TECHNISCHE UNIVERSITÄT DRESDEN

Kunst **I**nteraktiv

– Die intelligente Museumsapp –

Bild KI-generiert mit Disco-Diffusion (Disco Diffusion License)

EDUiNF EDUCATION IN INFORMATICS

DDI Professur für Didaktik der Informatik

BILDERKENNUNG

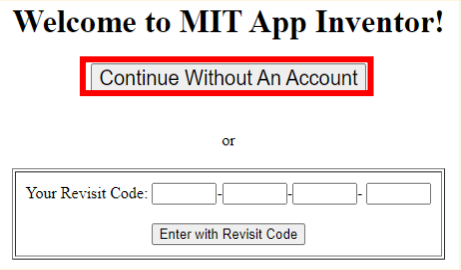
Sicher wart ihr schon einmal in einem Museum und habt euch einen **Audioguide** ausgeliehen. Damit erhält man beim Betreten eines entsprechenden Raumes oder beim Drücken einer Taste Informationen. Wir wollen eine App mit ähnlichem Prinzip erstellen. Diese funktioniert jedoch nicht auf Tastendruck, sondern kann auf Bildern das gesuchte Exponat selbstständig erkennen.

DAS LERNST DU

In diesem Lernmodul erfährst du, wie dein eigenes **Machine Learning Modell** in eine **App** eingebettet werden kann und wie du sie mit dem Smartphone benutzt.

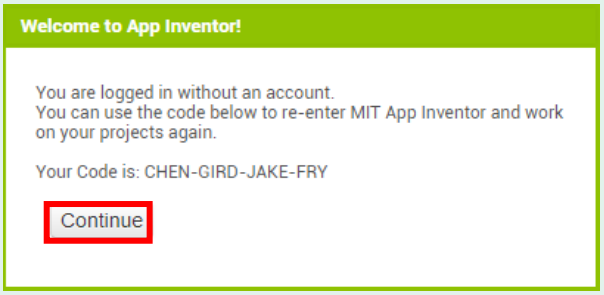
AUFGABE

Zum Einrichten der Programmierumgebung **öffne** zuerst **http://code.appinventor.mit.edu/login/** in deinem Browser und **melde** dich **ohne Google-Account** an (siehe Bild).



HINWEIS

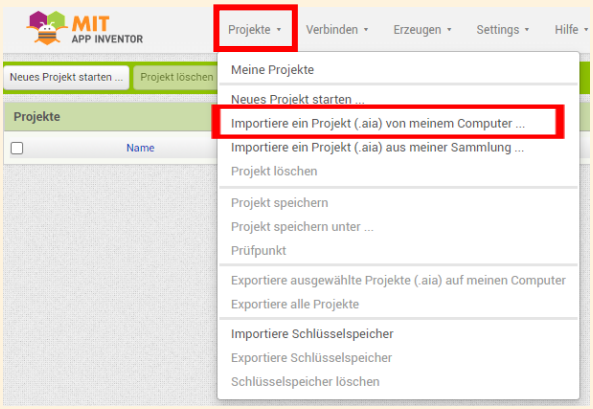
Wenn du auch zuhause auf dein Projekt zugreifen oder damit weiterarbeiten möchtest, **notiere** dir **den Code**, der dir nach dem Einloggen in einem grünen Feld angezeigt wird. Klicke anschließend auf *Continue*.



AUFGABE

a) **Ändere** bei Bedarf die **Sprache** oben rechts auf **Deutsch**.

b) **Importiere** dir das bereits vorgefertigte App-Projekt über **„Projekte“** von deinem Computer, wie im Bild gezeigt. Deine Lehrkraft stellt dir die Datei zur Verfügung.



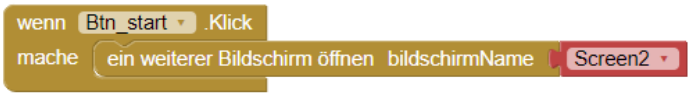
AUFBAU VON APPINVENTOR

Wenn du das Projekt erfolgreich importieren konntest, geht es weiter. Nun wollen wir uns den Aufbau und die Funktion der **Programmierumgebung „AppInventor“** anschauen. Auf dem Startbildschirm sehen wir ein kleines Logo und eine Beschreibung, was die App tun soll.

AUFGABE

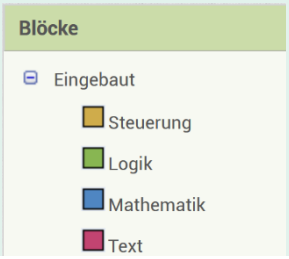
Wechsle mit den Buttons **oben rechts** Designer Blöcke in den *Blöcke*-Modus, um den bereits geschrieben Code im Blockformat anzusehen.

Dort sehen wir, dass durch den Klick auf den Button *Start* ein anderer Bildschirm **Screen2** geöffnet wird. Weitere Hinweise erhältst du in der grünen Box auf der nächsten Seite.



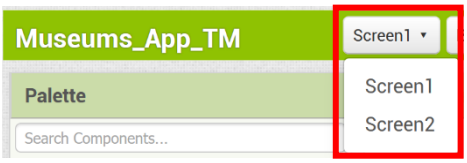
HINWEIS

Mit dem *Designer* wir die **App gestaltet**. Mit Hilfe des Auswahlmenüs links können verschiedene Texte, Bilder usw. durch „Drag and Drop“ gestaltet werden.

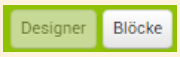


Im Modus *Blöcke* kann die **Oberfläche** mithilfe der **Programmierbausteine links** durch „Drag and Drop“ beliebig **programmiert** werden.

Zwischen **Screens** kann man wechseln, indem man oben links **Screen1** anklickt.



AUFGABE

Wechsle nun wieder in den *Design*-Modus mit den Buttons **oben rechts** 

.

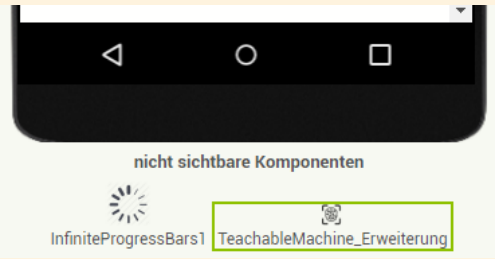
Schauen wir uns **Screen 2** an, sehen wir das Herzstück der App. Die Funktion der App ist es, mittels des Buttons **„Klassifizieren“** ein Bild aufzunehmen. Anschließend wird das Ergebnis ausgegeben, welches mit der **höchsten Wahrscheinlichkeit** erkannt wurde. Dann machen wir uns mal an die Umsetzung!

TEACHABLE MACHINE EINBINDEN

Fangen wir damit an, unser TeachableMachine-Modell (TM) einzubinden, welches wir bereits erstellt und trainiert haben.

AUFGABE 4

a) Wähle im *Designer* von **Screen2** die *TeachableMachine_Erweiterung* unter **nicht sichtbare Komponenten** aus.



b) Jetzt müssen wir unser TM-Modell mit der Erweiterung verbinden. Trage bei **Eigenschaften** (rechts) unter **URLModel** den Link zum Teachable Machine-Modell ein.

HINWEIS

Unterhalb des **Betrachters** im *Designer* sind die **nicht sichtbaren Komponenten** des jeweiligen Screens zu sehen. Beim **Anklicken** öffnet sich **rechts** ein **Fenster**, in dem **Eigenschaften** geändert werden können.

DIE TEACHABLE-MACHINE ERWEITERUNG

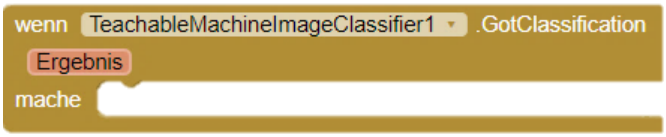
Die **TM-Erweiterung** dient der **Einbindung des ML-Modells** und der Erkennung der trainierten Objekte. **Rückgabewerte** sind **Wahrscheinlichkeiten** für einzelne Klassen.

AUFGABE 7

Wechsle auf **Screen 2** in den *Blöcke*-Modus und schau dir den Code an.
Was passiert, wenn dieser Code ausgeführt wird?

Mithilfe des 

- Befehls wird dem Modell ein Bild übergeben. Anschließend wird dieses durch das Modell ausgewertet.

Unter dem Befehl 

kann das Ergebnis dieser Klassifikation verarbeitet werden. Das **Ergebnis** ist dabei als Variable nutzbar. Sie hat die Form eines **Dictionaries**.

DICTIONARIES

Dictionaries sind eine besondere Form der Datenstruktur. Hier werden Daten nicht wie bei einem Array an einem bestimmten Platz gespeichert, sondern als Kombination aus einem **Schlüssel** und dem zugehörigen **Wert**.

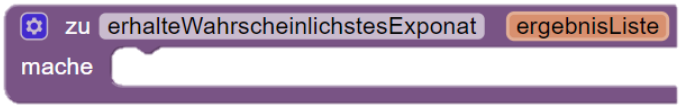
Haben wir ein Modell zum Erkennen von Schere, Stein, Papier erstellt, könnte ein Dictionary mit einem erkannten Ergebnis so aussehen →

Schlüssel		Wert
Schere	→	0,08
Stein	→	0,03
Papier	→	0,89

DAS WAHRSCHEINLICHSTE EXPONAT ERMITTELN

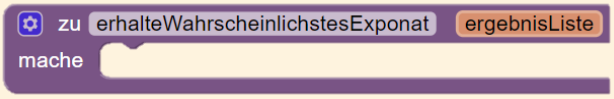
Das ML-Modell kann nicht zaubern! Es ermittelt anhand der eingegebenen Bilddaten, wie **wahrscheinlich** es ist, dass eine **Kategorie** (Klasse) getroffen wurde. Der Mensch muss diese Zahl dann interpretieren und ggf. das Modell weiter trainieren, wenn Unstimmigkeiten auftauchen.

Wir wollen in unserer App nur das Ergebnis anzeigen, das mit der höchsten **Wahrscheinlichkeit** erkannt wurde. Wir erstellen uns dazu eine **Funktion**



, die das **Ergebnis**-Dictionary erhält und den Eintrag mit dem höchsten Wert zurückgibt.

AUFGABE 5

Implementiere die Funktion 

.

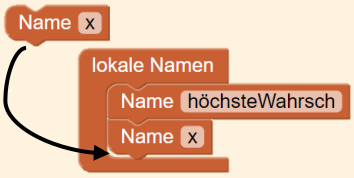
Nutze die folgenden *Hinweise*, um die Funktion zu befüllen.

Tipps:

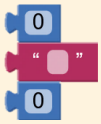
1. Mit 

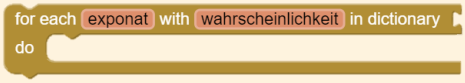
können in der Funktion lokale Variablen initialisiert werden.

a) Klicke auf das Zahnrad und ziehe die **Blöcke** mit der korrekten Bezeichnung in den Block „**lokale Namen**“.



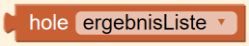
b) Initialisiere so drei Variablen (*höchsteWahrsch*, *hilfExpo* und *tempWahrsch*) mit Hilfe der Werte 0 bzw. mit leerem Text bei *hilfExpo*.



2. Die Schleife 

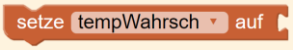
durchläuft einmal das Dictionary, so dass jedes Schlüssel-Wert-Paar erreicht wird, um nach dem größten Wert (der größten Wahrscheinlichkeit) zu suchen.

a) Füge die Schleife mit den beiden Variablen (*exponat* und *wahrscheinlichkeit*) wie abgebildet in den „mache“-Block ein.

b) Mit dem 

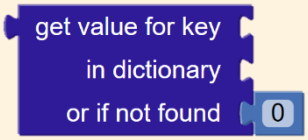
-Block wird angegeben, durch welches Dictionary iteriert wird.

3. In jedem Schleifenschritt wird ein Schlüssel unter Variable **exponat** abgelegt. Um nun den größten Wert zu finden, müssen wir in jedem Schleifendurchlauf den aktuellen Wert mit dem bisher größten vergleichen.

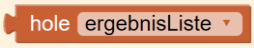
a) Den Wert des aktuellen Schleifenschritts speichern wir und in einer Hilfsvariable 

.

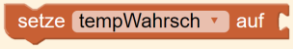
b) Um den aktuellen Wert zu erhalten, müssen wir nur in das **Dictionary** mit dem aktuellen Schlüssel **exponat** schauen.



c) Als Erinnerung: der aktuelle **Schlüssel** ist 

und das Dictionary ist 

.

4. Nachdem wir nun den aktuellen Wert (

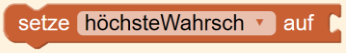
) haben, müssen wir ihn nur mit dem bisher höchsten Wert (

) vergleichen. Wenn der aktuelle Wert höher ist, dann wird er der neue höchste Wert.

a) Als erstes brauchen wir eine **Verzweigung** aus **Steuerung**, um die **wenn-dann** Beziehung aus *Punkt 4* umzusetzen.

b) Um die Werte miteinander zu vergleichen, benutzen wir den 

- Block aus **Mathematik**.

c) Wenn der aktuelle Wert **größer** ist als der bisherige, dann speichern wir uns auf 

den aktuellen Wert und mithilfe von 

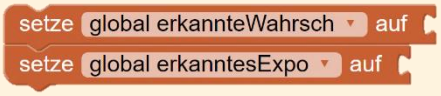
speichern wir uns den aktuellen Schlüssel.

5. Fast geschafft. Wenn unsere Schleife durchgelaufen ist haben wir auf 

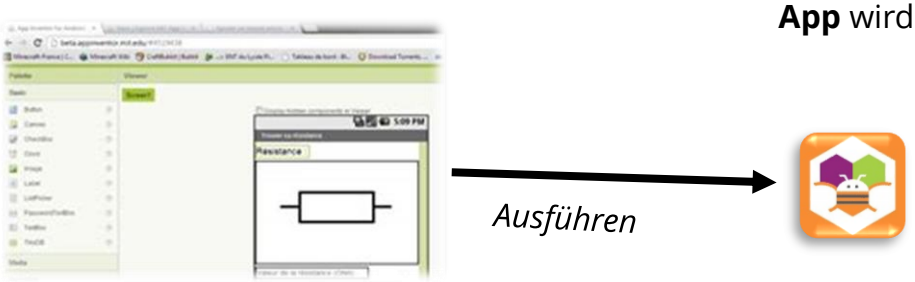
den höchsten Wert des **Dictionary** und auf 

den dazugehörigen Schlüssel.

a) Jetzt müssen wir nur noch die Werte auf die dazugehörigen globalen Variablen speichern, damit wir sie in unseren anderen Methoden benutzen können.



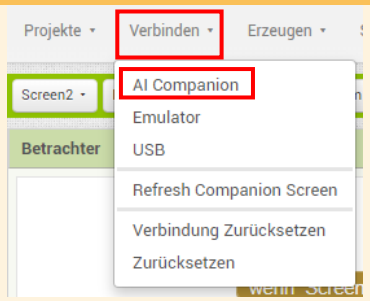
Um die App auf Tablet oder Smartphone zu testen, benutzen wir die **AI Companion**-App. Auf dem Tablet wird der Code „zum Leben erweckt“ und die **App** wird **sicht**- und **nutzbar**.



AUFGABE 5

a) Wähle oben links unter **Verbinden** → **AI Companion** aus.

b) Startet auf deinem Tablet oder Smartphone die *AI Companion App* und **scanne** den **QR-Code**.



Wenn alles funktioniert hat, solltest du nun ein Bild von einem Exponat machen können und anschließend den Namen des Exponates anzeigt bekommen.

HINWEIS

Wenn die App noch nicht zuverlässig die Exponate erkennt, musst du dein TM-Modell noch einmal **verbessern**. Wechsle dazu zurück zu Teachable Machine und verbessere dein Modell mittels weiterer Trainingsdaten.

ACHTUNG

Vergiss nicht, das Modell wieder neu zu **exportieren!**

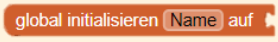
BESCHREIBUNG EINFÜGEN

Die App funktioniert zwar, aber wir bekommen noch keine Beschreibungen für die Exponate angezeigt. Das wollen wir nun ändern. Als erstes brauchen wir dafür die **Beschreibungstexte**.

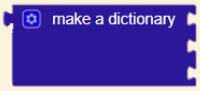
AUFGABE 6

Denk dir eine kurze Beschreibung zu jedem Exponat aus und notiere sie auf einem **Blatt oder neuen Dokument**. Das können zum Beispiel interessante oder lustige Fakten sein.

AUFGABE 8

a) Wähle unter **Variablen** den Block 

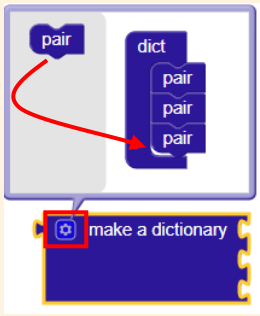
. Klicke auf **Name** und bezeichne dein Dictionary sinnvoll (zum Beispiel *beschreibungenExponate*)

b) Wähle nun unter **Dictionary** den Block 

.

c) Wenn du auf das Zahnrad klickt, kannst du für jedes Exponat einen Eintrag im Dictionary anlegen, indem du jeweils einen 

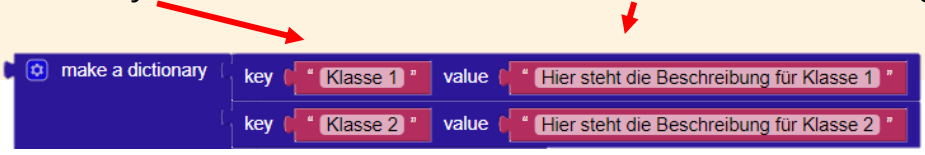
-Block für jedes Exponat in die Liste rechts daneben ziehst.



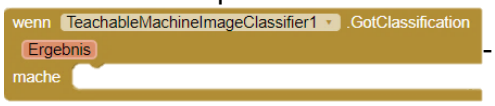
d) Nimm für jedes Exponat aus **Dictionary** den 

- Block und füge ihn am Dictionary hinzu.

e) In die freien Stellen fügst du aus **Text** die 
- Blöcke ein und gibst unter *key* die Klassennamen und unter *value* die Beschreibungen ein.



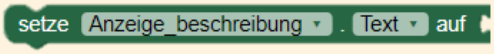
Super, die **Beschreibungen** sind jetzt in der App! Jetzt müssen wir nur noch dafür sorgen, dass sie **angezeigt** werden, wenn das entsprechende Exponat erkannt wurde. Das passiert nach der **Erkennung**. Wir arbeiten ab jetzt in der



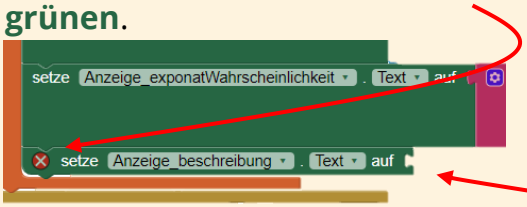
-Methode.

AUFGABE 9

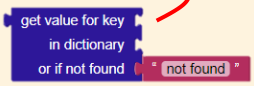
a) Scrolle links im *Blöcke-Abschnitt* runter und wähle unter *Screen 2* das Element **Anzeige_beschreibung** aus. Wähle nun den Block



aus und setze ihn unter den anderen **grünen**.



b) Füge nun aus **Dictionaries** den

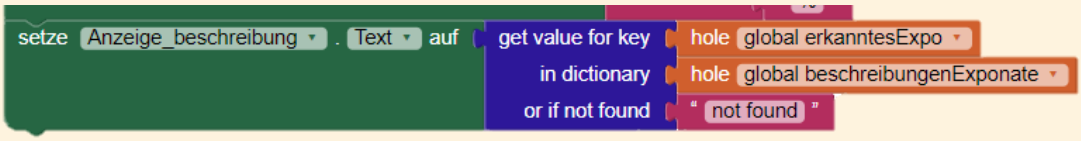


- Block an.

c) Für die obere freie Stelle benötigen wir den Schlüssel, also den Namen des erkannten Exponats. Diesen bekommen wir mit dem Block

unter **Variablen**.

d) Als letzten Schritt müssen wir nur noch das Dictionary angeben, in dem die Beschreibung liegen. Also genau das, welches du in Aufgabe 8 a) benannt hast.



Super, du hast es geschafft! Nun kannst du deine fertige App wieder mit der **AI Companion App** verbinden und ausprobieren.