



ROBOTİK BİLİM



PROJECT-1 TRANSLATOR

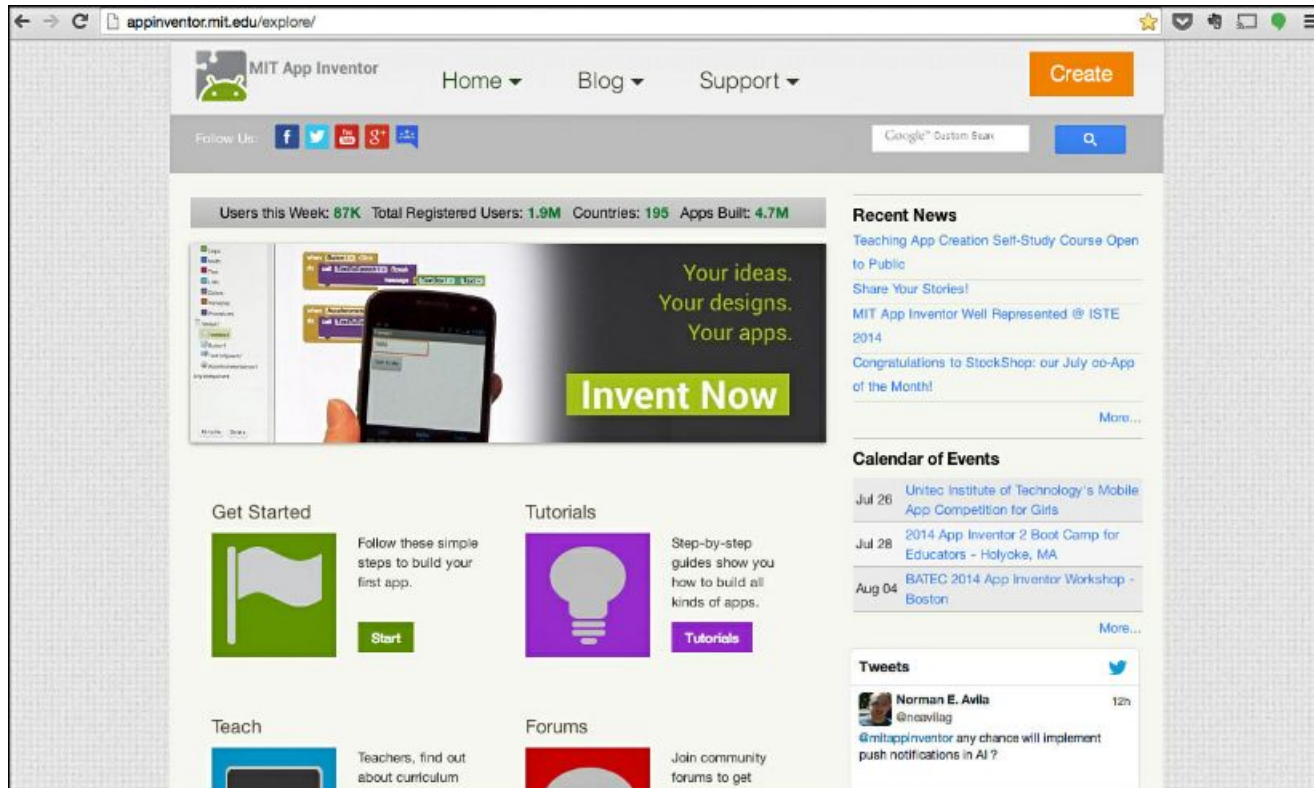


ROBOTIKBİLİM



APP INVENTOR

BUILDING APPLICATIONS WITH APP INVENTOR



- App Inventor works over the internet.
- You do not need to download any other program.
- The web browser you use is important because it works over the Internet.
- Google Chrome or firefox is the most suitable web browser.
- You can login at appinventor.mit.edu.



ROBOTIKBİLİM



← → ↻ appinventor.mit.edu

 [Create Apps!](#) [About](#) [Educators](#) [News](#) [Resources](#) [Blogs](#) [Give](#) ENHANCED BY GOC

Anyone can create Android and iPhone apps with global impact

[Learn More](#)

Active Users today: 103.0K	Active Users this week: 351.7K	Active Users this month: 1.2M	Registered Users: 14.9M	Countries: 195	Apps Built: 67.8M
---	---	--	--------------------------------------	--------------------------	-----------------------------

MIT Master Trainer
in
Educational Mobile Computing



Become an
MIT Master Trainer in Educational Mobile Computing

Program resumes fully virtual online on March 1, 2022.
Registration starts February 1, 2022. [Click here](#) to learn more.

BUILDING APPLICATIONS WITH APP INVENTOR

After logging in to AppInventor, we press

[Create Apps!](#)

the menu to program our first

application.



ROBOTIKBİLİM



BUILDING APPLICATIONS WITH APP INVENTOR

- Since Appinventor works on Android, you need to have a Google account.
- If you do not have a Google account, you must create one from the menu. [Create an account](#)
- If you have a Google account, you must enter your account and password and log in.

Google

Sign in to add another account

Email

Password

Sign In

Need help?

Create an account

One Google Account for everything Google



ROBOTIKBİLİM



To use App Inventor for Android, you must accept the following terms of service.

Terms of Service

MIT App Inventor Privacy Policy and Terms of Use

MIT Center for Mobile Learning

Welcome to MIT's Center for Mobile Learning's App Inventor website (the "Site"). The Site runs on Google's App Engine service. You must read and agree to these Terms of Service and Privacy Policy (collectively, the "Terms") prior to using any portion of this Site. These Terms are an agreement between you and the Massachusetts Institute of Technology. If you do not understand or do not agree to be bound by these Terms, please immediately exit this Site.

MIT reserves the right to modify these Terms at any time and will publish notice of any such modifications online on this page for a reasonable period of time following such modifications, and by changing the effective date of these Terms. By continuing to access the Site after notice of such changes have been posted, you signify your agreement to be bound by them. Be sure to return to this page periodically to ensure familiarity with the most current version of these Terms.

Description of MIT App Inventor

From this Site you can access MIT App Inventor, which lets you develop applications for Android devices using a web browser and either a connected phone or emulator. You can also use the Site to store your work and keep track of your projects. App Inventor was originally developed by Google. The Site also includes documentation and educational content, and this is being licensed to you under the Creative Commons Attribution 4.0 International license ([CC BY 4.0](#)).

Account Required for Use of MIT App Inventor

In order to log in to MIT App Inventor, you need to use a Google account. Your use of that account is subject to Google's Terms of Service for

I accept the terms of service!

BUILDING APPLICATIONS WITH APP INVENTOR

- After logging in to your Google account, a screen will appear about confirming the terms of use.
- After confirming all the conditions, you can start the first project.



ROBOTIKBİLİM



MIT APP INVENTOR

Projects ▾ Connect ▾ Build ▾ Settings ▾ Help ▾

My Projects View Trash Guide Report an Issue English ▾ recep@robotikbilim.com.tr ▾


Start new project Move To Trash View Trash Login to Gallery Publish to Gallery

Projects

<input type="checkbox"/>	Name	Date Created
--------------------------	------	--------------

Welcome to App Inventor!

Welcome to MIT App Inventor



join

MIT App Inventor

Release

nb188

Release nb188 out

[Read About It](#)

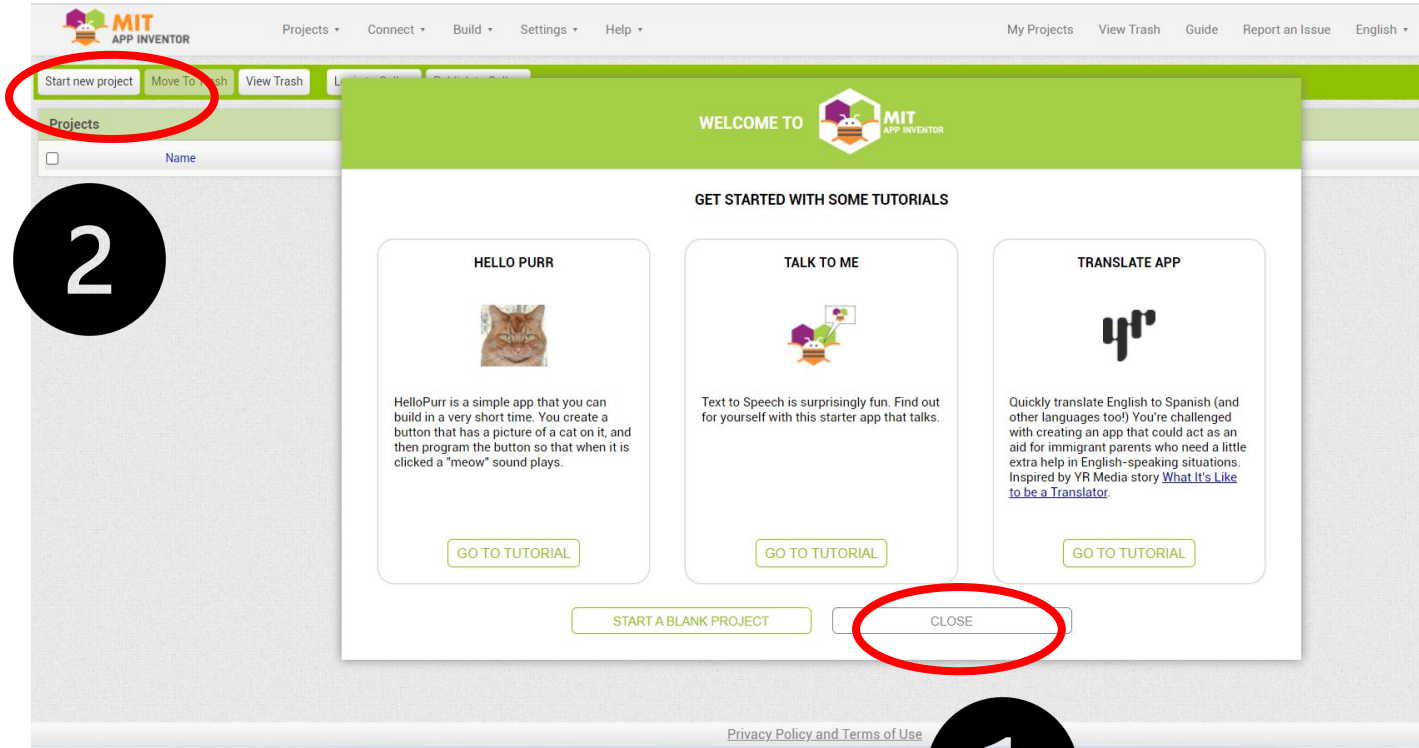
Got an iPhone or Android phone? Find out how to [set up and connect an iOS or Android device.](#)

Continue Do Not Show Again

Privacy Policy and Terms of Use



ROBOTIKBİLİM

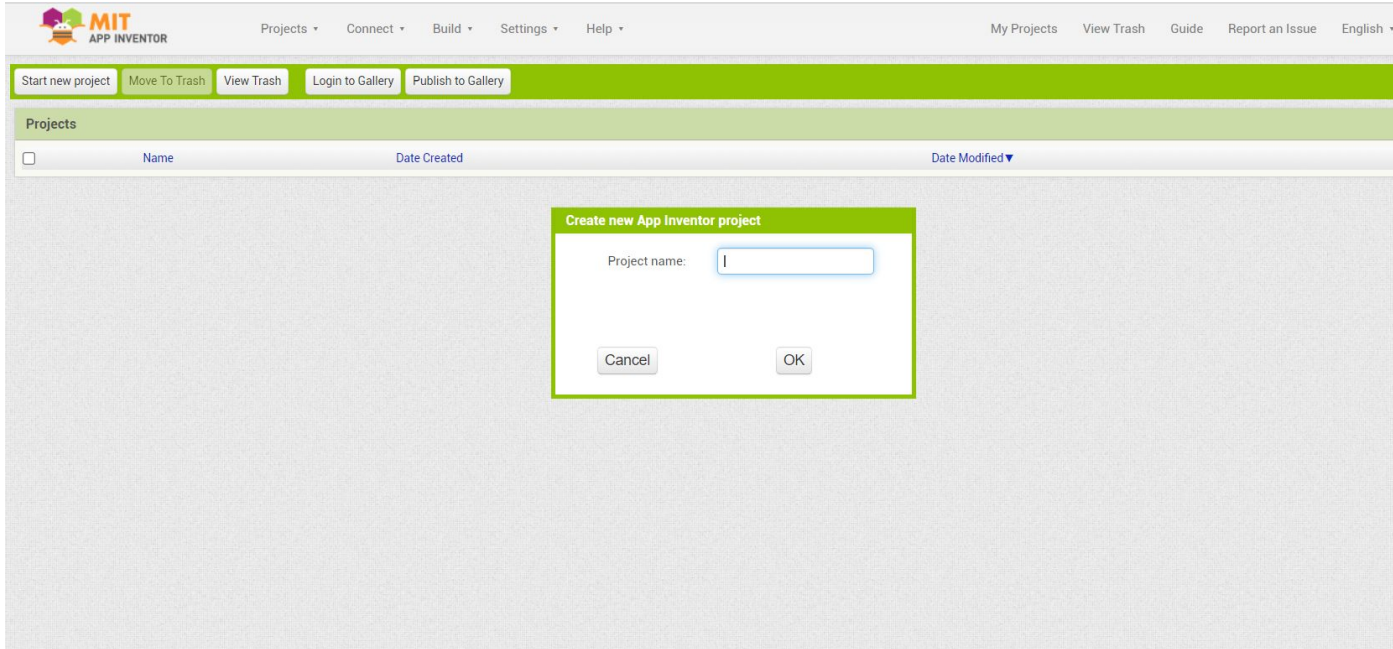


BUILDING APPLICATIONS WITH APP INVENTOR

- After logging in to the main screen, 3 tutorials appear. You can find detailed information about appinventor with these tutorials.
- Let's start creating our first project by pressing the Start a Blank Project button.

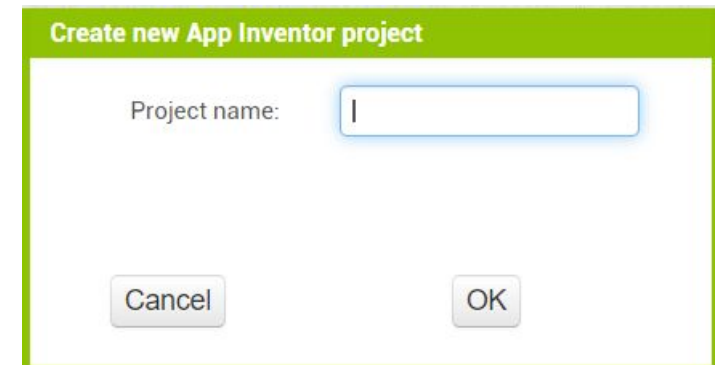


ROBOTIKBİLİM



Application 1: TRANSLATOR

- Before we start programming the Mobile App, we must first give our project a suitable name.
- The name we will give will also be the name that will appear on our phone.
- **App Inventor does not allow spaces and punctuation in the project name.**





Create new App Inventor project

Project name:

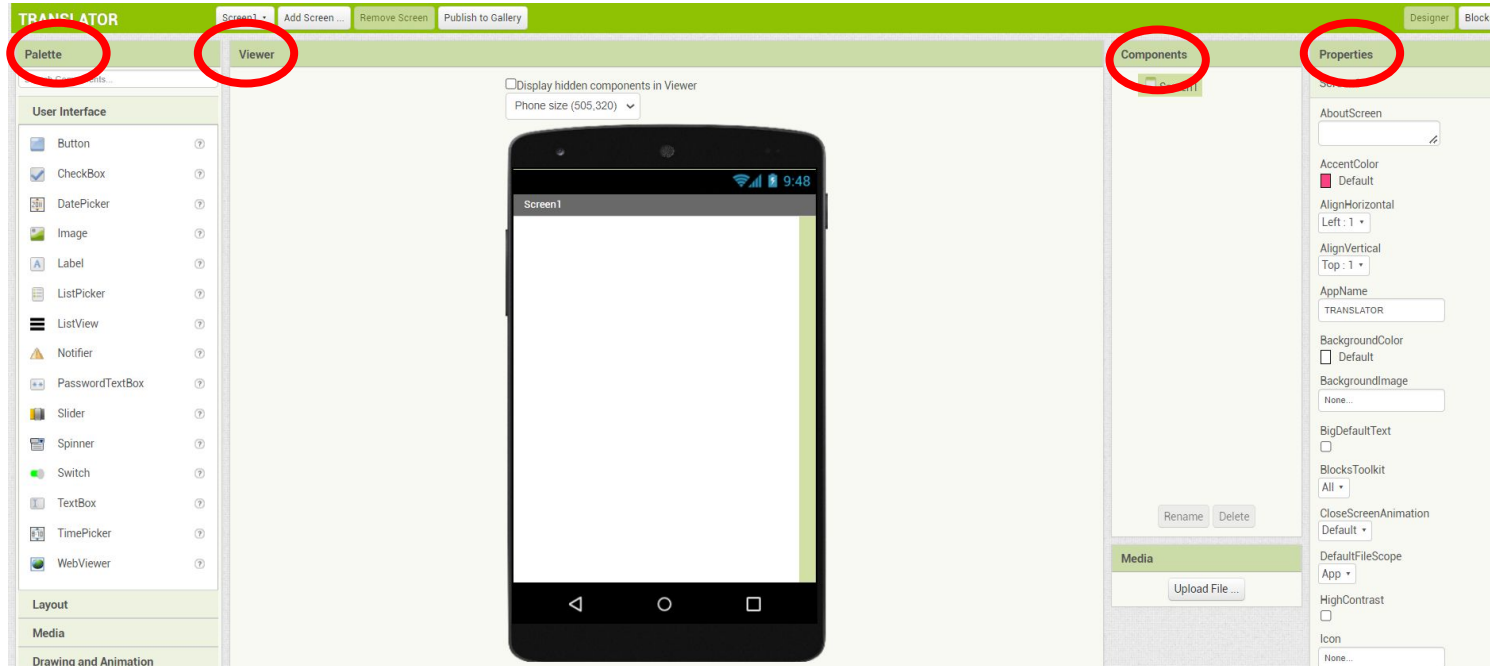
Cancel OK

Application 1: TRANSLATOR

- Let the name of our first project be TRANSLATOR.
- Let's write the name as seen on the screen and click the ok button.



ROBOTIKBİLİM



Application 1: TRANSLATOR

Home Page: Designer

- The main page consists of 4 sections.





PALETS

User Interface

Layout

Media

Drawing and Animation

Maps

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Experimental

Extension

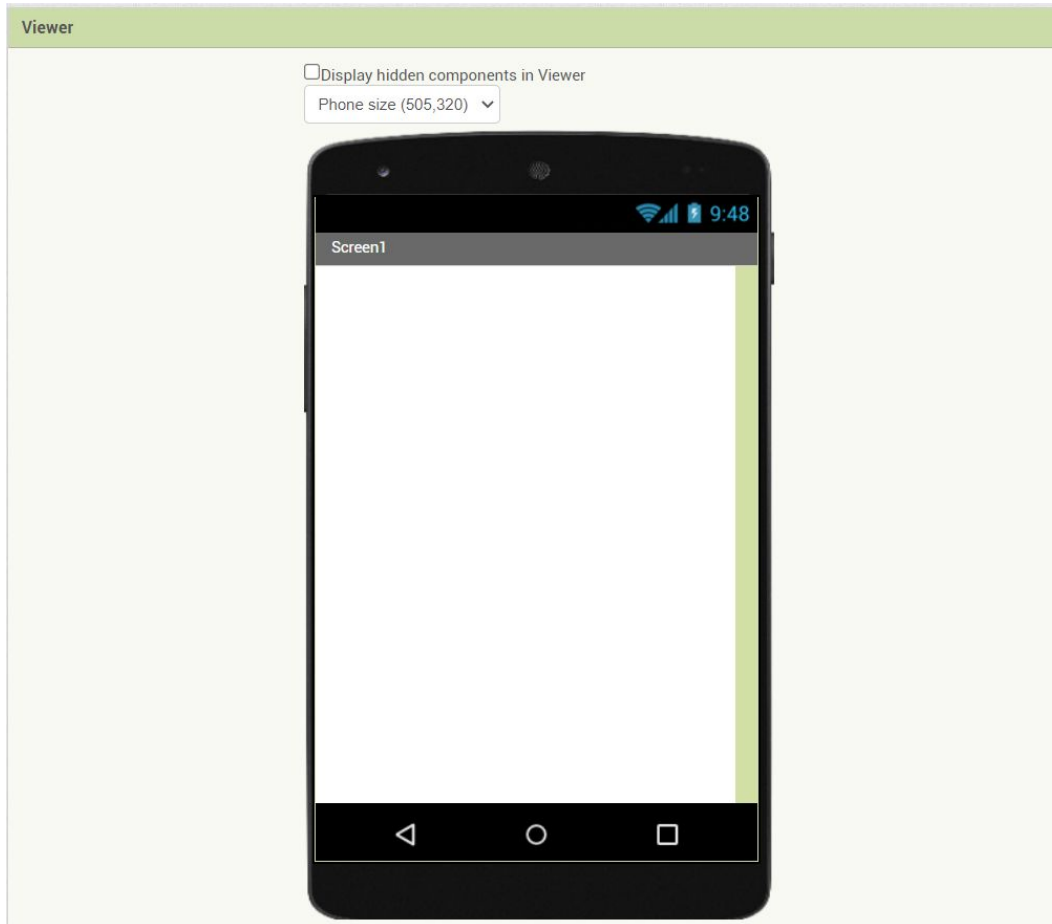
Application 1: TRANSLATOR

Palets:

- The left side of the screen features the Palette , which, as the name implies, is the space for all the creation tools.



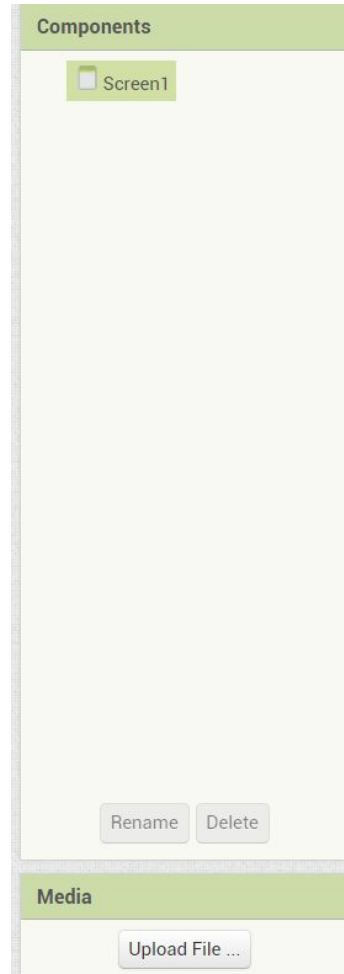
ROBOTIKBİLİM



Application 1: TRANSLATOR

Viewer:

- App Inventor is the part that shows how the Mobile Application will appear on the user's screen.
- In the upper part, the size of the smartphone or tablet on which we will install the application is determined..



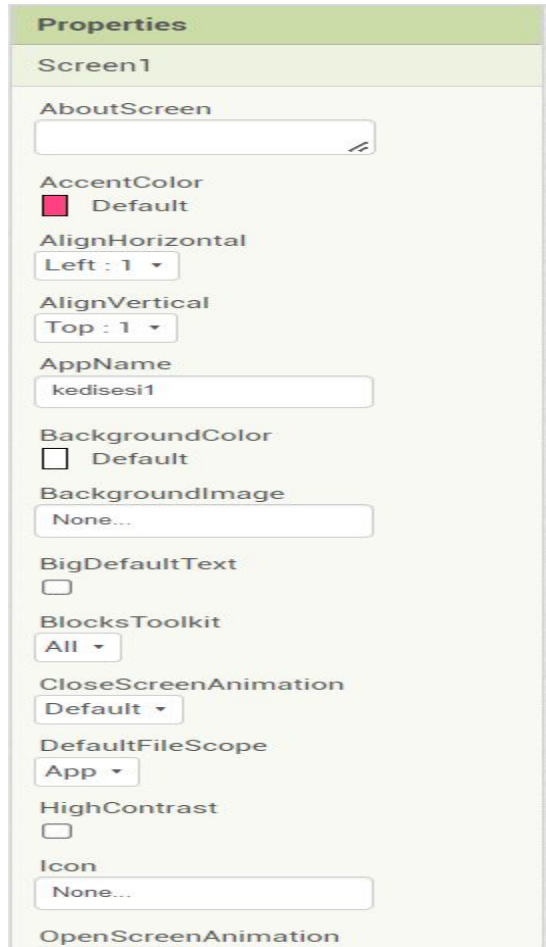
Application 1: TRANSLATOR

Components:

- In this section, we see the components that we will use in App Inventor.
- We can upload files from the media section.
- In the picture on the right you can see that the screen widget has been added.



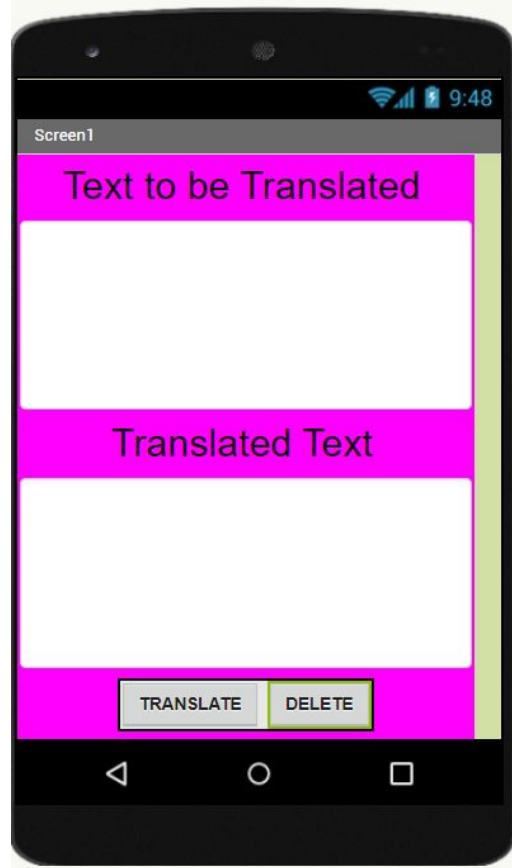
ROBOTIKBİLİM



Application 1: TRANSLATOR

Properties:

- We can set the properties of the components we use in the properties section in App Inventor.
- In this section, we can assign the media files that we have uploaded to the component.

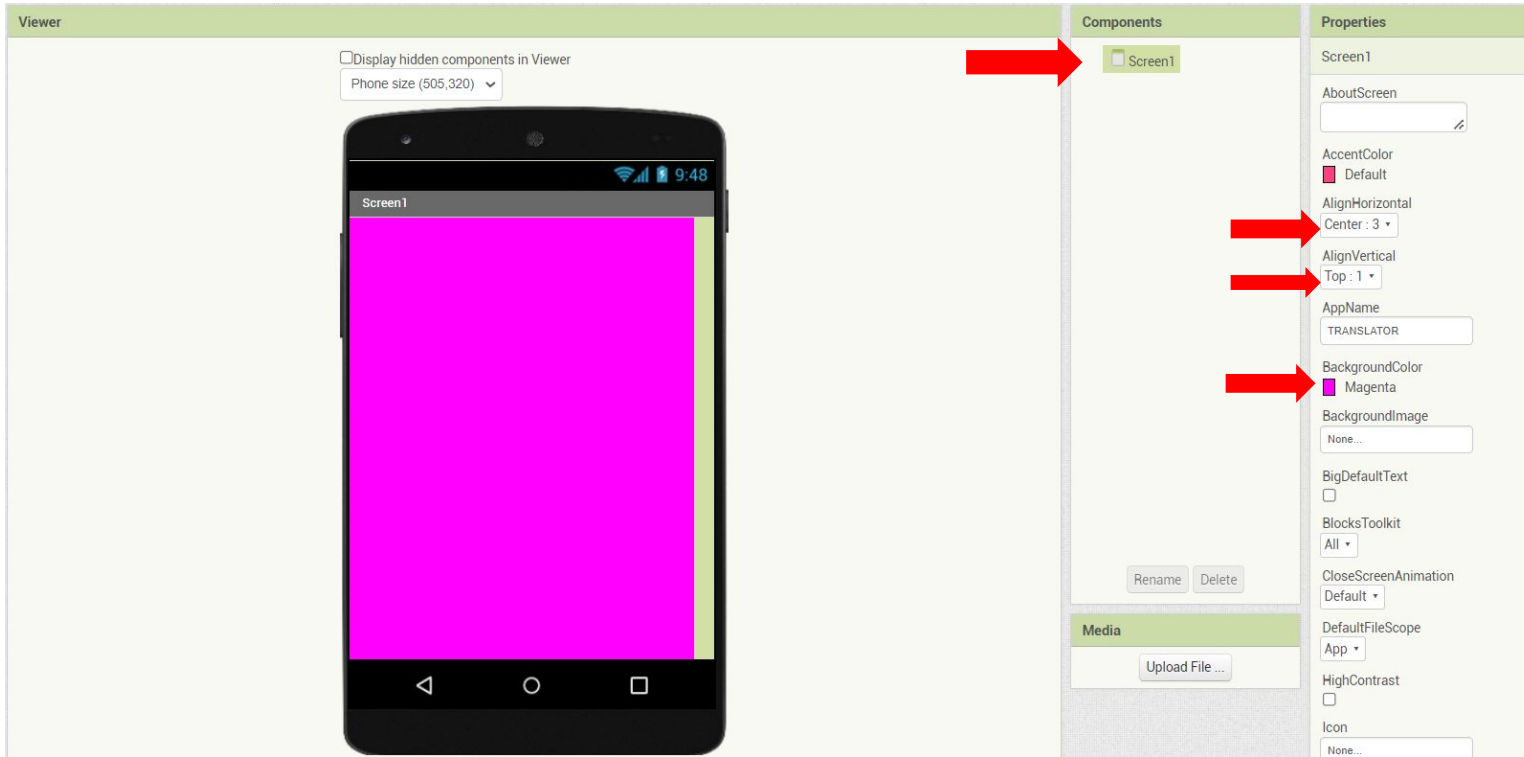


Application 1: TRANSLATOR

1-) We will make a translation application using App Inventor. We will make an application that translates the English text entered in the application into Turkish.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)Screen1 Properties:

- Let's choose "center" for the AlignHorizontal property.
- Let's select the AlignVertical property as "Top 1".
- Let's choose the backgroundColor color magenta.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-) DRAG LABEL:

Let's drag and drop the label component to the screen.





ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left, the 'Viewer' pane displays a mobile phone screen with a pink background and a text box containing 'Text to be Translated'. On the right, the 'Components' pane shows a 'Label1' component selected and circled in red. A blue arrow points from the 'Label1' component to the 'Properties' pane. The 'Properties' pane shows various settings for the label, including 'FontSize' set to 30 and 'Text' set to 'Text to be Translated'. Another blue arrow points from the 'Text' field to the 'Text' property value.

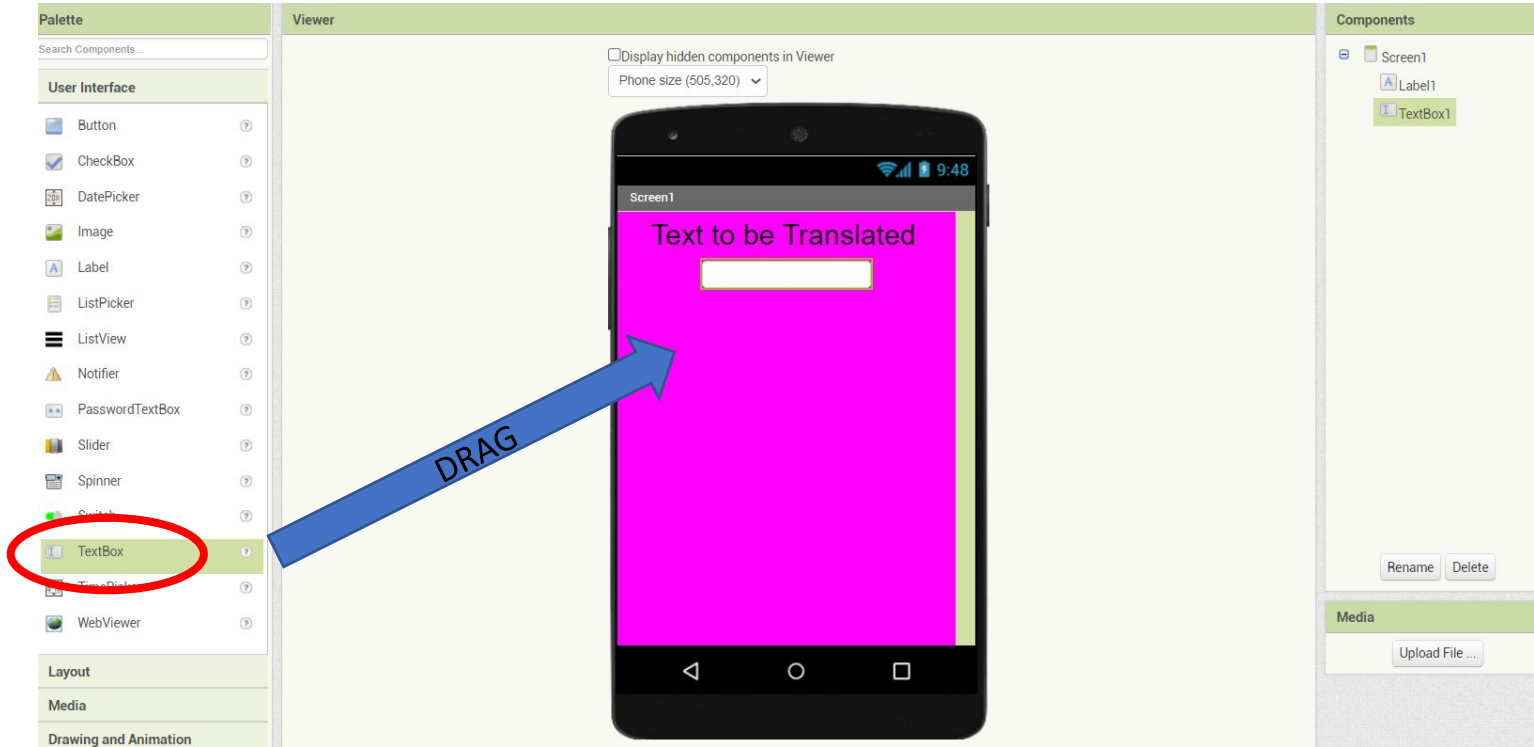
Application 1: TRANSLATOR

1-)PROPERTIES LABEL:

- Change FontSize to "30"
- Let's write the "Text to be Translated" in the Text section.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)DRAG TEXTBOX:

Let's drag and drop the
TextBox component to the
screen.



ROBOTIKBİLİM



The screenshot displays the App Inventor interface. On the left, the 'Viewer' pane shows a mobile app preview with a pink header containing the text 'Text to be Translated' and a large white text area below it. The top of the viewer has a checkbox for 'Display hidden components in Viewer' and a dropdown for 'Phone size (505,320)'. On the right, the 'Components' pane shows a tree view with 'Screen1' containing 'Label1' and 'TextBox1'. Below this are 'Rename' and 'Delete' buttons. The 'Properties' pane shows the settings for 'TextBox1', including 'BackgroundColor' (Default), 'Enabled' (checked), 'FontBold' (unchecked), 'FontItalic' (unchecked), 'FontSize' (14.0), 'FontTypeface' (default), 'Height' (Fill parent...), 'Width' (Fill parent...), 'Hint' (empty), 'MultiLine' (unchecked), 'NumbersOnly' (unchecked), 'ReadOnly' (unchecked), and 'Text' (empty). Three blue arrows point from the 'Height', 'Width', and 'Hint' fields in the Properties pane towards the mobile app preview.

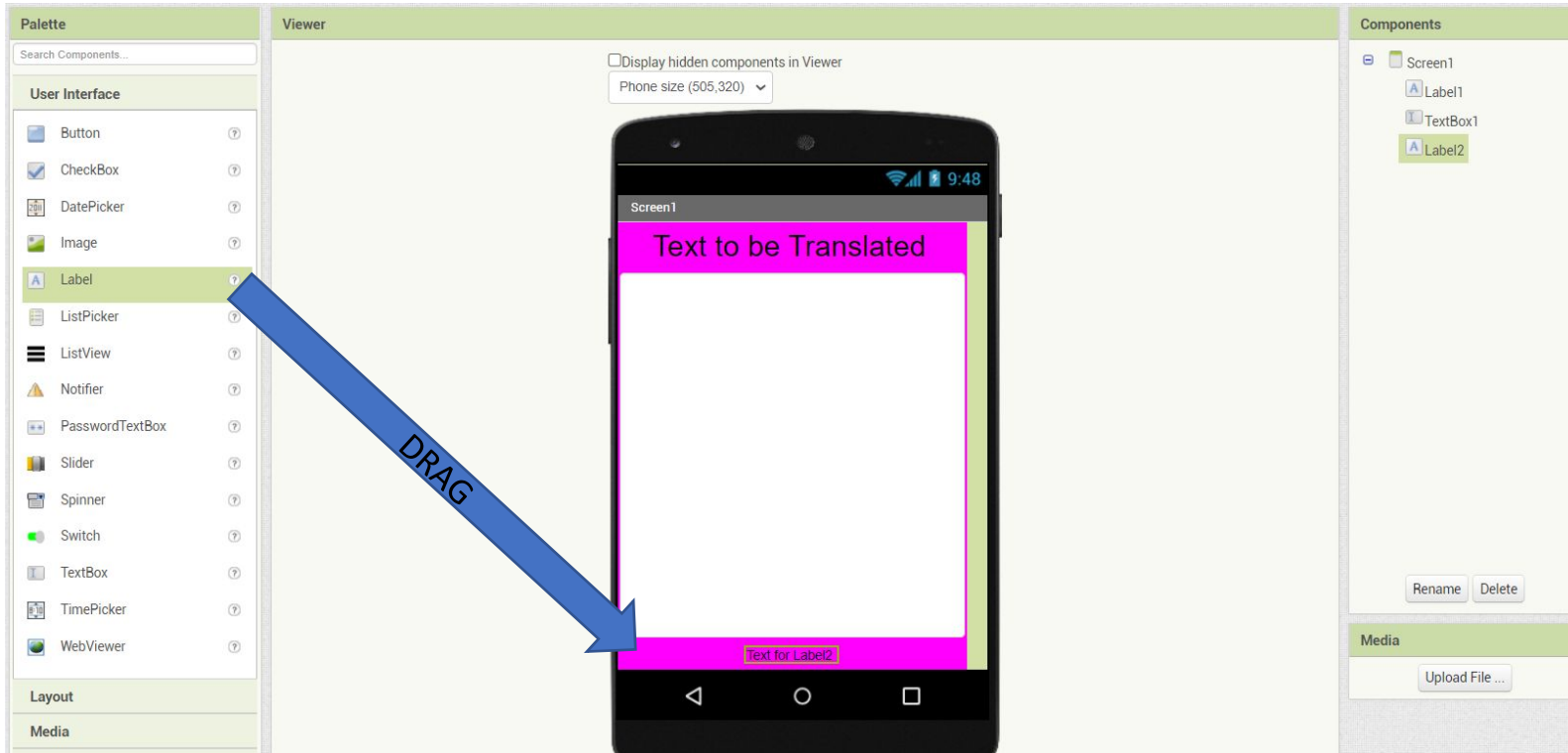
Application 1: TRANSLATOR

1-)PROPERTIES TEXTBOX:

- Let's choose height to fill parent.
- Select the width to fill parent.
- Let's delete the text in the Hint menu.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)DRAG LABEL:

Let's drag and drop the label 2 component to the screen.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left is the 'Viewer' pane showing a mobile app preview with a pink header 'Text to be Translated' and a pink footer 'Translated Text'. On the right are the 'Components' and 'Properties' panes. In the 'Components' pane, 'Label2' is circled in red. A blue arrow points from 'Label2' in the Components pane to the 'Properties' pane. In the 'Properties' pane, the 'FontSize' is set to '30' and the 'Text' is 'Translated Text'. Other properties like 'BackgroundColor', 'FontBold', 'FontItalic', 'FontTypeface', 'HTMLFormat', 'HasMargins', 'Height', 'Width', 'TextAlignment', 'TextColor', and 'Visible' are also visible.

Application 1: TRANSLATOR

1-)PROPERTIES LABEL:

- Change FontSize to “30”
- Let's write the “Translated Text” in the Text section.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface with three main panels: Palette, Viewer, and Components. The Palette on the left lists various UI components under 'User Interface', with 'TextBox' highlighted in a red circle. A blue arrow labeled 'DRAG' points from the 'TextBox' component to the Viewer. The Viewer shows a mobile device screen with a pink header 'Text to be Translated', a large white text area, and a pink footer 'Translated Text' containing a white text box. The Components panel on the right shows a list of components for 'Screen1': Label1, TextBox1, Label2, and TextBox2. The 'Media' panel at the bottom has an 'Upload File ...' button.

Application 1: TRANSLATOR

1-)DRAG TEXTBOX 2:

Let's drag and drop the
TextBox component to the
screen.



ROBOTIK BİLİM



The screenshot shows the App Inventor interface. On the left is the 'Viewer' pane showing a mobile app preview with a pink header 'Text to be Translated', a white text input area, a pink footer 'Translated Text', and another white text input area. On the right are the 'Components' and 'Properties' panes. The 'Components' pane shows a tree view with 'Screen1' containing 'Label1', 'TextBox1', 'Label2', and 'TextBox2'. The 'Properties' pane shows the settings for 'TextBox2', with blue arrows pointing to the 'Height', 'Width', and 'Hint' fields. The 'Height' and 'Width' fields are set to 'Fill parent...'. The 'Hint' field is empty.

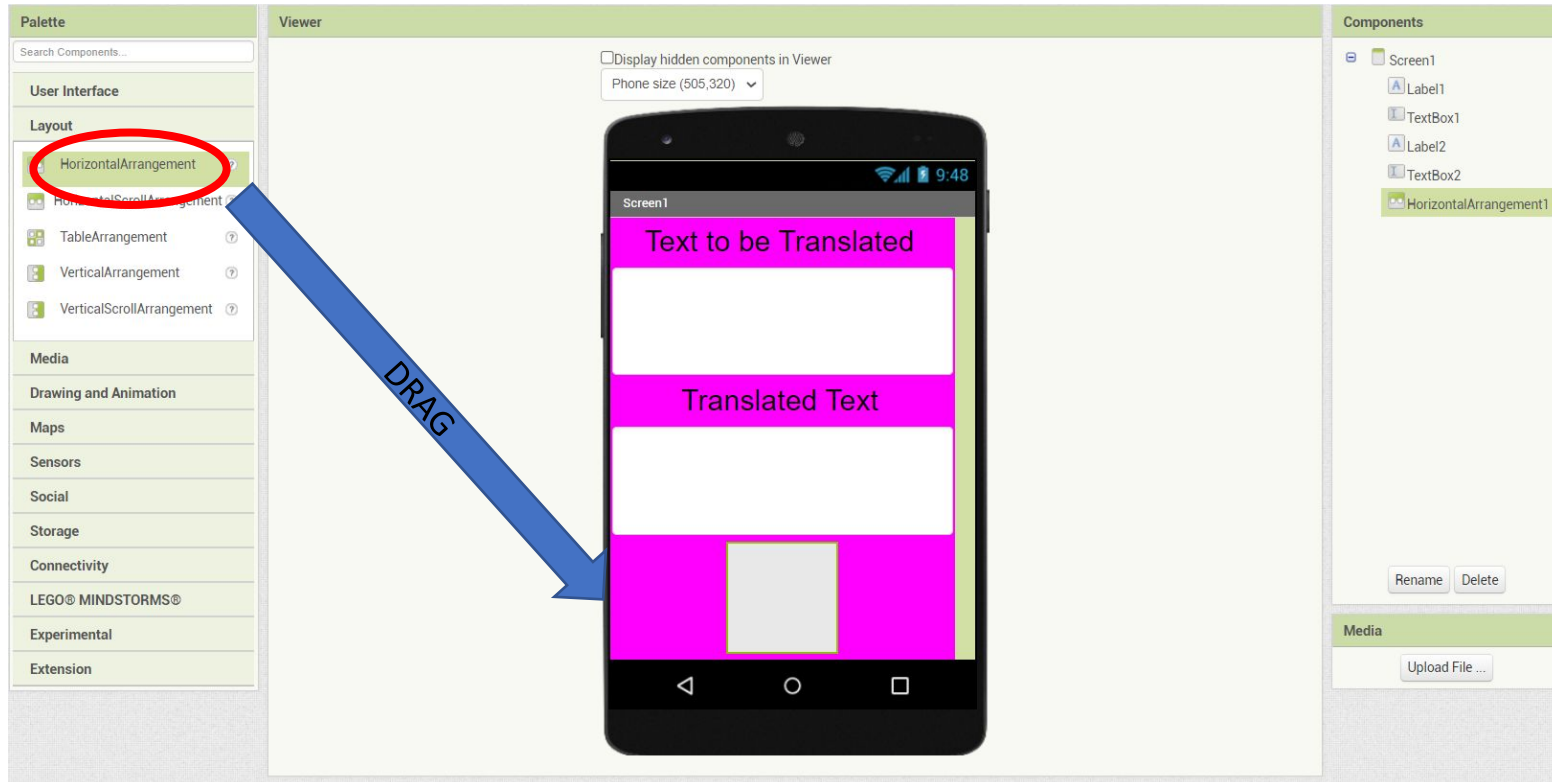
Application 1: TRANSLATOR

1-)PROPERTIES TEXTBOX 2:

- Let's choose height to fill parent.
- Select the width to fill parent.
- Let's delete the text in the Hint menu.



ROBOTIKBİLİM



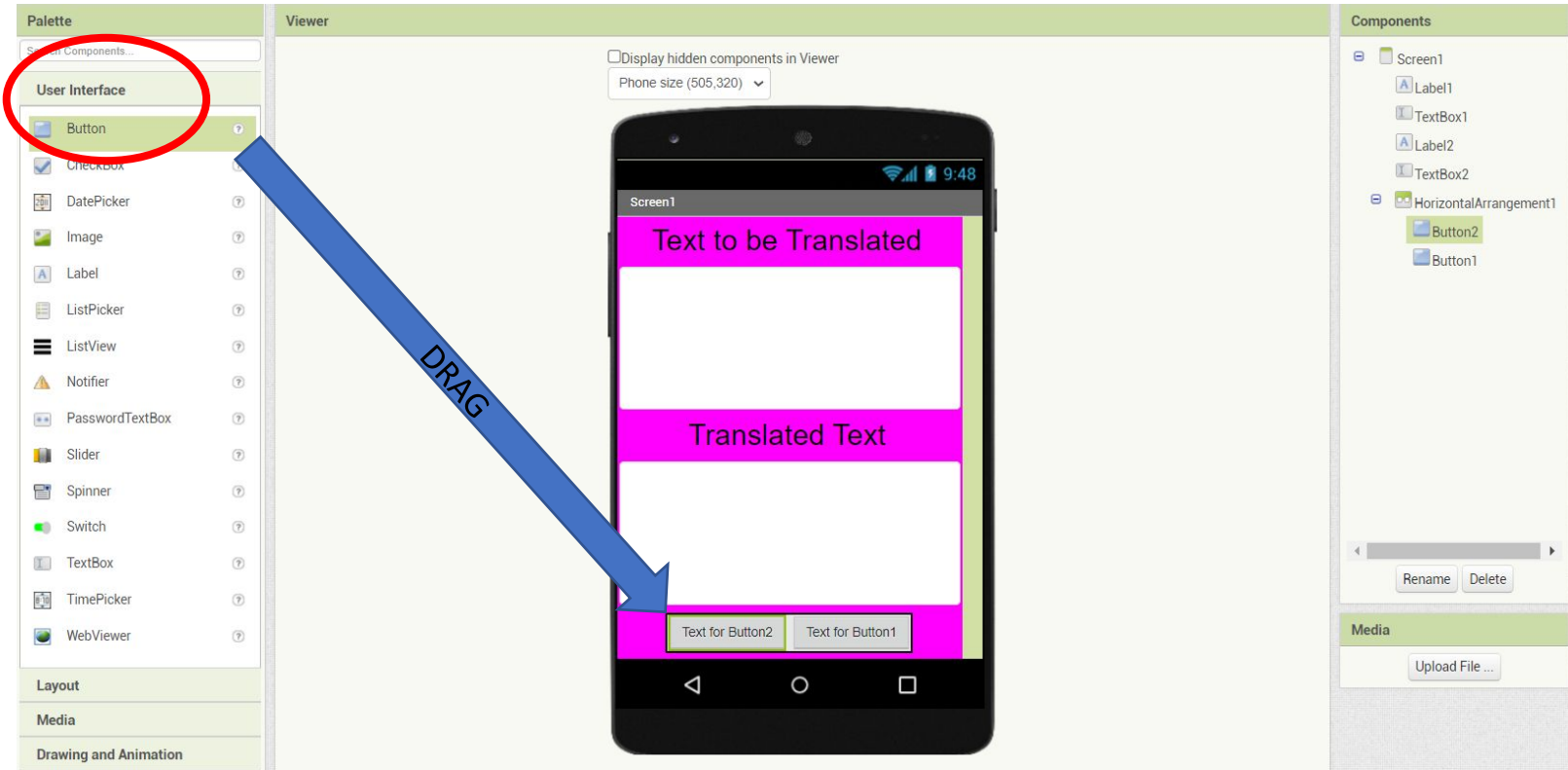
Application 1: TRANSLATOR

1-)DRAG HORIZONTAL ARRANGEMENT:

Let's drag and drop the HorizontalArrangement component to the screen.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)DRAG BUTTON COMPONENTS:

Let's drag and drop the **TWO BUTTON** component into the **HorizontalArrangement** component.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)PROPERTIES BUTTON COMPONENT:

- Select the FONT to FontBold.
- Let's write the "Translate" in the Text section.

The screenshot displays the App Inventor interface. On the left, the 'Viewer' pane shows a mobile app preview with a pink header 'Text to be Translated', a white text input field, a pink header 'Translated Text', another white text input field, and two buttons labeled 'TRANSLATE' and 'DELETE' at the bottom. On the right, the 'Components' pane shows a tree view of the app's components, with 'Button2' selected. A blue arrow points from 'Button2' in the Components pane to the 'Properties' pane. The 'Properties' pane shows the following settings for 'Button2':

- BackgroundColor: Default
- Enabled:
- FontBold:
- FontItalic:
- FontSize: 14
- FontTypeface: default
- Height: Automatic...
- Width: Automatic...
- Image: None...
- Shape: default
- ShowFeedback:
- Text: TRANSLATE
- TextAlignment: center : 1

Below the Components pane, the 'Media' section shows an 'Upload File ...' button with a blue arrow pointing to it.



ROBOTIKBİLİM



Application 1: TRANSLATOR

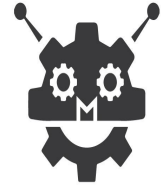
1-) PROPERTIES BUTTON COMPONENT:

- Select the FONT to FontBold.
- Let's write the "Delete" in the Text section.

The screenshot displays the App Inventor interface. On the left, the 'Viewer' shows a mobile app preview with a pink header 'Text to be Translated', a white text input field, a pink footer 'Translated Text', and two buttons labeled 'TRANSLATE' and 'DELETE'. On the right, the 'Components' panel shows a tree view with 'Screen1' containing 'Label1', 'TextBox1', 'Label2', 'TextBox2', 'HorizontalArrangement1', 'Button2', and 'Button1'. A blue arrow points from 'Button1' in the Components panel to the 'Properties' panel. The 'Properties' panel for 'Button1' shows various settings: 'BackgroundColor' is 'Default', 'Enabled' is checked, 'FontBold' is checked, 'FontItalic' is unchecked, 'FontSize' is '14.0', 'FontTypeface' is 'default', 'Height' is 'Automatic...', 'Width' is 'Automatic...', 'Image' is 'None...', 'Shape' is 'default', 'ShowFeedback' is checked, 'Text' is 'DELETE', and 'TextAlignment' is 'center : 1'. A second blue arrow points from the 'Media' panel's 'Upload File...' button to the 'Text' property field.



ROBOTIKBİLİM



JUNIOR MAKER LAND

The screenshot shows the App Inventor interface with three main panels: Palette, Viewer, and Components. The Palette panel on the left has a search bar and several categories. The 'Media' category is expanded, and the 'YandexTranslate' component is circled in red. A blue arrow labeled 'DRAG' points from this component to the Viewer panel. The Viewer panel shows a mobile phone screen with a pink background and two text input fields labeled 'Text to be Translated' and 'Translated Text'. At the bottom of the screen are two buttons labeled 'TRANSLATE' and 'DELETE'. The Components panel on the right shows a tree view of the application's components, including 'Screen1', 'Label1', 'TextBox1', 'Label2', 'TextBox2', 'HorizontalArrangement1', 'Button2', 'Button1', and 'YandexTranslate1'. The 'YandexTranslate1' component is highlighted in green. Below the Viewer panel, there is a 'Non-visible components' section with a 'YandexTranslate1' component.

Application 1: TRANSLATOR

1-)DRAG YANDEX TRANSLATE COMPONENT:

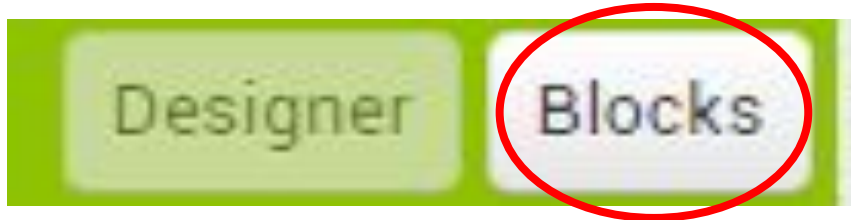
Let's drag the Yandex translate component from the media menu and drop it on the screen.



Application 1: TRANSLATOR

1-)PROGRAMMING
TRANSLATOR:

Select the **BLOCKS** menu at
the top right of the screen.





ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left, the 'Blocks' panel lists various categories like Logic, Math, Text, Lists, Dictionaries, Colors, Variables, and Procedures. Under the 'Screen1' category, there are several UI components: Label1, TextBox1, Label2, TextBox2, HorizontalArrangement1, and a 'TRANSLATE' block which is circled in red. Below the 'TRANSLATE' block is a 'YandexTranslate1' component. On the right, the 'Viewer' window shows a list of event-driven blocks for the 'TRANSLATE' component. The first block, 'when TRANSLATE .Click', is circled in red. A red arrow points from this block to a separate 'when TRANSLATE .Click' block shown to the right of the viewer window.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- First let's code the translate button.
- When you click on Translate button, the text in the text to be translated into the TextBox is translated into Turkish with the YandexTranslate tool.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- First let's code the translate button.
- When you click on Translate button, the text in the text to be translated into the TextBox is translated into Turkish with the YandexTranslate tool.

The screenshot shows the App Inventor interface. On the left, the 'Blocks' panel lists various components, with 'YandexTranslate1' circled in red. In the center, the 'Viewer' shows the code for the 'TRANSLATE' button. The code consists of a 'when TRANSLATE.Click' event with a 'do' block containing a 'call YandexTranslate1.RequestTranslation' block. A red circle highlights the 'call YandexTranslate1.RequestTranslation' block in the 'Viewer', and a red arrow points to the 'call YandexTranslate1.RequestTranslation' block in the 'Blocks' panel.



ROBOTIKBilim



The screenshot shows the App Inventor interface. On the left, the 'Blocks' panel is visible, with the 'Text' category selected. On the right, the 'Viewer' panel shows a script starting with 'when TRANSLATE.Click' followed by 'do call YandexTranslate1.RequestTranslate'. The 'RequestTranslate' block has two input fields: 'languageToTranslateTo' and 'textToTranslate'. The 'textToTranslate' field contains the value 'tr'. A red circle highlights the 'Text' block in the 'Blocks' panel, and a red arrow points from it to the 'tr' value in the 'RequestTranslate' block.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- Let's write to "tr" into the string block.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left is the 'Blocks' palette with categories like Control, Logic, Math, Text, Lists, Dictionaries, Colors, Variables, and Procedures. Under 'Text', 'TextBox1' is highlighted with a red circle. On the right is the 'Viewer' showing a list of blocks for 'TextBox1' including Hint, MultiLine, NumbersOnly, ReadOnly, and Text. The 'Text' block is also highlighted with a red circle. A red arrow points from this 'Text' block to the 'TRANSLATE' button's click event in the Viewer, which contains a 'call YandexTranslate1 .RequestTranslation' block with 'languageToTranslateTo' set to 'tr' and 'textToTranslate' set to 'TextBox1 . Text'.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- Let's drag the Translate Text code from the translate button and drop it end of yandex translate code.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left, the 'Blocks' palette contains various categories like Math, Text, Lists, etc. A red circle highlights the 'when YandexTranslate1 . GotTranslation' block in the 'Text' category. A red arrow points from this block to the 'Viewer' screen. On the 'Viewer' screen, the code blocks are arranged as follows: a 'when YandexTranslate1 . GotTranslation' block (circled in red), followed by a 'call YandexTranslate1 . RequestTranslation' block, a 'set YandexTranslate1 . ApiKey to' block, and a 'YandexTranslate1' block. The 'TRANSLATE' block is also visible in the 'Blocks' palette.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- After the YandexTranslate tool finishes the translation process, the result is thrown into the TextBox translation. Thus, this text is displayed in the TextBox.
- Let's drag the GetTranslation code from the YandexTranslate1 and drop it on the code screen.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left, the 'Blocks' palette is visible, with the 'YandexTranslate1' component selected. On the right, the 'Code Viewer' shows the following code:

```
when TRANSLATE .Click
do
  call YandexTranslate1 .RequestTranslation
    languageToTranslateTo "tr"
    textToTranslate TextBox1 .Text

when YandexTranslate1 .GotTranslation
  responseCode translation
do
  set TextBox2 .Text to
```

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- After the YandexTranslate tool finishes the translation process, the result is thrown into the TextBox translation. Thus, this text is displayed in the TextBox.
- Let's drag the GetTranslation code from the YandexTranslate1 and drop it on the code screen.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left, the 'Blocks' palette is visible, with the 'Variables' category highlighted by a red circle. A red arrow points from this category to a 'get' block in the Viewer. The Viewer shows a script for a translator application. The script starts with an 'initialize global name to' block, followed by a 'get' block. Below this, there are two 'initialize local name to' blocks. The main logic is in a 'when TRANSLATE .Click' event handler, which calls 'YandexTranslate1 .RequestTranslation' with 'languageToTranslateTo' set to 'tr' and 'textToTranslate' set to 'TextBox1 .Text'. Below this, there is a 'when YandexTranslate1 .RequestTranslation' event handler that sets 'TextBox2 .Text' to the result of a 'get translation' block.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- Let's drag the “get” code from the variables and drop it end of the text code.



ROBOTIKBilim



when DELETE Click
do

when DELETE GotFocus
do

when DELETE LongClick
do

when DELETE LostFocus
do

when DELETE TouchDown
do

when DELETE TouchUp
do

DELETE BackgroundColor

set DELETE BackgroundColor to

DELETE Enabled

set DELETE Enabled to

when TRANSLATE Click
do
call YandexTranslate1 .RequestTranslation
languageToTranslateTo "tr"
textToTranslate TextBox1 . Text

when YandexTranslate1 GotTranslation
responseCode translation
do set TextBox2 . Text to get translation

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- When you click Delete Button, the text to be translated in the TextBox and the text in the Translation of the TextBox are emptied and cleared.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- When you click Delete Button, the text to be translated in the TextBox and the text in the Translation of the TextBox are emptied and cleared.

The screenshot displays the App Inventor interface. On the left, the 'Blocks' palette is visible, with the 'DELETE' block under the 'Any component' category circled in red. A red arrow points from this block to the 'when DELETE Click' block in the 'Viewer' area. This block contains a 'set TextBox1 . Text to' block with a red 'X' icon, indicating that the text in the TextBox is being cleared. Other blocks in the Viewer include 'when TRANSLATE Click' (calling 'YandexTranslate1 . RequestTranslation'), 'when YandexTranslate1 Got Translation' (setting 'TextBox2 . Text' to 'get translation'), and various 'set' blocks for 'TextBox1' properties like 'MultiLine', 'NumbersOnly', 'ReadOnly', 'Text', 'TextColor', 'Visible', 'Width', and 'WidthPercent'.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left is the 'Blocks' palette with categories like Logic, Math, Text, Lists, Dictionaries, Colors, Variables, and Procedures. The 'Text' category is circled in red. On the right is the 'Viewer' showing a 'Join' block, also circled in red. A red arrow points from the 'Join' block to the 'RequestTranslation' block in the code snippet below.

```
when TRANSLATE Click
do
  call YandexTranslate1 .RequestTranslation
    languageToTranslate "tr"
    textToTranslate TextBox1 .Text
```

```
when YandexTranslate1 .GotTranslation
  response Code translation
do
  set TextBox2 .Text to get translation
```

```
when DELETE Click
do
  set TextBox1 .Text to ""
```

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- When you click Delete Button, the text to be translated in the TextBox and the text in the Translation of the TextBox are emptied and cleared.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- When you click Delete Button, the text to be translated in the TextBox and the text in the Translation of the TextBox are emptied and cleared.

The screenshot displays the App Inventor interface. On the left is the 'Blocks' palette, and on the right is the 'Viewer' showing the visual representation of the code blocks. A red circle highlights the 'set TextBox2 . Text to' block in the Blocks palette, with a red arrow pointing to the same block in the Viewer. The Viewer shows a sequence of blocks for TextBox2: MultiLine, NumbersOnly, ReadOnly, Text, TextColor, Visible, Width, and WidthPercent. A 'when TRANSLATE .Click' block calls YandexTranslate1 .RequestTranslation with languageToTranslateTo set to 'tr' and textToTranslate set to TextBox1 .Text. A 'when YandexTranslate1 .GotTranslation' block sets TextBox2 .Text to get translation. A 'when DELETE .Click' block sets TextBox1 .Text to '' and sets TextBox2 .Text to ''.



ROBOTIKBİLİM



The screenshot shows the App Inventor interface. On the left is the 'Blocks' palette with various categories like Lists, Variables, and Procedures. The 'Text' block is highlighted with a red circle. On the right is the 'Viewer' showing a visual representation of the app. A red circle highlights a 'Text' block in the Viewer. A red arrow points from this 'Text' block to the 'DELETE' event handler code block in the code view.

Application 1: TRANSLATOR

1-)PROGRAMMING TRANSLATOR:

- When you click Delete Button, the text to be translated in the TextBox and the text in the Translation of the TextBox are emptied and cleared.



ROBOTIKBilim



The screenshot shows the MIT App Inventor interface. The 'Connect' menu is open, and the 'AI Companion' option is highlighted with a red circle and a red arrow. The interface also shows a 'TRANSLATOR' screen with various code blocks in the 'Viewer' pane and a 'Blocks' pane on the left.

Application 1: TRANSLATOR

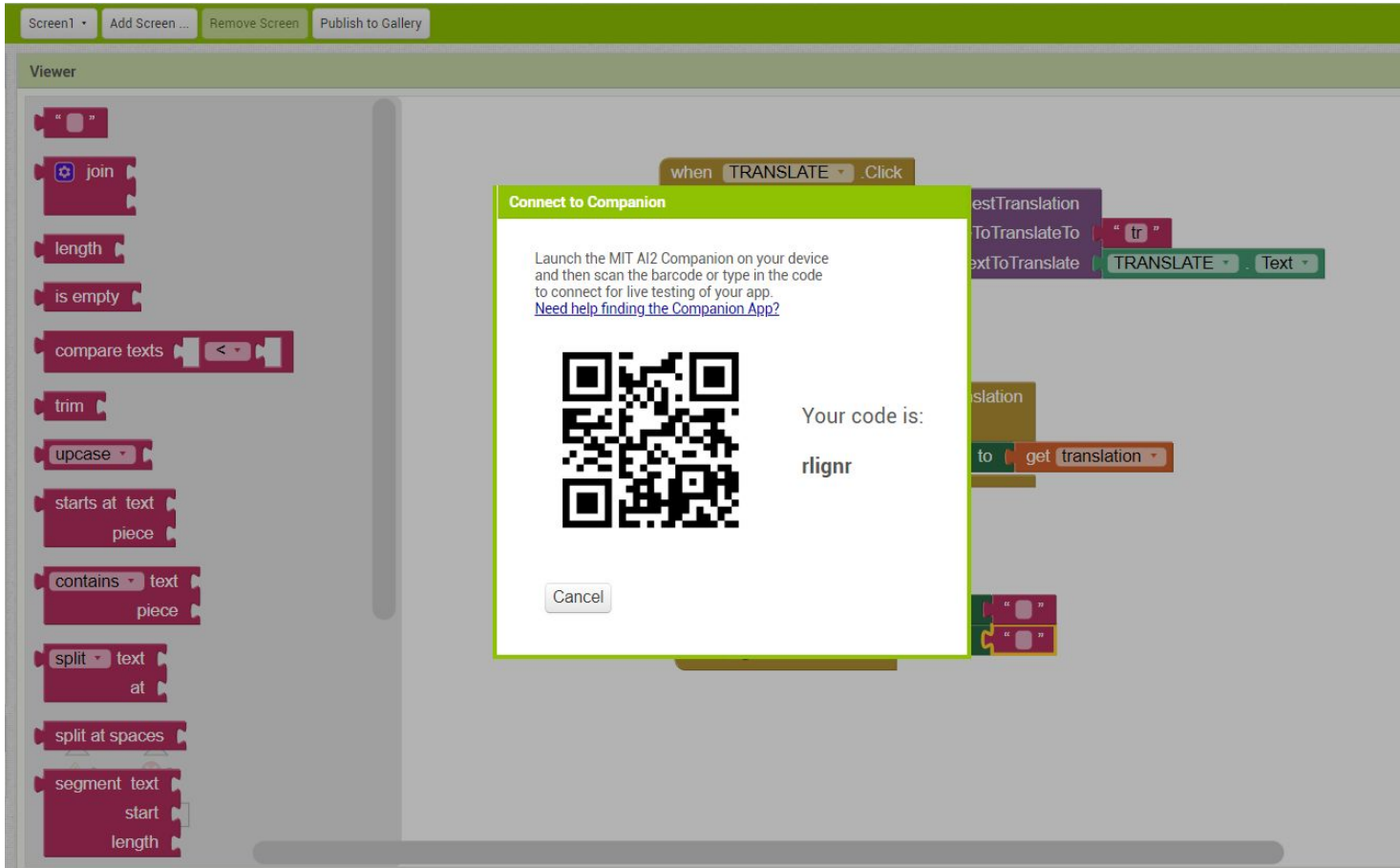
1-)LOADING TRANSLATOR:

Let's choose the connect menu from the top menu of the screen.

Then let's select the AI companion menu.



ROBOTIKBİLİM



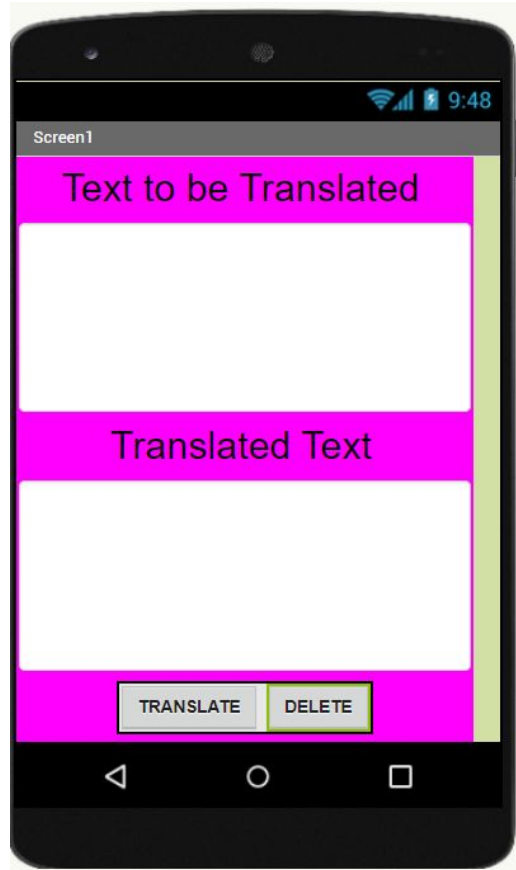
Application 1: TRANSLATOR

1-)LOADING TRANSLATOR:

Let's scan the QR code that appears on the screen to the application we have downloaded on our phone.



ROBOTIKBİLİM



Application 1: TRANSLATOR

1-)LOADING TRANSLATOR:

Translator is ready...



THANK YOU