

# MIT App Inventor: ChatGPT App

麻省理工学院应用发明者：ChatGPT 应用

---

< ESC Club/App In Club >





# Table of contents

# 目录

## 01

### App

We will make our own project  
so you will create your own  
app with MIT App Inventor!

我们将制作自己的项目，以便您  
可以使用 MIT App Inventor  
构建您自己的应用程序！

¡Haremos nuestro propio  
proyecto para que usted cree  
su propia aplicación con MIT  
App Inventor!

# Today you will need..

今天你将需要.....

- A computer
- A android or apple mobile device

● 一台 ☒ ☒

● 安卓或苹果移 ☒ ☒  
☒



or



# What are we doing in Mit app Inventor?

## 我们在 Mit 应用程序 Inventor 中做什么？

We are going to make an  
**AI Chatbot APP!**

我们要做一个AI聊天机器人  
APP！



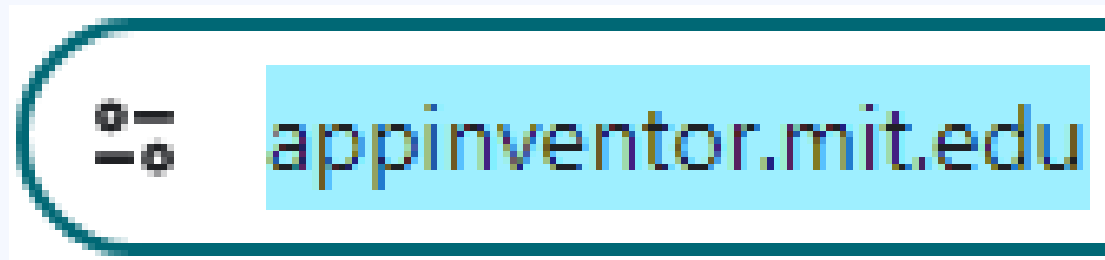
In this tutorial, you will build a simple app with MIT App Inventor that allows users to connect to ChatGPT, ask questions, and have a conversation.

在本教程中，您将使用 MIT App Inventor 构建一个应用程序，允许用户连接到 ChatGPT、提出问题并进行对话。

Open a Browser and put in this  
Link:

打开浏览器并输入此链接：

[https://ai2.appinventor.mit.edu/?locale=en&repo=http://appinventor.mit.edu/yrtoolkit/yr/aiaFiles/SimpleChatBot/simpleChatBot\\_Starter.asc#5967332813897728](https://ai2.appinventor.mit.edu/?locale=en&repo=http://appinventor.mit.edu/yrtoolkit/yr/aiaFiles/SimpleChatBot/simpleChatBot_Starter.asc#5967332813897728)



# Now Follow my screen:

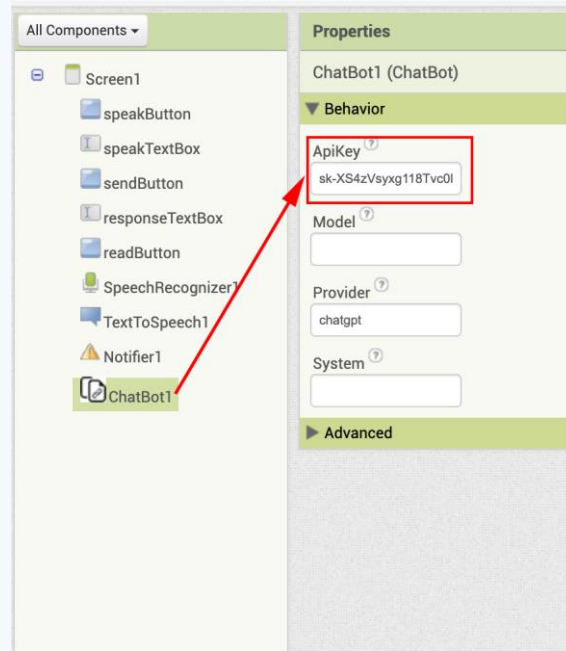
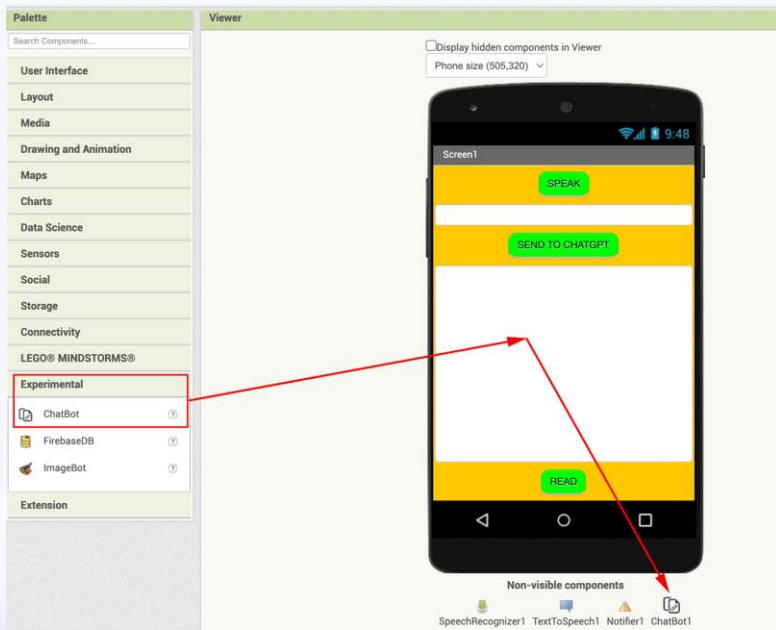
现在关注我的屏幕：

Use my API key:

使用我的 API 密钥：

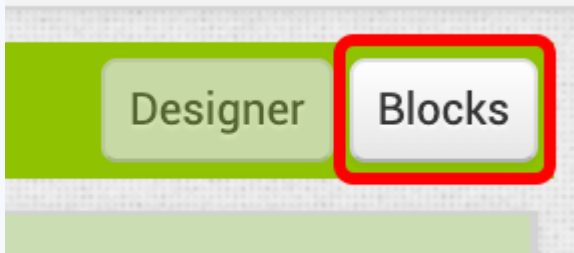
**sk-proj-GrqYbL7oJtvpIi7bkzXGT3B1bkFJDJ8EdbUwJcZ61QJZcuxW**

(I'll send it in chat)



# Do this:

## 做这个：



The screenshot shows the MIT App Inventor interface for a project named 'simpleChatBot\_Starter'. The 'Blocks' palette is open, and a 'when SpeakButton.Click' event is being programmed with several text manipulation blocks. The blocks are:

- when SpeakButton.Click
- do
- set speakTextBox.Text to
- set responseTextBox.Text to
- call SpeechRecognizer1.GetText

The 'Blocks' palette on the left contains the following categories and blocks:

- Built-in
  - Control
  - Logic
  - Math
  - Text
    - length
    - is empty
    - compare texts
    - trim
    - uppercase
    - starts at text piece
    - contains text piece
    - split text at
    - split at spaces
    - segment text
      - start
      - length
    - replace all text
      - segment
      - replacement
    - Obfuscated Text
    - is a string? thing
    - reverse
    - replace all occurrences
  - Lists
  - Dictionaries
  - Colors
  - Variables
  - Procedures
- Screen1
  - speakButton
  - speakTextBox
  - sendButton
  - responseTextBox
  - readButton
  - SpeechRecognizer1
  - TextToSpeech1

The 'Media' section at the bottom left has an 'Upload File...' button.

# Do this:

## 做这个：

The screenshot displays the MIT App Inventor web-based IDE for a project named "simpleChatBot\_Starter". The interface is divided into several sections:

- Top Bar:** Contains navigation and project management options: "Toggle Tutorial", "Screen1", "Add Screen...", "Remove Screen", "Project Properties", "Publish to Gallery", "Designer", and "Blocks".
- Left Panel (Blocks):** A categorized list of components for drag-and-drop. Categories include Math, Text, Lists, Dictionaries, Colors, Variables, Procedures, and Screen1. Under "Screen1", components like speakButton, speakTextBox, sendButton, responseTextBox, readButton, SpeechRecognizer1, TextToSpeech1, and Notifier1 are listed. A "ChatBot1" component is also visible under "Any component".
- Right Panel (Viewer):** The main workspace for building the app. It contains two event-driven code blocks:
  - when speakButton.Click:** A sequence of three actions: "set speakTextBox.Text to", "set responseTextBox.Text to", and "call SpeechRecognizer1.GetText".
  - when SpeechRecognizer1.AfterGettingText:** A sequence of two actions: "result partial" and "set speakTextBox.Text to get result".
- Bottom Panel:** Includes a "Media" section with an "Upload File..." button, a "Show Warnings" button with warning and error icons, and a vertical toolbar with zoom in (+), zoom out (-), and a trash icon.



# Do this:

## 做这个：

The screenshot displays the Visual Studio Code interface for a project named "simpleChatBot\_Starter". The interface is divided into several sections:

- Blocks:** A sidebar on the left containing various code blocks categorized by type (Math, Text, Lists, Dictionaries, Colors, Variables, Procedures) and screen components (Screen1, speakButton, speakTextBox, sendButton, responseTextBox, readButton, SpeechRecognizer1, TextToSpeech1, Notifier1, ChatBot1).
- Viewer:** The main workspace showing three event-driven code blocks:
  - when speakButton .Click:** Sets the text of `speakTextBox` and `responseTextBox` to empty strings, then calls `SpeechRecognizer1 .GetText`.
  - when SpeechRecognizer1 .AfterGettingText:** Sets the text of `speakTextBox` to the result of `get result`.
  - when sendButton .Click:** Checks if `speakTextBox .Text` is not empty. If true, it shows a progress dialog with the message "Generating a response" and title "Please wait...", then calls `ChatBot1 .Converse` with the text from `speakTextBox`.
- Media:** A section at the bottom left with an "Upload File ..." button.
- Bottom:** A "Show Warnings" button and a link for "Privacy Policy and Terms of Use".

# Do this:

## 做这个：

The screenshot displays the Visual Studio Code interface for a project named "simpleChatBot\_Starer". The interface is divided into several sections:

- Blocks Panel (Left):** Contains a list of available blocks categorized by type: Math, Text, Lists, Dictionaries, Colors, Variables, Procedures, and Screen1. Under "Screen1", there are components like speakButton, speakTextBox, sendButton, responseTextBox, readButton, SpeechRecognizer1, TextToSpeech1, Notifier1, and ChatBot1. A "Media" section at the bottom left has an "Upload File..." button.
- Viewer (Center):** Shows a visual programming workspace with four event-driven code blocks:
  - when speakButton .Click:** A "do" block containing "set speakTextBox .Text to", "set responseTextBox .Text to", and "call SpeechRecognizer1 .GetText".
  - when SpeechRecognizer1 .AfterGettingText:** A "do" block containing "result partial" and "set speakTextBox .Text to get result".
  - when sendButton .Click:** A "do" block containing an "if not is empty speakTextBox .Text" condition. If true, it calls "Notifier1 .ShowProgressDialog" (with message "Generating a response" and title "Please wait...") and "ChatBot1 .Converse" (with question "speakTextBox .Text").
  - when ChatBot1 .GotResponse:** A "do" block containing "call Notifier1 .DismissProgressDialog" and "set responseTextBox .Text to get responseText".
- Bottom Panel:** Includes a "Show Warnings" button with a warning icon and a "0" count, and a "Privacy Policy and Terms of Use" / "Accessibility: accessibility.mt.edu" link.

# Do this:

## 做这个：

The screenshot displays the MIT App Inventor IDE for a project named "simpleChatBot\_Starter". The interface is divided into a "Blocks" palette on the left and a "Viewer" area on the right. The "Blocks" palette includes categories like Math, Text, Lists, Dictionaries, Colors, Variables, Procedures, and Screen1, with various UI components like speakButton, speakTextBox, sendButton, responseTextBox, readButton, SpeechRecognizer1, TextToSpeech1, Notifier1, and ChatBot1. The "Viewer" area shows several event-driven code blocks:

- when speakButton.Click**:
  - do set speakTextBox.Text to
  - do set responseTextBox.Text to
  - do call SpeechRecognizer1.GetText
- when SpeechRecognizer1.AfterGettingText**:
  - do result partial
  - do set speakTextBox.Text to get result
- when SendButton.Click**:
  - do if not empty speakTextBox.Text
    - then call Notifier1.ShowProgressDialog (message: Generating a response, title: Please wait...)
    - call ChatBot1.Converse (question: speakTextBox.Text)
- when ChatBot1.GetResponse**:
  - do responseText
  - do call Notifier1.DismissProgressDialog
  - do set responseTextBox.Text to get responseText

The bottom status bar shows "Show Warnings" and "Privacy Policy and Terms of Use".

# Demo Time!

Try out the app  
for yourself!

Any questions?

演示！

自 一下 用程  
序！

# End of Class 4

**HW: Use the project  
we made**

Stay around if you have any immediate questions or:

Email me at [jpengzhao@gmail.com](mailto:jpengzhao@gmail.com) if you have any questions



# 第 4 课结束

**HW : 使用我们做的  
的项目**

如果您有任何问题：

请给我发电子邮件  
[:jpengzhao@gmail.com](mailto:jpengzhao@gmail.com)