Entwicklung einer KI-App zur Bildgenerierung mit dem MIT App Inventor

Lennard Dorst – Peer Stechert

Neben ChatGPT als prominentem Vertreter von Künstlicher Intelligenz (KI) gibt es auch zahlreiche KI-Modelle zur Generierung von Bildern. In dieser Aufgabe wird mit dem MIT App Inventor eine kleine App erstellt, die Bilder generiert: In der App wird eine Beschreibung (Prompt) angegeben. Daraufhin wird ein Workflow bei einer KI-API gestartet, die extern ein Bild generiert, das anschließend in der App angezeigt wird (Abb. 1).

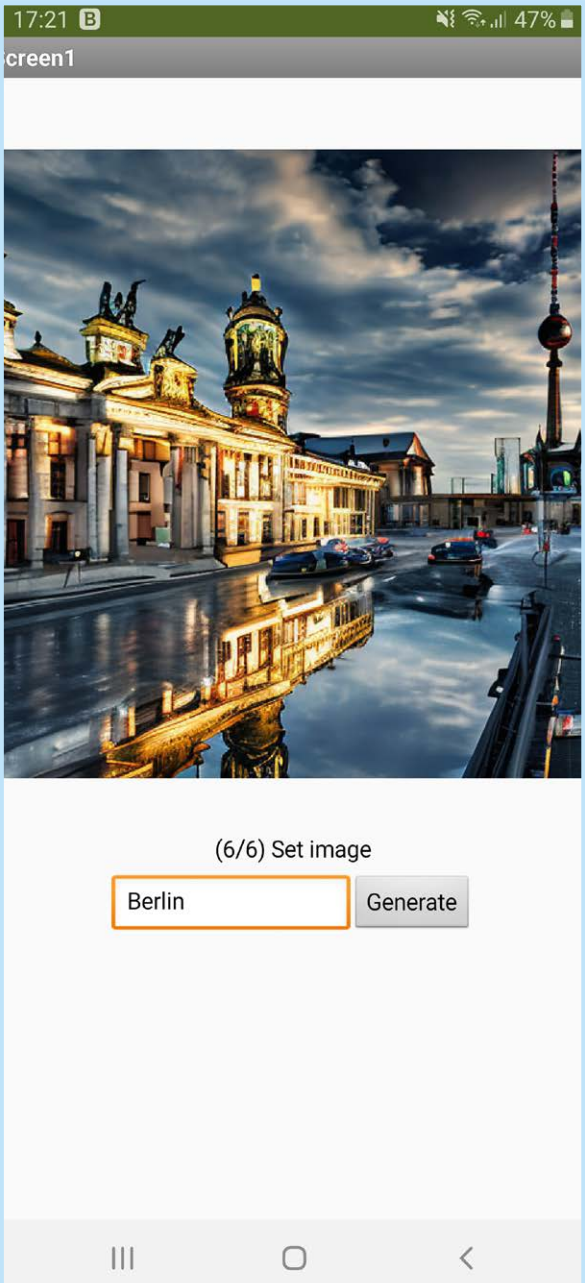

Abb. 1. Die fertige App (mittels AI Companion App) liefert zu einem Prompt ein futuristisches Bild

Hinweis: Der API-Aufruf (Application Programming Interface) erfolgt über LeapML (https://www.tryleap.ai/). Für LeapML ist eine Anmeldung erforderlich, so dass die Lehrperson aus Datenschutzgründen ggf. Konten für die Schüler/innen anlegen muss.

1. Erstelle ein Konto bei www.tryleap.ai. Erstelle einen einfachen Workflow, importiere unter „Settings“ die in der online-Ergänzung verfügbare json-Datei und gehe auf „publish“ (Basis ist das allgemeine Workflow-Template „Fast Image Generation“). Speichere die Workflow-ID und generiere den API-Key.
2. Melde dich beim MIT App Inventor an und erstelle ein neues Projekt.
3. Füge eine TextBox hinzu, in der der Benutzer den Prompt als Input eingeben kann.
4. Füge einen Button hinzu, der gedrückt werden kann, um die Bild-Generierung zu starten.
5. Informiere dich im Netz über die Funktionsweise der Web-Komponente. Füge sie zweimal, mit den Namen *„postRequest“* und *„getRequest“*, als „non-visible components“ der App hinzu. Informiere dich außerdem über die Funktionsweise der LeapAPI: https://docs.tryleap.ai/
6. Erstelle eine Funktion (Abb. 2), die aufgerufen wird, wenn der Button angewählt wird (Abb. 3). Der Prompt-Parameter wird durch das Input-Feld erfasst.

   Header und JSON-Body sind zwei verschiedene Felder, die an die API gesendet werden können. Im Header befinden sich Informationen über die erwartete Rückgabe (z.B. Content-Type) und Autorisierung. Im JSON-Body befinden sich genaue Anweisungen an die API, in diesem Fall auch der Prompt und die Workflow-ID. Später wird außerhalb, nach dem „runWorkflow“-Procedere, der Timer gestartet.
7. Erstelle ein Event (Abb. 4), das aufgerufen wird, wenn die POST-Anfrage eine Antwort und einen Status zurückgegeben hat. Die POST-Anfrage wird uns eine ID des Runs geben, über die später die Outputs des Workflows gespeichert werden.
8. Füge der App die clock als „non-visible component“ hinzu und erstelle eine Funktion, welche, über einen „Timer-Baustein“, jede Sekunde eine Anfrage versendet, um zu schauen, ob der Workflow ein Ergebnis hat. Diese wird in Abb. 5 „pollImage“ genannt.
9. Erstelle eine Funktion, um das Bild herunterzuladen, und es in der App anzuzeigen (Abb. 6)
10. Teste die App, indem du als Prompt unterschiedliche Beschreibungen eingibst und auf den Button klickst, um das generierte Bild anzuzeigen.

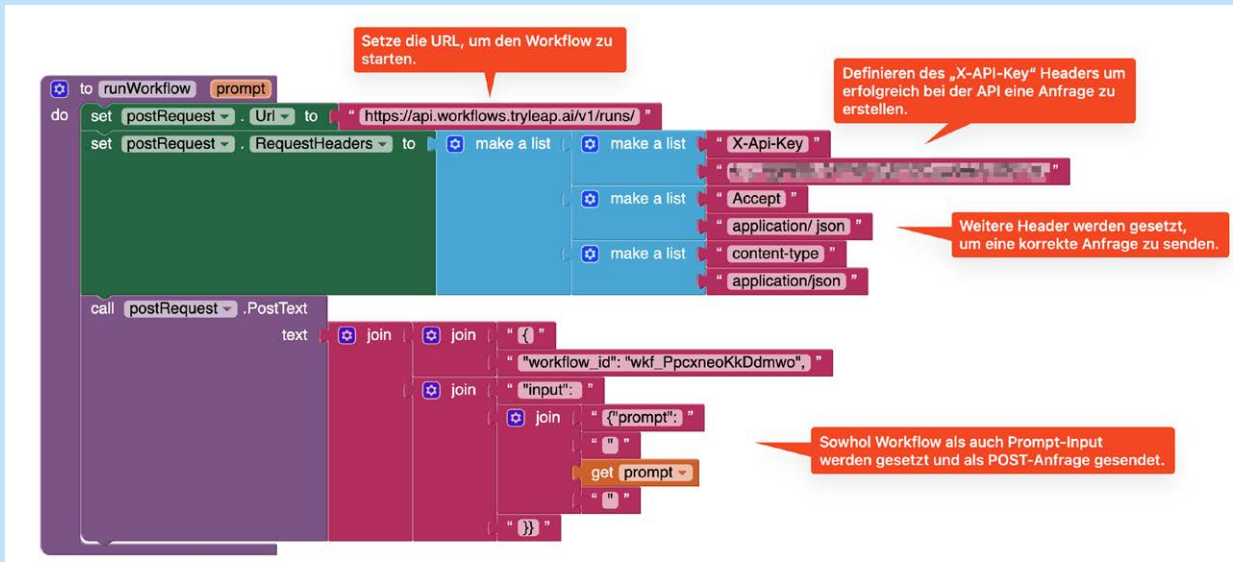

Abb. 2. Funktion runWorkflow mit *postRequest*

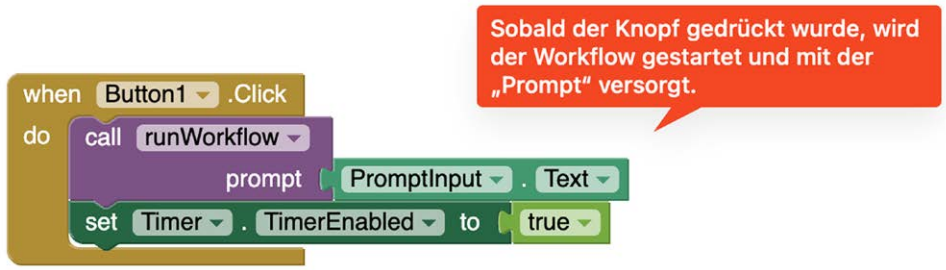

Abb. 3. Event zum Aufruf der runWorkflow-Funktion

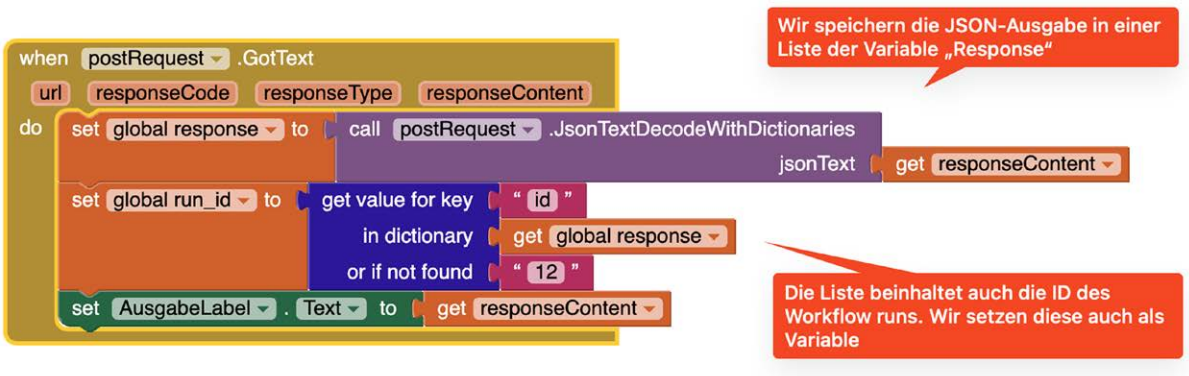

Abb. 4. Event, wenn die POST-Anfrage eine Antwort und einen Status zurückgegeben hat.

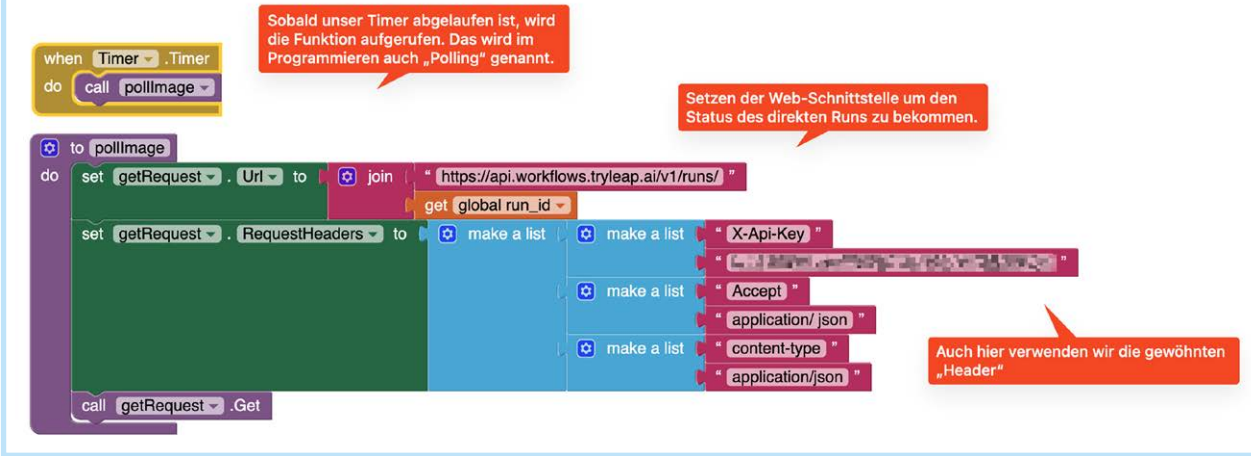

Abb. 5. Funktion pollImage mit Timer-Baustein

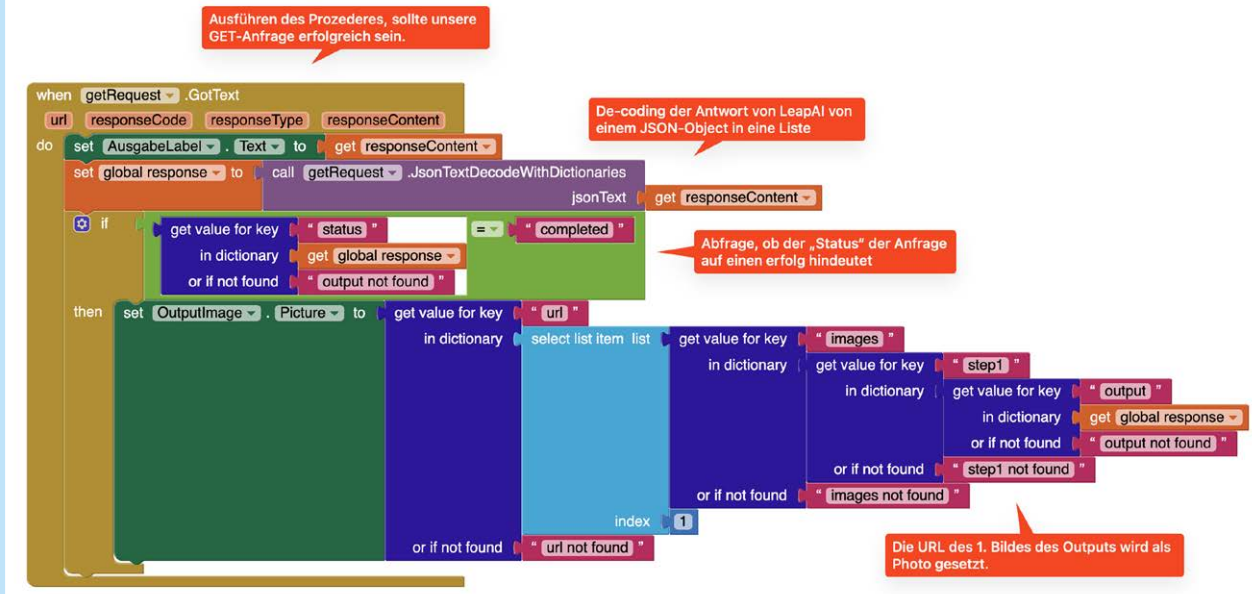

Abb. 6. Event, um eines der generierten Bilder anzuzeigen.

Wie bei allen Aufgaben sind Lösungs- und Bearbeitungshinweise zu dieser Aufgabe für MNU-Mitglieder im internen Bereich der MNU-Homepage zum Download bereitgestellt.

mit online ergänzung www.mnu.de