



Sieb des Eratosthenes

Die App – Anleitung

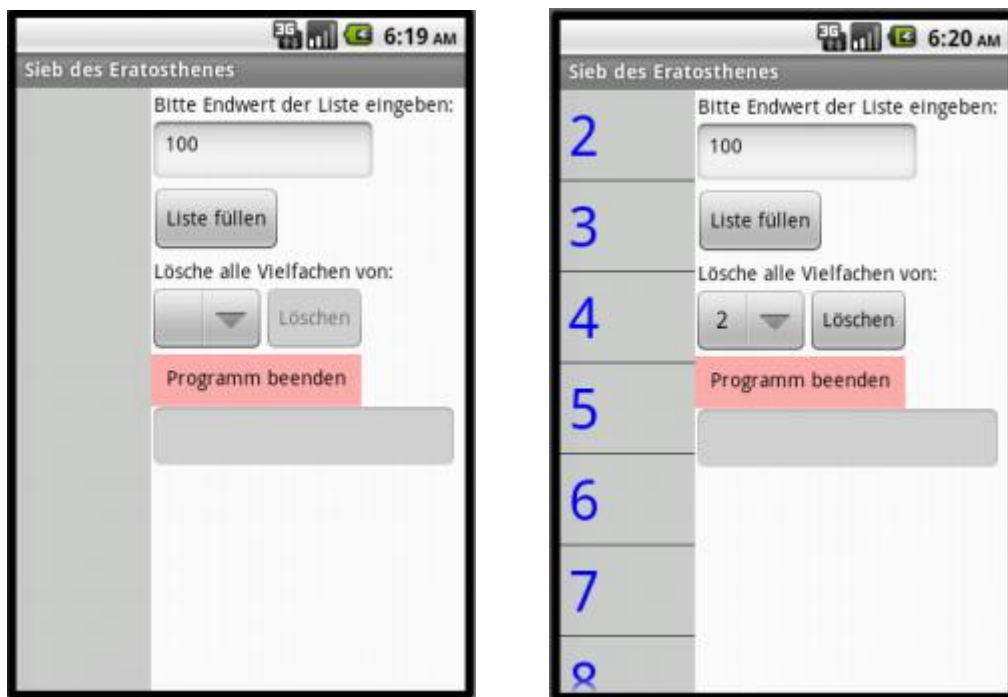
Zunächst werden in der App natürliche Zahlen von 2 bis zu der eingegebenen Zahl in eine Liste geschrieben. Diese wird der ListView auf der linken Seite und der Auswahlliste rechts übergeben.

Um die Vielfachen einer Zahl zu löschen, hat man zwei Möglichkeiten. Entweder man klickt einfach links auf eine der noch übrigen Zahlen (alle Vielfachen werden durch Sterne ersetzt), oder man wählt rechts aus der Auswahlliste eine Zahl aus und drückt auf „Löschen“.

Alle noch übrigen Zahlen werden sowohl in beiden Listen, als auch im Überblick rechts unten im Textfeld dargestellt. Dieses Textfeld ist scrollbar, damit auch bei größeren Zahlenmengen alles sichtbar gemacht werden kann.

Der Screen

In der ersten Abbildung sieht man den Startbildschirm der App. Nach Eingabe einer Zahl und dem Klick auf „Liste Füllen“ sieht man den Screen der zweiten Abbildung.



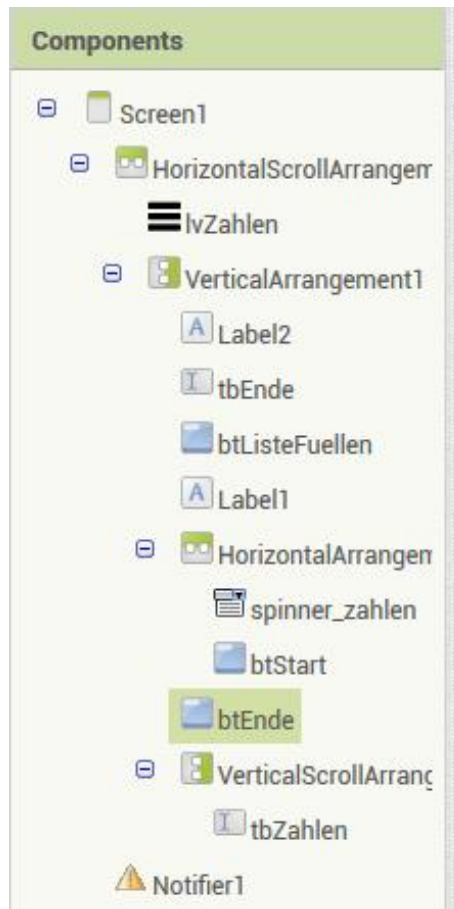
Als erstes wurde hier die Zahl 2 ausgewählt (dritter Screen), danach alle weiteren übrigen Zahlen bis zur Zahl 7 (inklusive). Da hier die Zahlen 2 bis 100 in der Liste stehen, genügt es, die Vielfachen der (Prim-)Zahlen bis 7 löschen zu lassen, da das Quadrat der nächsten übrigen Zahl (11) schon über 100 liegt.

Es bleiben am Ende nur noch die Primzahlen übrig.



Der Screen im MIT-App-Inventor

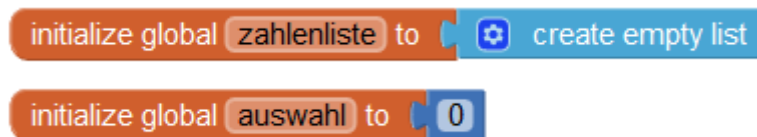




Komponenten, die lediglich Information geben aber im Code nicht verwendet werden, wurden nicht umbenannt.

Der Code

Initialisierung der Variablen:



In der Variable *auswahl* wird die Zahl (oder ein Stern) abgespeichert, die ausgewählt wurde.

In der Liste *zahlenliste* werden später alle Zahlen von 2 bis zur eingegebenen Zahl als Elemente gespeichert.

Das OnClick-Ereignis des Buttons mit der Aufschrift „Liste Füllen“:



```

when btListeFuellen .Click
do
  set tbZahlen . Text to " "
  set global zahlenliste to create empty list
  for each number from 2
    to tbEnde . Text
    by 1
  do
    add items to list list get global zahlenliste
    item get number
  set lvZahlen . Elements to get global zahlenliste
  initialize local liste to create empty list
  in for each number from 2
    to
    if is number? tbEnde . Text
    then round square root tbEnde . Text
    else 100
    by 1
  do
    add items to list list get liste
    item get number
  set spinner_zahlen . Elements to get liste
  set btStart . Enabled to true
  set global auswahl to 2
  
```

Da der Button auch dazu da ist, die Liste erneut zu füllen, müssen zunächst das Textfeld rechts geleert und die Liste neu initialisiert werden.

Mithilfe einer for-Schleife werden die Zahlen von 2 bis zur Endzahl in der Liste gespeichert. Die Elemente werden im Anschluss an die ListView übergeben.

Die Elemente der Auswahlliste rechts werden zunächst extra erzeugt bis zur Wurzel der Endzahl. Allerdings wird diese Liste in jedem weiteren Schritt später mit der Gesamtliste überschrieben, deshalb ist die gesonderte Liste nur zu Beginn sichtbar.

Der Button zum Löschen der Vielfachen wird aktiviert und die erste Auswahl auf die Zahl 2 gesetzt.

Möglichkeit 1 zum Löschen der Vielfachen:

Möglichkeit 2:



```

when spinner_zahlen .AfterSelecting
do
  call sterntest
  auswahl ← get selection
  
```

```

when lvZahlen .AfterPicking
do
  call sterntest
  auswahl ← lvZahlen . Selection
  call streichen
  call zahlen_anzeigen
  
```

```

when btStart .Click
do
  call streichen
  call zahlen_anzeigen
  
```

In beiden Varianten werden drei Unterprogramme aufgerufen.

Im Unterprogramm *sterntest* wird überprüft, ob die Auswahl eine Zahl ist. Falls ja, wird diese in der Variable *auswahl* gespeichert. Falls nicht, wird eine Nachricht ausgegeben.

```

to sterntest auswahl
do
  if (get auswahl ≠ "*" )
  then
    set global auswahl to get auswahl
    set btStart . Enabled to true
  else
    call Notifier1 . ShowMessageDialog
    message " Bitte Zahl auswählen. "
    title " ACHTUNG: "
    buttonText " OK "
  
```

Im Unterprogramm *streichen* wird die Zahlenliste durchlaufen und alle Zahlen, die Vielfache von *auswahl* sind, durch einen Stern ersetzt.

Die veränderte Liste wird an die beiden entsprechenden Komponenten übergeben und der Button zum Löschen wird deaktiviert bis zur nächsten Auswahl.

```

to streichen
do
  for each number from 2
  to length of list list ← get global zahlenliste + 1
  by 1
  do
    if (get number ≠ get global auswahl and modulo of get number ÷ get global auswahl = 0)
    then
      replace list item list ← get global zahlenliste
      index ← get number - 1
      replacement "*"
  
```

```

set lvZahlen . Elements to get global zahlenliste
set spinner_zahlen . Elements to get global zahlenliste
set btStart . Enabled to false
  
```



Im Unterprogramm *zahlen_anzeigen* wird das Textfeld rechts unten zunächst geleert, dann werden dort die Elemente der Zahlenliste eingetragen.

```

to zahlen_anzeigen
do
  set tbZahlen . Text to ""
  for each number from 1
    to length of list list get global zahlenliste
    by 1
  do
    set tbZahlen . Text to join
      tbZahlen . Text
      ""
      select list item list
        get global zahlenliste
        index
        get number
  
```

Beenden der App:

```

when btEnde .Click
do
  close application
  
```

Bemerkungen

Es gibt keine Fehlerabfrage, sollte eine Dezimalzahl eingegeben werden.

Den Code zum Bearbeiten findet man unter *sieb_des_eratosthenes_v4.aia*, die App zum Installieren unter *sieb_des_eratosthenes.apk*.

Der MIT App Inventor (<http://appinventor.mit.edu>) wurde ursprünglich von einem Entwicklerteam um Mark Friedman und Hal Abelson bei Google entwickelt und 2012 an das MIT übergeben.
 Der MIT App Inventor wird unter der Creative Commons Attribution-ShareAlike 3.0 Unported License veröffentlicht: <https://creativecommons.org/licenses/by-sa/3.0>

Scratch wird von der Lifelong-Kindergarten-Group am MIT-Media-Lab entwickelt. Siehe <http://scratch.mit.edu>.
 Scratch ist lizenziert unter CC BY-SA 2.0 (<https://creativecommons.org/licenses/by-sa/2.0/deed.en>).