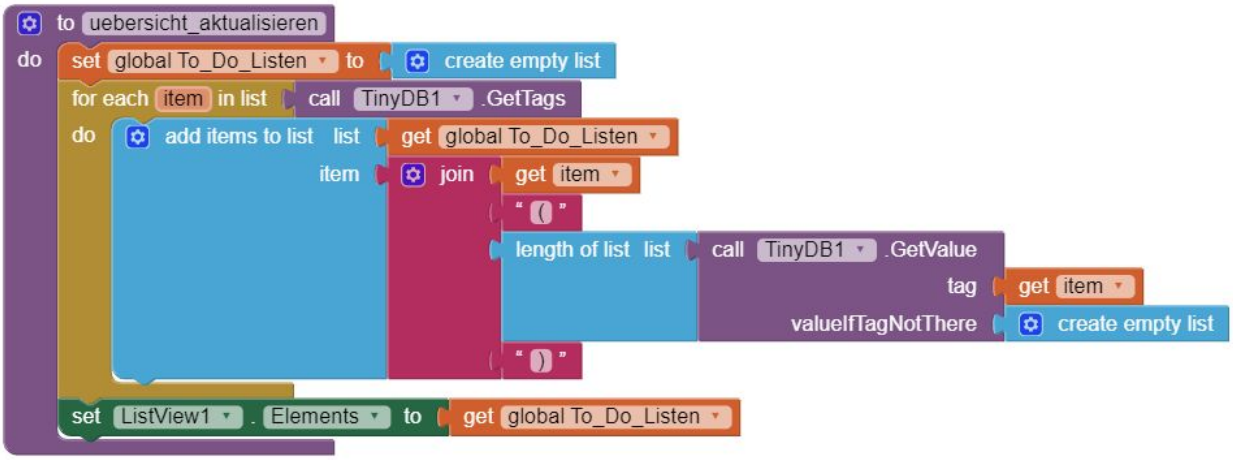
Fortsetzung Aufgabe c)

Ersetze also die Blöcke *set ListView1.Elements to call TinyDB1.GetTags* an den folgenden Stellen durch den Aufruf deiner eigenen Funktion mit *call uebersicht_aktualisieren.*





Die neue Liste mit Listentiteln und Anzahlen muss Schritt für Schritt zusammengebaut werden. In der Datenbank sind nur jeweils Titel und Liste gespeichert. Angezeigt werden soll aber Titel und Anzahl.
Dafür musst du die Anzahl erst ermitteln und dann an den Listentitel anhängen. Mit *for each item in list* gehst du alle Listentitel nacheinander durch und tust genau das.
Du speicherst das, indem du jeden Listentitel mit Anzahl an die Variable *To_Do_Listen* anhängst. Diese Liste zeigst du dann am Ende in der *ListView1* an.
Am Anfang muss die Variable *To_Do_Listen* immer leer sein, da sie jedes Mal neu gefüllt werden soll, wenn die Funktion aufgerufen wird.



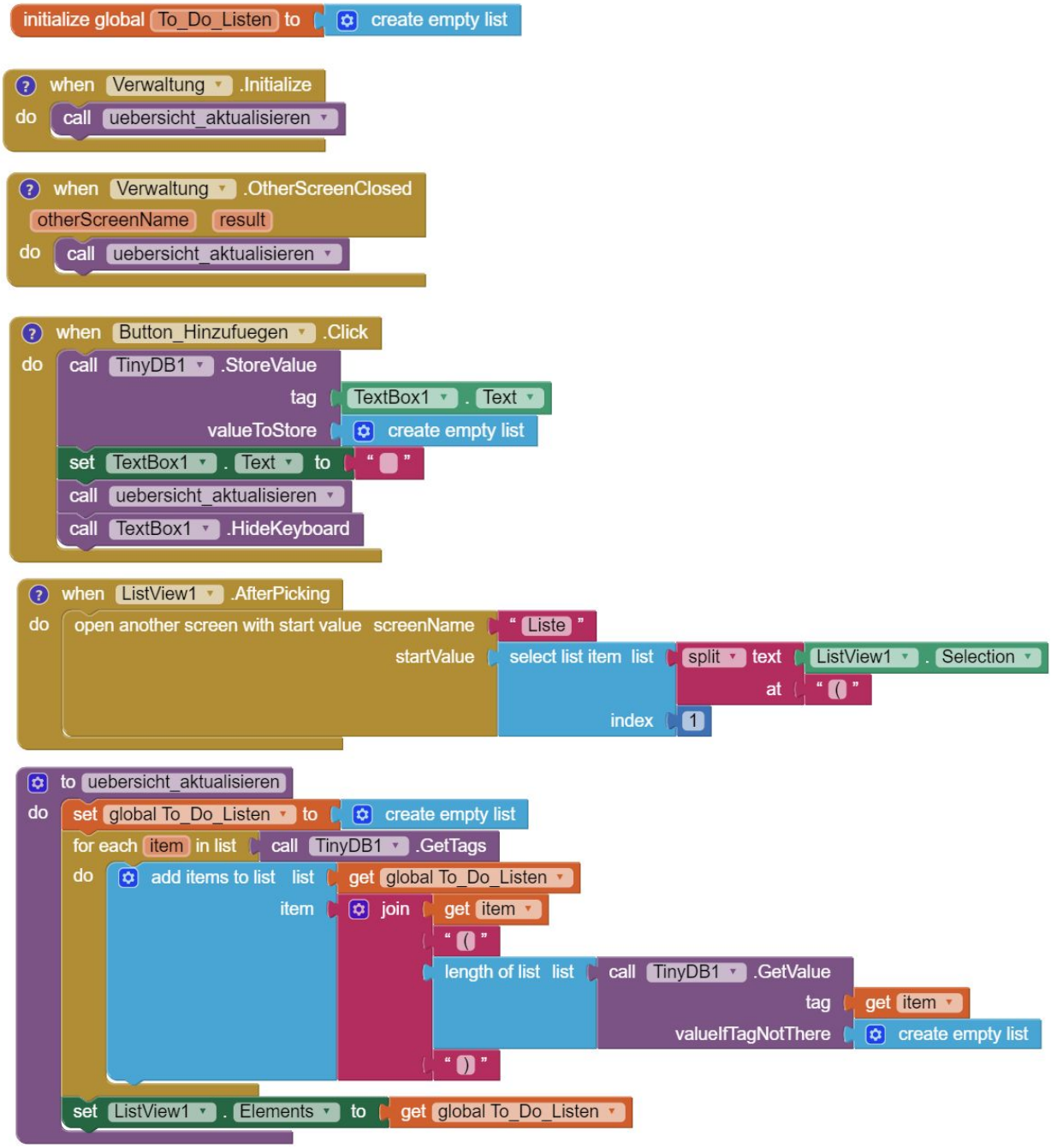
Da in der *ListView1* nun nicht mehr nur Listentitel angezeigt werden, sondern eine Kombination aus Listentitel und Anzahl, muss bei der Auswahl einer Liste auch das Vorgehen angepasst werden.
Um zu ermitteln, welcher Listentitel ausgewählt wurde, wird die Kombination aus Titel und Anzahl bei der geöffneten Klammer unterteilt (engl.: “split” = teilen). Der vordere Teil (index = 1) entspricht dann dem Titel, den du mit *select list item* auswählst.

Nun noch viel Spaß mit deinen eigenen Ideen!

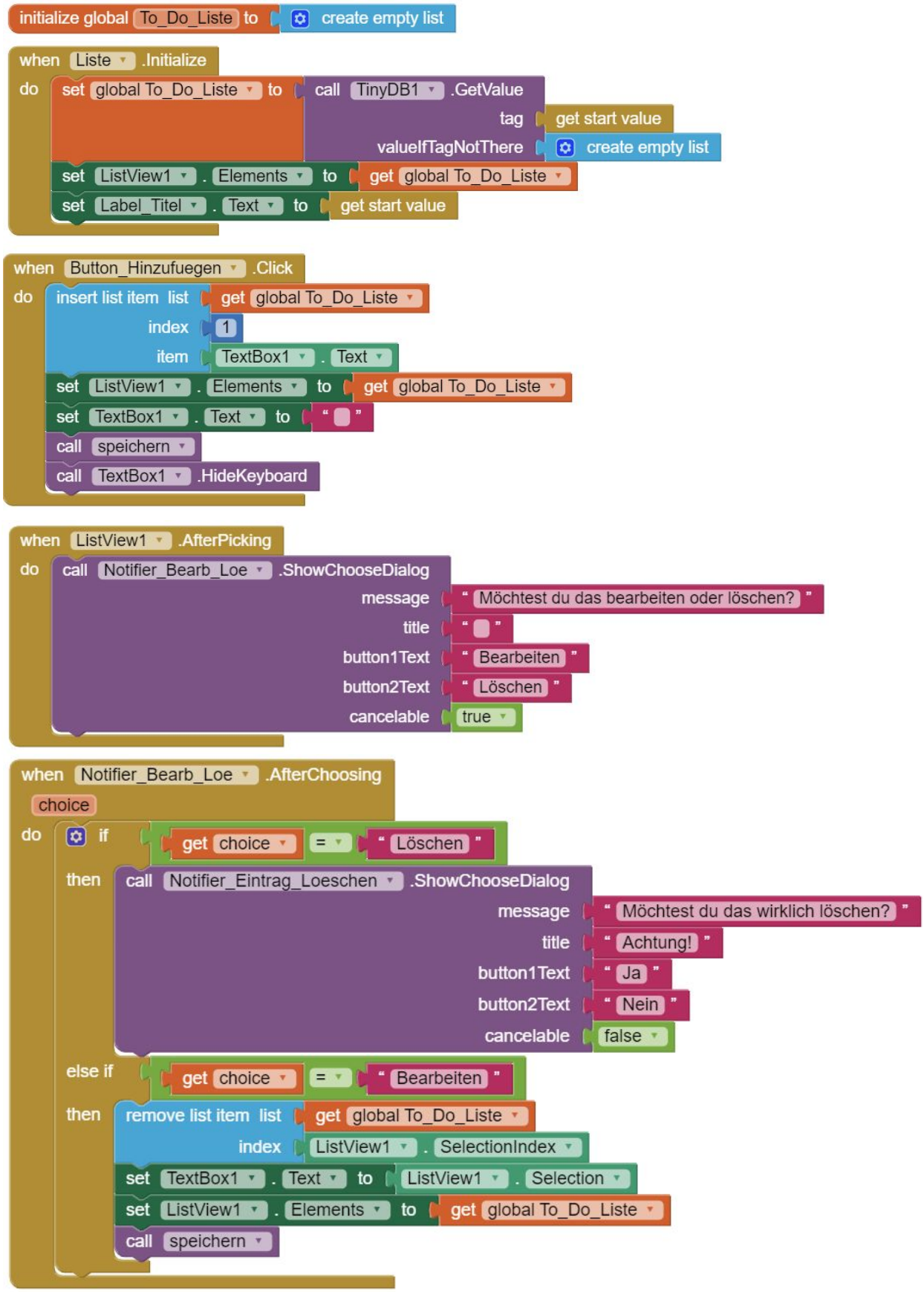
Auf den folgenden Seiten siehst du nochmal in einer Übersicht alle Blöcke, die für die drei Screens in der endgültigen Version nötig sind.

Übersicht Blöcke *Screen1*

Übersicht Blöcke Screen *Verwaltung*



Übersicht Blöcke Screen *Liste*



Übersicht Blöcke Screen *Liste* - Fortsetzung