CREATING APPS WITH MIT APP INVENTOR

BRIDGE COURSE

GRADE-6

PRIMARY PREVIEW

- What are Apps?
- Categories of Apps
- Types of UI Components
- Testing Your App
- Features of Mobile Apps
- App Inventor
- Blocks Editor and Its Interface
- Types of Mobile Apps
- Starting MIT App Inventor
- Creating an App

WHAT ARE APPS?

Apps are software programs for smartphones, tablets or computers that perform tasks like learning, gaming, chatting or watching videos. The **app** is a modern term derived from the word **application**. Many apps are free, while some need to be purchased. They're designed to meet various needs, such as gaming, online classes, music, communication and video streaming.

You can download apps from stores like the Google Play Store (for Android) or App Store (for iPhone).

FEATURES OF MOBILE APPS

Mobile apps have many features that make them fun, helpful and easy to use. Here are some of the main features of mobile apps:

- **Easy to use:** Apps are simple and user-friendly, allowing even children to learn quickly by tapping icons.
- **Safe and secure:** Verified apps protect your data and let you control settings like notifications and privacy.
- **Notifications:** Apps can send alerts or reminders, like when you get a message or when it's time to study. These help you stay updated.

- **Touch-friendly:** Most apps are designed for touch, allowing you to tap, swipe or pinch the screen without needing a mouse or keyboard.

TYPES OF MOBILE APPS

Mobile apps are divided into three types: Native Apps, Web Apps and Hybrid Apps. Let's learn about them in detail.

NATIVE APPS

A **native app** is software built for a specific operating system, like iOS or Android. Downloaded from app stores, it runs smoothly by interacting directly with the device's hardware and can often work without the Internet. For example, Instagram, Amazon, WhatsApp, etc.

WEB APPS

A **web app** is a software program that runs on a remote server and is accessed through a web browser over the Internet. It doesn't take up space on your device and updates are automatically applied. For example, Google Docs, Canva, Wikipedia, etc.

HYBRID APPS

A **hybrid app** combines features of both native and web apps, offering cost-effectiveness and easy maintenance. It provides access to device features for better performance. Once downloaded, it connects to the platform's capabilities through an embedded browser. For example, Facebook, Gmail, Netflix, etc.

FACT File

Netflix has over **23 crore** subscribers worldwide.

RAPID RECALL Tick (✓) if you know this.

1. Instagram and WhatsApp are native apps.
2. Hybrid apps are cost-effective and easy to maintain.

CATEGORIES OF APPS

Categories of apps are groups that help you understand the purpose or function of different mobile applications. Each category helps in a different area like:

EDUCATIONAL APPS

Educational apps are specially designed to make learning more interactive for children. They help teach subjects like maths, science, coding and technical skills through lessons, quizzes, e-books and videos. For example, BYJU'S, Duolingo, Scratch Junior, etc.

LIFESTYLE APPS

Lifestyle apps help manage daily routines, health, hobbies and goals, making tasks easier and more enjoyable. They act as personal assistants, guiding better choices in areas like health, shopping and time management. For example, Google Maps, MakeMyTrip, Ola, Uber, etc.

SOCIAL MEDIA APPS

Social media apps let users create, share and exchange content within virtual communities. They allow sharing of live videos, images and conversations and are also used for business promotion. For example, Facebook, Instagram, X (formerly Twitter), etc.

GAMING APPS

Gaming apps provide entertainment, allowing users to play on phones, tablets or computers. They offer fun, skill improvement and competition with others, including sharing scores on social media. For example, Candy Crush, Angry Birds, Subway Surfers, etc.

FACT File

Mobile apps account for over **60%** of total digital media time globally.

ART INTEGRATION ACTIVITY

Create a poster that represents different categories of apps (e.g., educational, gaming, social media, lifestyle) using drawings, colours and labels. Choose one app from each category and illustrate its features. Present your poster to the class.

APP INVENTOR

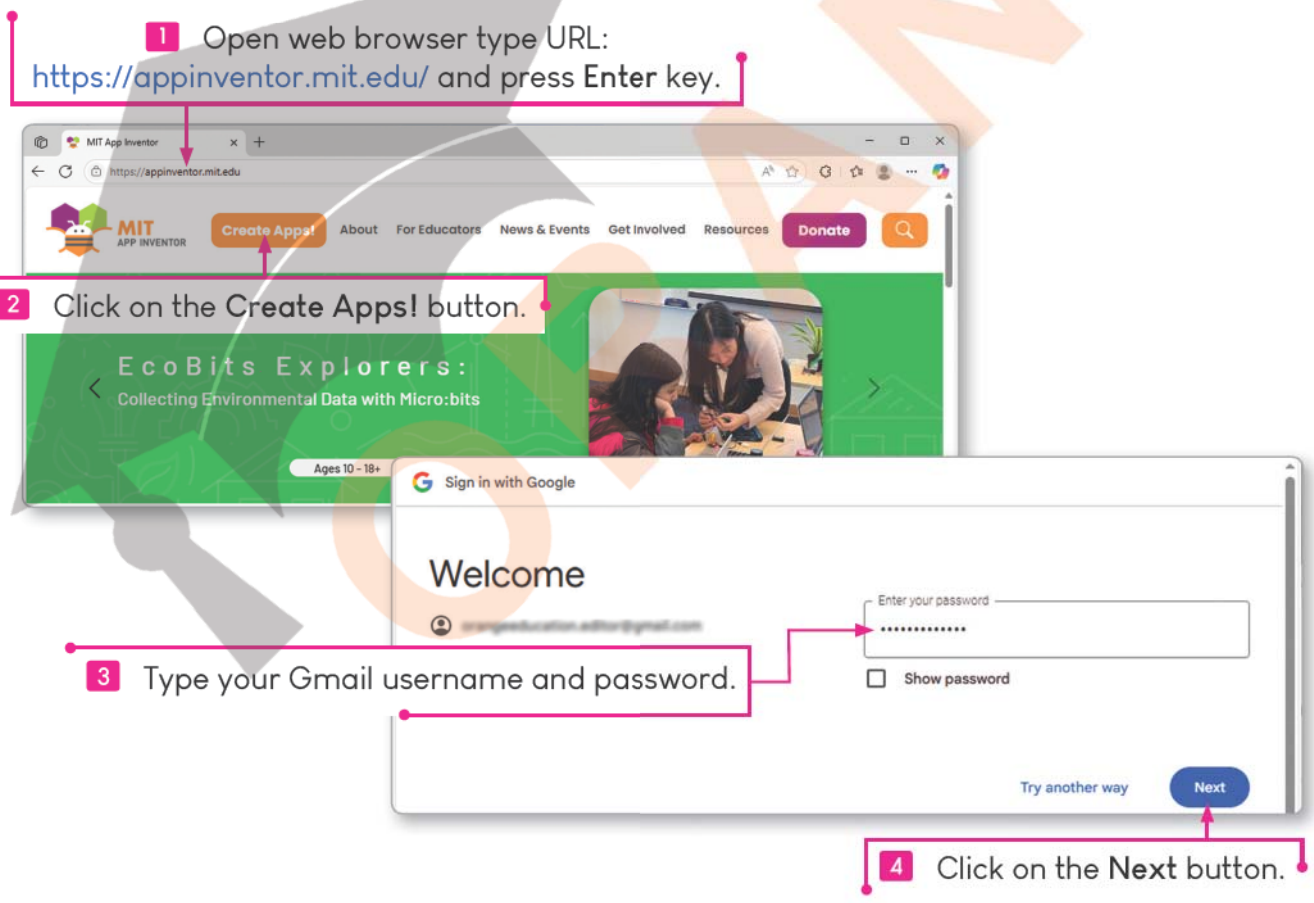
MIT App Inventor is a user-friendly tool that lets you create mobile apps without complex coding. You can design the app's layout, choose features and add actions using simple blocks, similar to Scratch programming. It is free, cloud-based and accessible via web browsers like Chrome, Firefox or Safari.

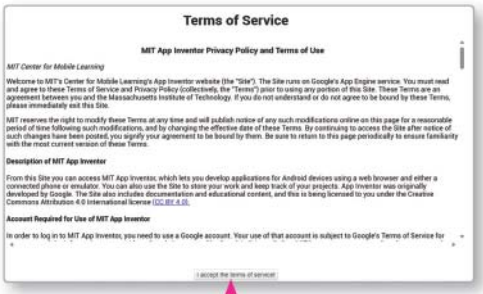
The user interface of MIT App Inventor consists of two main sections that you will use to create your app:

1. **Designer view:** This view contains all the components required to design an application.
2. **Block editor view:** This view is the place where you combine blocks to execute an application.

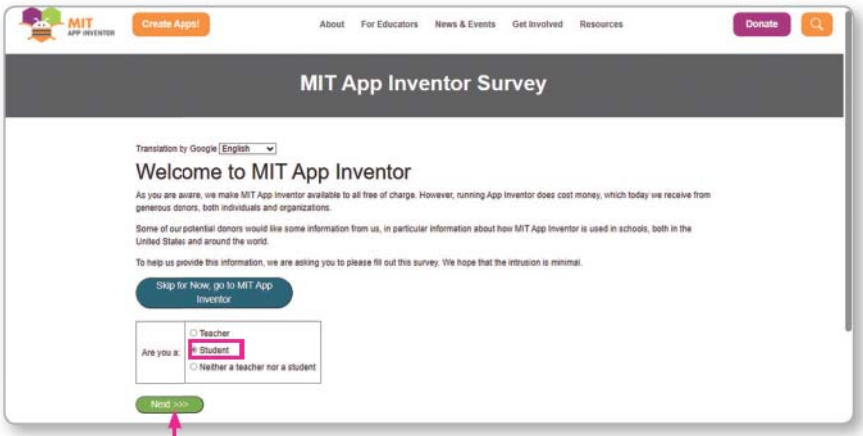
STARTING MIT APP INVENTOR

You should have a **Gmail account** to use the MIT App Inventor. To start the app, follow the given steps:



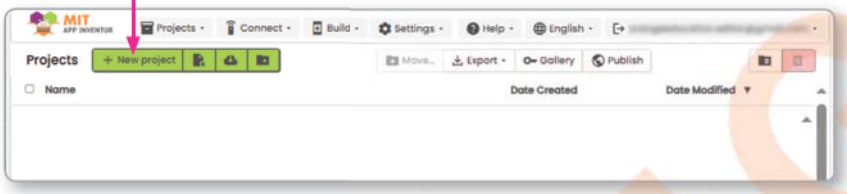


5 Click on the **I accept the terms of service!**.

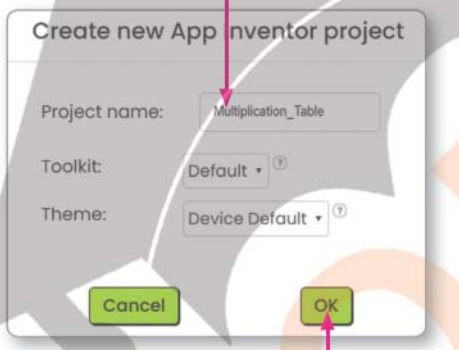


6 Choose student and press the **Next** button.

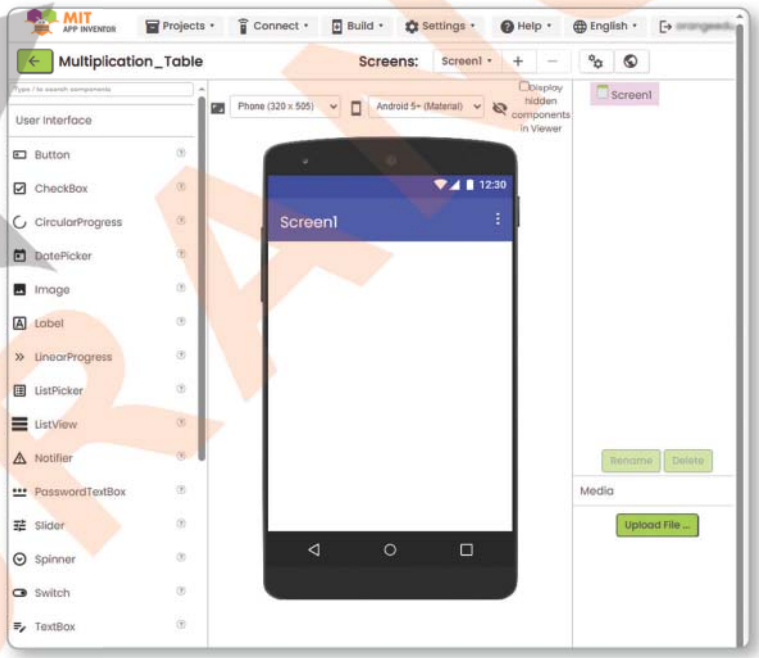
7 Click on the **New project** button to start your first project.



8 Type the name of your project in the **Project name** box.



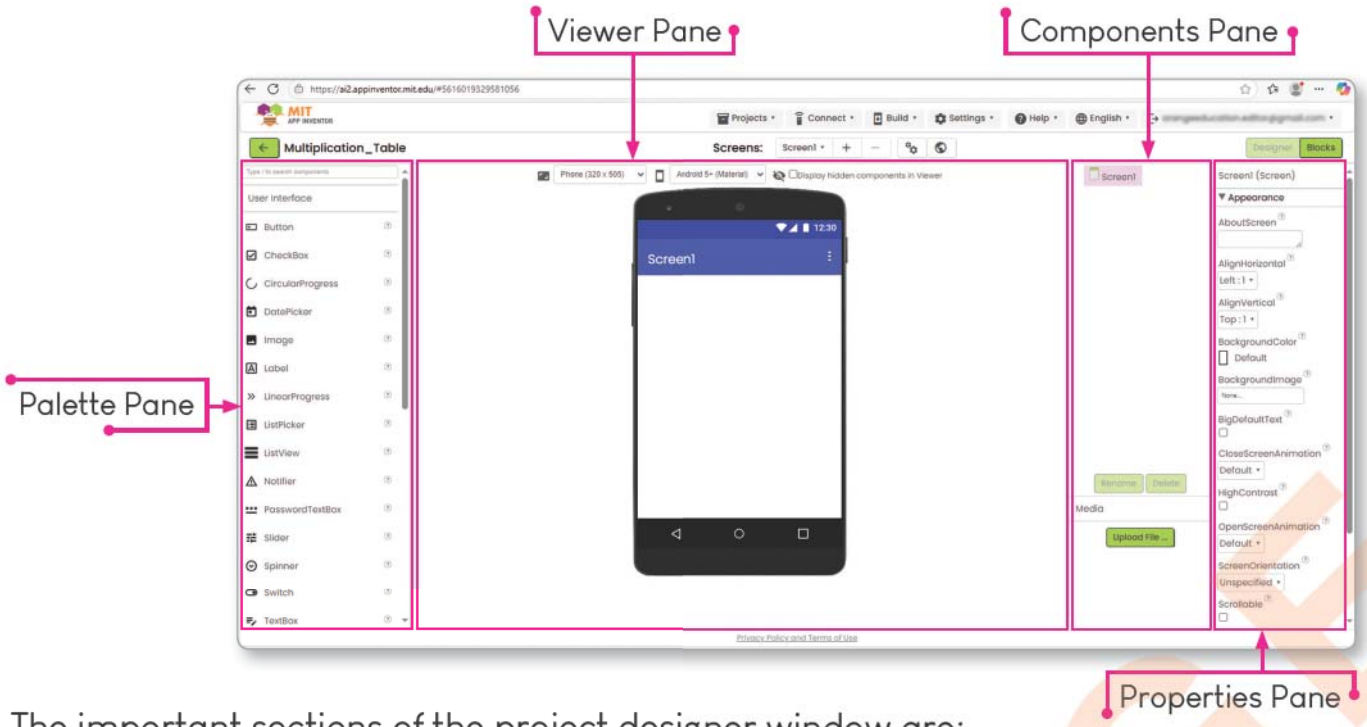
9 Click on the OK button.



The final project designer window appears.

PROJECT DESIGNER WINDOW OF MIT APP INVENTOR

The **Project Designer window** in the MIT App Inventor is where you create and design the user interface (UI) of your app. It is the main workspace for building the visual layout of your app and arranging how components interact.

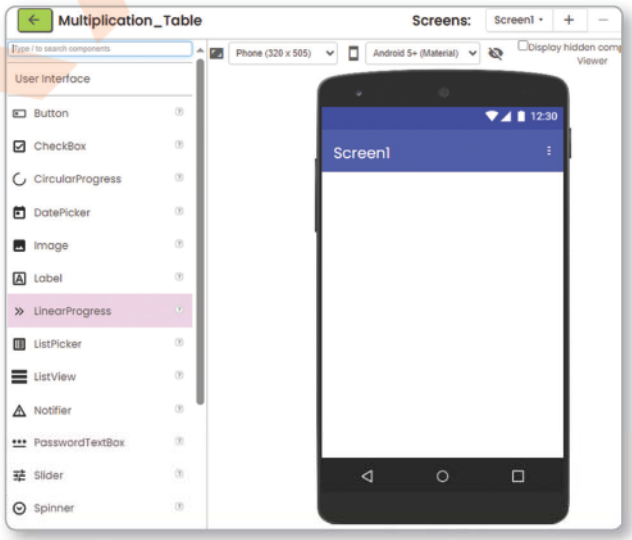


The important sections of the project designer window are:

- **Palette Pane:** Contains components like buttons and images to add to your app.
- **Viewer Pane:** The white area to arrange and view components.
- **Components Pane:** Lists all added components in a hierarchy.
- **Properties Pane:** Shows and lets you change the properties of selected components.

TYPES OF UI COMPONENTS

User Interface (UI) components are the visual elements that users can see and interact within your app. It provides a wide range of UI components that you can drag and drop to design your app.



Here's a list of the most commonly used UI components in MIT App Inventor:

1. **Buttons:** Triggers an action when tapped, like opening a screen or showing a message.
2. **Label:** Displays text such as messages or instructions.

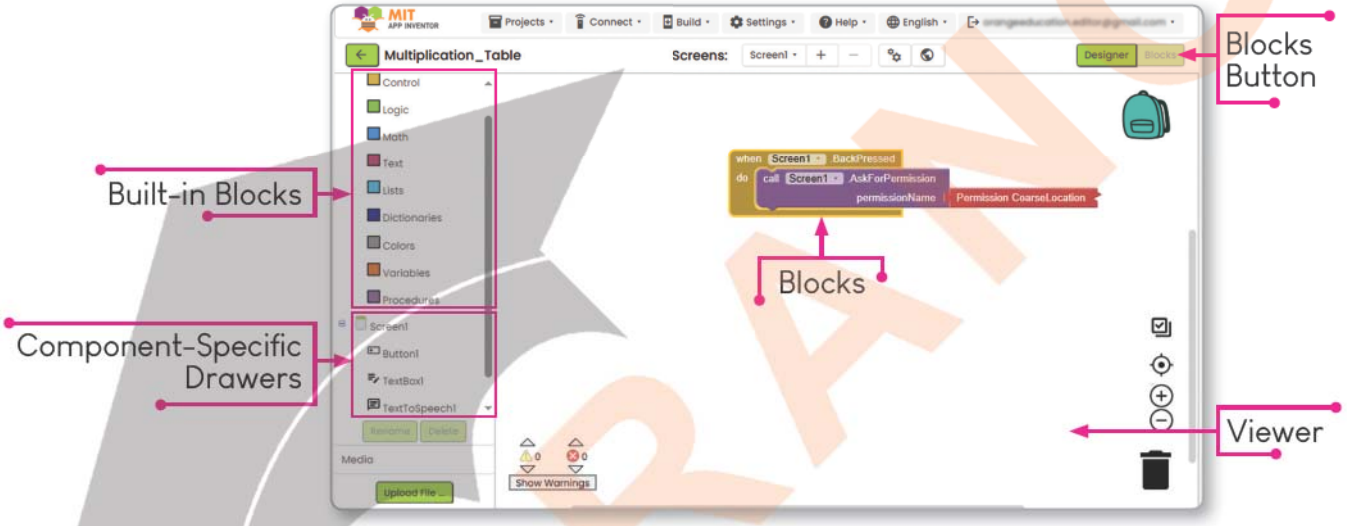
3. **TextBox:** Lets users enter text, like a name or email.
4. **Image:** Displays pictures on the screen.
5. **ListView:** Shows a list of items such as text, images or buttons.
6. **Switch:** A toggle with On and Off states, similar to a checkbox.

HINTS & HACKS

You can search for any UI components that are not visible on the home screen by typing their name in the **search box**.

BLOCKS EDITOR AND ITS INTERFACE

In MIT App Inventor, the **Blocks Editor** is where you design your app's actions. You can drag and drop blocks to make things happen, like showing a message or changing the screen, much like assembling a puzzle. The Blocks Editor lets you program the app's behaviour by putting blocks together.



The important sections of the blocks editor window:

- **Built-in Blocks:** These are the basic blocks in MIT App Inventor (e.g., Control, Math, Text) used to program app behaviour.
- **Component-Specific Drawers:** Each component (e.g., Button, Label) has its own set of blocks for controlling actions and settings.
- **Blocks Button:** Switches you from the design area to the blocks area to write the app's code.
- **Viewer:** The viewer is where you drag and drop blocks to create your app's functionality by connecting them like puzzle pieces.

DIFFERENT BLOCKS OF BLOCKS EDITOR

Built-in blocks are always available and provide basic tools for logic, text, variables and more in the **Blocks Editor**.

1. **Control Blocks:** These blocks manage the flow of the program using conditions, loops and event triggers. They help you decide what happens and when. Example:

Tests a given condition. If the condition is true, it runs the specified blocks; if not, it skips them.

Tests a given condition. If it is true, the blocks in the then part run; if it is false, the blocks in the else part run.

2. **Logic Blocks:** These blocks allow you to make decisions using true/false values. You can compare values and use logical operations like AND, OR and NOT. Example:

Represents the constant value true. It is used to set the Boolean properties of components or assign a condition value to a variable.

Represents the constant value false. It is used to set the Boolean properties of components or assign a variable to represent a condition.

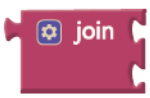
3. **Math Blocks:** These blocks perform basic and advanced mathematical calculations. You can add, subtract, multiply, divide and more. Example:

This block lets you display numbers in decimal form and can also show them in other formats, such as binary or hexadecimal.

This block compares two values to check if one is greater than, less than or equal to the other.

4. **Text Blocks:** These blocks are used to work with words and characters. You can join, split, change and compare text. Example:

This string can contain any characters (letters, numbers or other special characters). On App Inventor it will be considered a Text object.



Appends all of the inputs to make a single string. If there are no inputs, it returns an empty string.

5. **Lists Blocks:** These blocks help you store and manage collections of items. You can add, remove or find items in a list. Example:

Creates an empty list with no element.

Returns the number of items in the list.

6. **Dictionaries Blocks:** These blocks store data in key-value pairs. They help you organise and access information using labels. Example:

This block creates a dictionary without any key-value pairs.

This block removes the key-value mapping in the dictionary for the given key.

7. **Colors Blocks:** These blocks allow you to pick and use colours in your app. You can set background colours or text colours using these. Example:

This basic colour block is a small square block that shows a specific colour in the middle, representing the colour stored inside it.

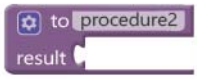
This block takes in a list of 3 or 4 numbers. These numbers in this list represent values in an RGB code.

8. **Variables Blocks:** These blocks store values that can change while the app runs. You can create, set and get variable values. Example:

This block allows you to access and use the current value stored in a variable.

This block assigns a new value to a variable. Like the **get** block, only variables within scope appear in the drop-down.

9. **Procedures Blocks:** These blocks let you create reusable sets of instructions. They help you avoid repeating code and keep your app organised.
   Example:



Collects a sequence of blocks together into a group.

Same as a procedure do block, but this procedure returns a result when called.

CREATING AN APP

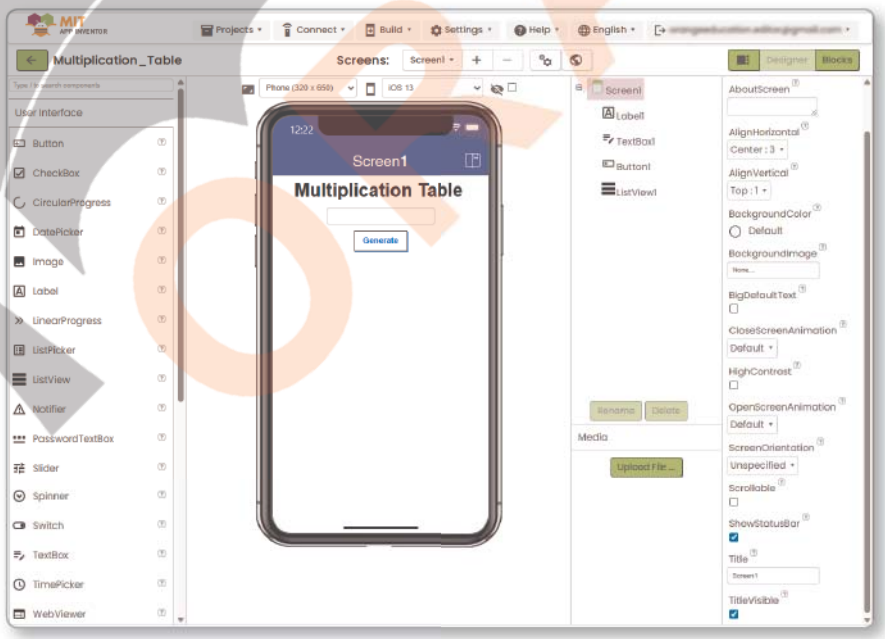
In this activity, you will create an app that shows a multiplication table when a number is entered. The app will display the results of multiplying the number by 1 to 10.

To make the Multiplication Table App, follow the given steps:

Step 1 – Click on the **New Project** button and name it **Multiplication_Table**.

Step 2 – Click on the Screen1 option and set **AlignHorizontal to Center: 3 and AlignVertical to Top 1.**

Step 3 – Drag a **Label** from the Palette to the Viewer, then set its properties: Text to **Multiplication Table**, **FontSize** to 30, check **FontBold** and align it to Center.



Step 4 Add a **TextBox** for input by dragging it from the **Palette** to the **Viewer**, then set its Hint to **Enter a number**.

Step 5 Drag a **Button** from the **Palette** to the **Viewer**, then set its **Text** to **Generate**.

Step 6 Drag a **ListView** from the **Palette** to the **Viewer**. The **ListView** will display the multiplication table after the user enters a number and clicks the **Generate** button.

Step 7 Click on the **Blocks** in the top-right corner to switch to the **Blocks Editor** and start coding the app's functionality.

Step 8 Click on the **Variables** from the Blocks menu. The list of coding blocks appears.

Step 9 Create the Global Variable List, drag the **initialize global name to** block the **Viewer**, then click on the **name section** and rename it to **List**.

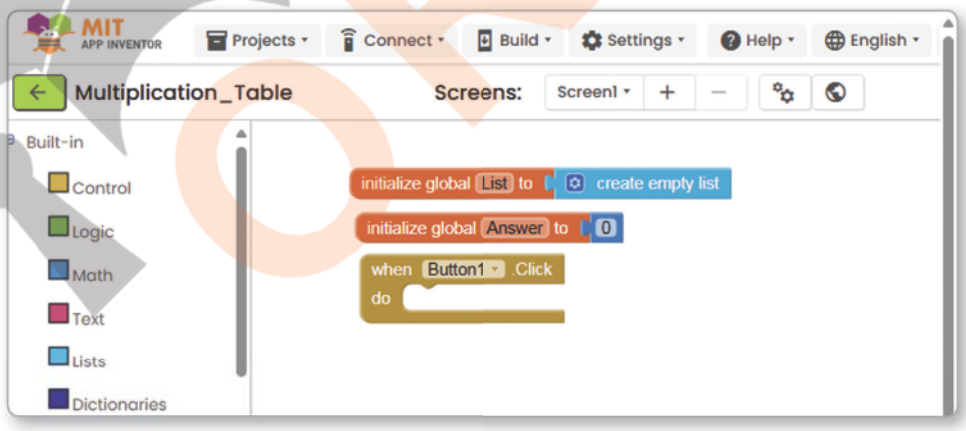
Step 10 Add an Empty List from the **Lists** category, drag the **create empty list** block and connect it to the **List** variable.

Step 11 Drag the **initialize global name to** block into the **Viewer**, then click on the **name section** and rename it to **Answer**.

Step 12 From the **Math** category, drag the **number block (0)** and connect it to the **Answer** variable.

Step 13 Click on the **Button1** from the list of components. The list of coding blocks appears.

Step 14 Drag the **when Button1**. Click block into the **Viewer**.



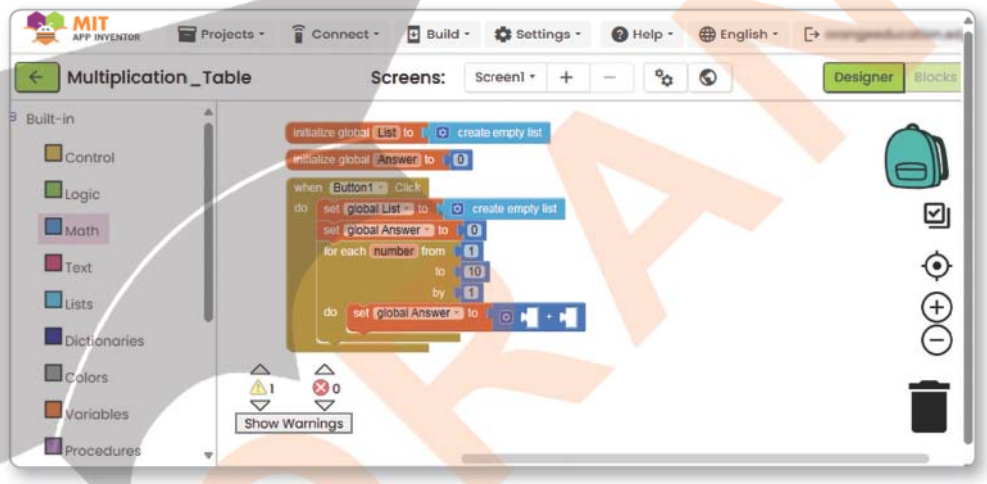
Step 15 Inside the **Button1** block, drag **set global List to** and connect **create empty list** from the Lists category.

Step 16 Drag **set global Answer to** and connect the **number block (0)** from the **Math** category.

Step 17 Add a Loop for multiplication, inside the **Button1** block, drag the block '**for each number from 1 to 10 by 1**' from the **Control** category. This loop will repeat 10 times, calculating the multiplication values from 1 to 10.

Step 18 Drag the **set global Answer to** block from the **Variables** section.

Step 19 Add the multiplication logic from the **Math** category, drag the **addition block (+)** and connect it to **set global Answer to**.



Step 20 Set the first slot to **get global Answer** from the **Variables** category, drag **get global Answer** and place it in the first slot of the addition block.

Step 21 Add the input number from the **TextBox1** section, drag **TextBox1. Text** and place it in the second slot of the addition block.

Step 22 From the **Lists category**, drag the **add items to list** block into the **Viewer** and connect it to **global List** to store the multiplication results.

SHORT SIGN

To Refresh Companion Screen: Alt + R

Step 23 Use the **join** block to format the multiplication statement from the **Text** category, drag the **join** block and connect it to the item socket of the **add items to list** block.

Step 24 Add **TextBox1** input as the first number from the **TextBox1** category, drag **TextBox1**. **Text** and place it in the first input of the **join** block.

Step 25 Add the multiplication symbol (×) from the **Text** category, drag a text block (empty quotes " ") and type ×, then connect it as the next input in the **join** block.

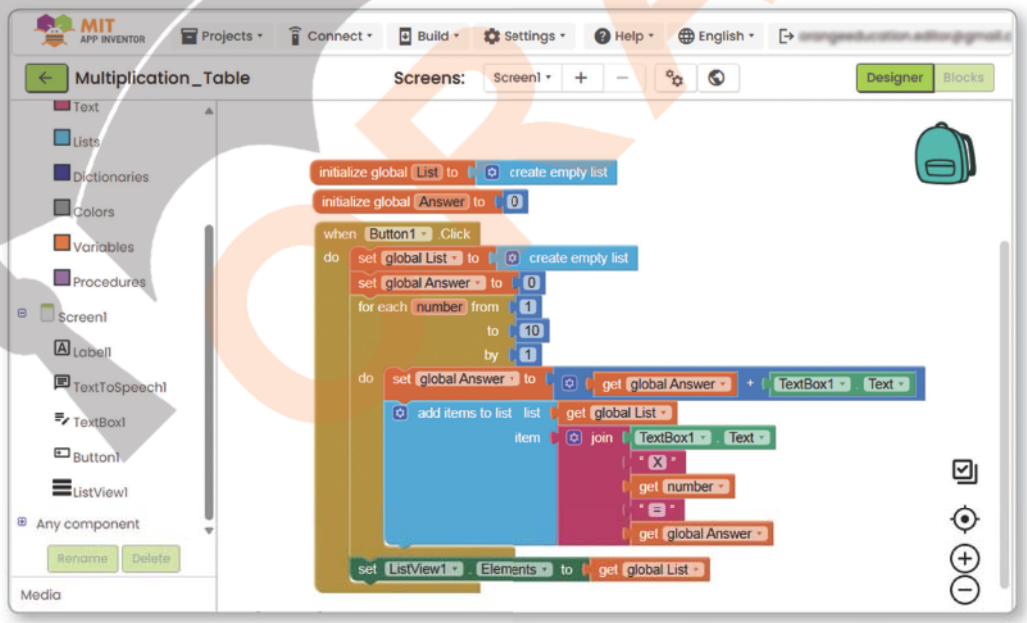
Step 26 Add the loop number from the **Variables** category, drag "**get number**" and place it in the **join** block after the × symbol. This represents the multiplier (1 to 10).

Step 27 Add the equals symbol (=) from the **Text** category, drag another text block, type = and connect it as the next input in the **join** block.

Step 28 Add the computed answer from the **Variables** category, drag the **get global Answer** and place it as the final input in the **join** block. The **join block** is used to format the multiplication equation (e.g., 5 × 1 = 5).

Step 29 Inside the **ListView1 Click** block, Drag the **set ListView1. Elements to** block into the **Viewer** and connect the **get global List** to it. This step updates the **ListView** to display the multiplication table stored in the **global List**.

Here is the final preview of the code.



TESTING YOUR APP

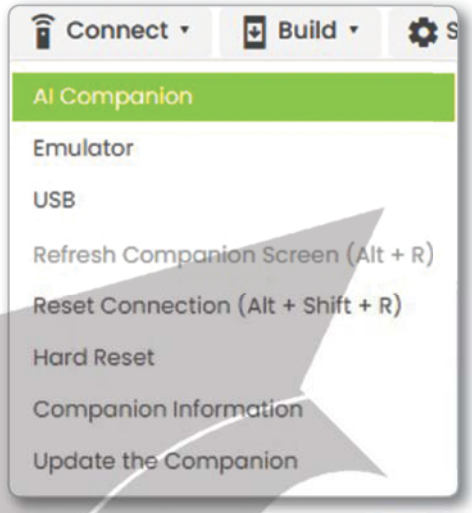
After completing the design and blocks for your **Multiplication_Table App**, follow the given steps to test it on your mobile device:

Step 1 – Download and install the **MIT AI2 Companion** app on your mobile phone from the **Google Play Store**.

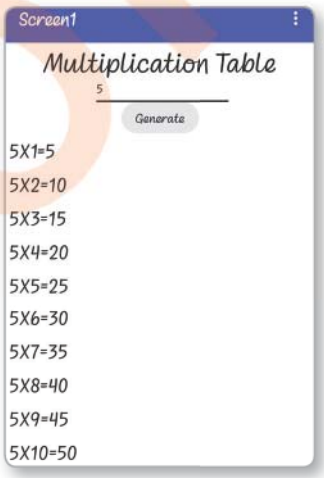
Step 2 – Click on the **Connect** from the top menu.

Step 3 – Click on the **AI Companion** from the drop-down list. The **Connect to Companion** message box appears.

Step 4 – Open the **MIT A12 Companion** on your device and connect by scanning the QR code or typing the code displayed on your screen.



Once you finish testing the app, you can start using it. Make sure it generates the multiplication table correctly, is easy to use and displays the results properly in the **ListView**.



HINTS & HACKS

To make your app for Android, click on the **Build button** and choose **Android App** (.apk). Then, download the file when it's ready.

Ask AI AGENT

Design a Quiz App using MIT app inventor where users answer questions and get immediate feedback.

+ | Study

TECH TERMS

- **Operating System (OS):** The software that runs on your phone, tablet or computer and helps manage all the apps and hardware.
- **Cloud-based:** A type of service that stores data or runs applications over the Internet rather than on your device.
- **QR code:** A type of barcode that you can scan with your phone to quickly access websites, apps or information.
- **Loop:** A programming tool that repeats a set of actions over and over again until a condition is met.

REWIND RUN

- Apps are software programs that run on mobile devices, providing various functionalities.
- Mobile apps are divided into three types: Native Apps, Web Apps and Hybrid Apps.
- App categories include educational apps for learning, lifestyle apps for managing daily activities, social media apps for connecting with others and gaming apps for entertainment and competition.
- MIT App Inventor allows users to create apps with a visual, block-based interface.
- The User Interface (UI) and Blocks Editor are key components for designing and programming apps.
- You can test your app using the MIT AI2 Companion on a mobile device to check its functionality.