

Make This

Chapter 12 - Using the Sharing Component

Sharing pictures or messages from within apps is very common. In this tutorial, you construct an app that facilitates sharing messages or pictures from your device.

This tutorial teaches the following skills:

- Using the Sharing Component
- Using the Camera Component

Note: Before attempting this exercise, complete the Chapter 2 and 3 exercises to familiarize yourself with the App Inventor interface, getting your Android device connected, and starting a new project.

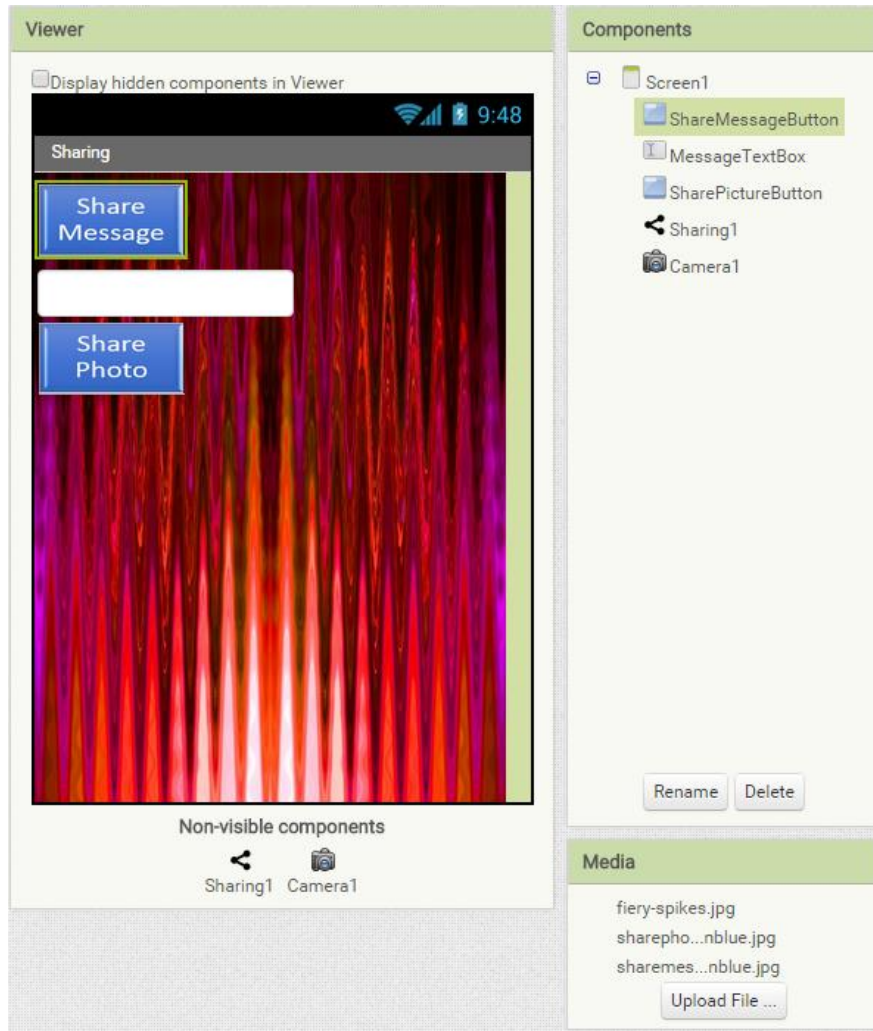
Building the Sharing App

1. Navigate to <http://appinventor.mit.edu/explore/>. If necessary, sign in with your Google Account.
2. Start a new project named *SharingApp*. Change the **Title** of **Screen1** to *Sharing*.
3. Connect App Inventor to your Android device.

Part 1 - Constructing the Interface

The interface in this example is simple: the user either enters a message in a text box then clicks a button to share it, or clicks a button to launch the camera app to take and share a picture.

Your layout for your interface should look like this:



Build the interface shown above by dragging out the components shown in the following table:

Component	Palette Group	Component Name	Function
Button	User Interface	ShareMessageButton	To launch a messaging-capable app to share your message
TextBox	User Interface	MessageTextBox	Box for entry of user message
Button	User Interface	SharePictureButton	To launch the camera to take a picture to be shared
Sharing	Social	Sharing1	Non-visible component that facilitates launching a sharing app
Camera	Media	Camera1	Non-visible component that allows you to access the device's camera

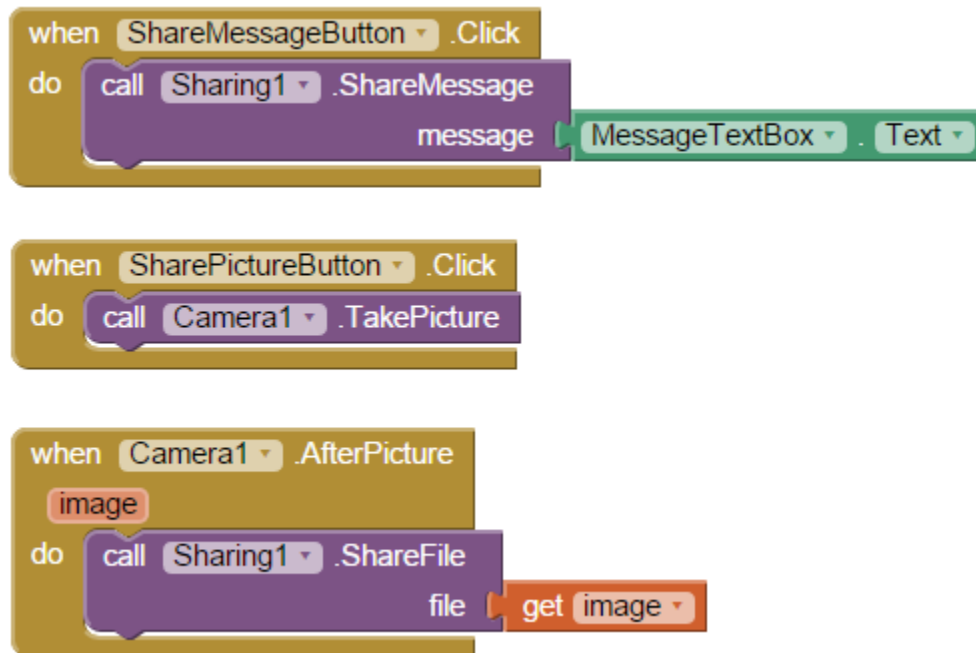
Set the properties of each component in the following ways:

- Buttons can contain a picture as a background. Change the **Image** for **ShareMessageButton** to *sharemessagebuttonblue.jpg* (upload the file provided). Change the **Width** of the button to *100 pixels* and change the **Height** to *50 pixels*. Delete the default **Text** for the button.
- Set the **Hint** of **MessageTextBox** to *Enter message here*.
- Change the **Image** for **SharePictureButton** to *sharephotobuttonblue.jpg* (upload the file provided). Change the **Width** of the button to *100 pixels* and change the **Height** to *50 pixels*. Delete the default **Text** for the button.
- Change the **BackgroundImage** of **Screen1** to *fiery-spikes.jpg* (upload the file provided) or a file of your own preference. Change the **BackgroundColor** of **Screen1** to *None*.

Part 2 - Program Functionality of the App

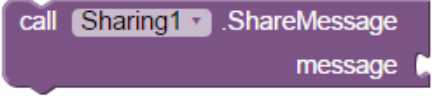
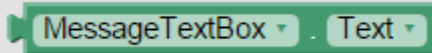
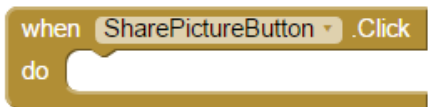
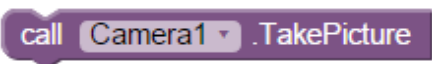
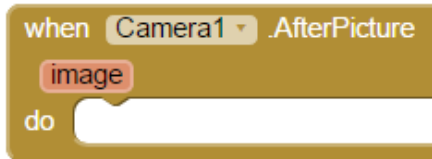

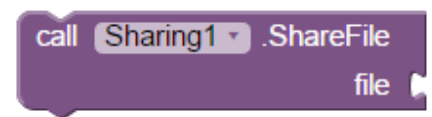


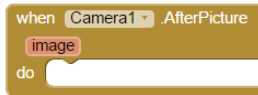
There are many apps installed on devices that are capable of sending messages or pictures. The Sharing Component in App Inventor brings up a list of available apps to facilitate choosing an appropriate one.



1. Change to **Blocks** view.

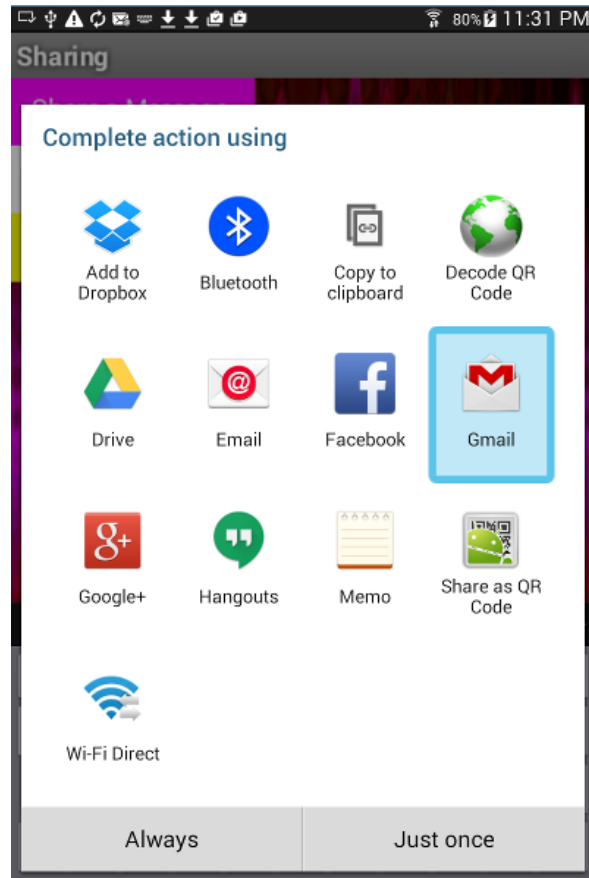


2. Now drag out the blocks (as indicated in the table below) and arrange them as shown above.

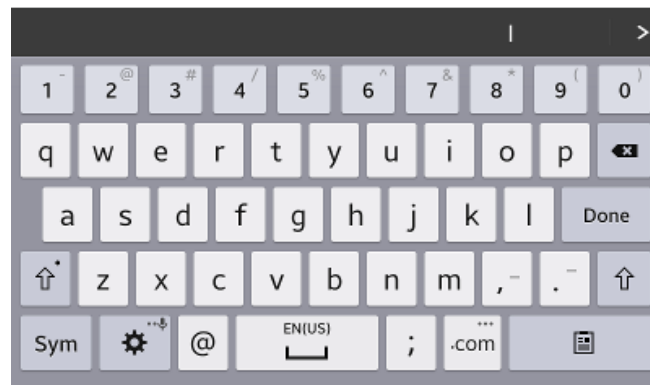
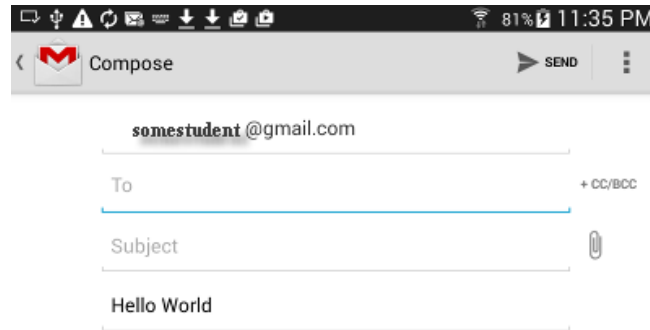
Block	Drawer	Function
	ShareMessageButton	Controls what happens when ShareMessageButton is clicked

	Sharing1	Finds available sharing apps on the device and passes a message to the app chosen.
	MessageTextBox	Returns the value entered by the user as their message
	SharePictureButton	Controls what happens when SharePictureButton is clicked
	Camera1	Activates device's camera to take a picture
	Camera1	Controls what happens after the picture is taken with the camera. Has the argument  that provides access to the picture just taken.
	Sharing1	Finds available sharing apps on the device and passes a file to the app chosen.
	Obtained by pointing to the  argument in  the block	Defines the file that will be passed to the sharing app chosen.

Now test the functionality of the  button by entering the message *Hello World* into the **MessageTextBox** and pressing the  button. Your device display should look something like this:



The Sharing component brings up a list of the apps installed on the device capable of sharing the selected media (in this case, the text string *Hello World*). You should select an application then, at the bottom of the screen, touch *Always* or *Just once* to choose whether this type of data is always shared by the app chosen or just in this particular instance. If you choose Gmail, your Gmail app will open and look similar to the following:

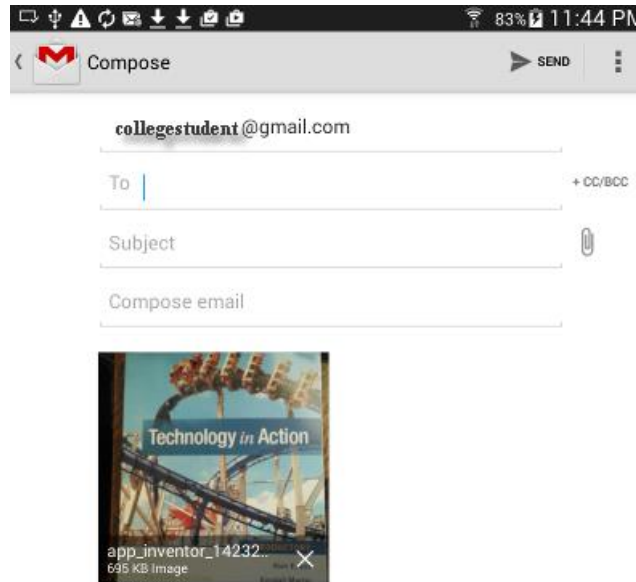


Notice that the message entered in the text box in your app (Hello World) has been transferred to the body of the e-mail. All you need to do is address it, add a subject, and send it! But in this case, discard the message, which will close Gmail and return you to your app.

Alert: The Share Photo button will not produce the expected results when using the emulator.



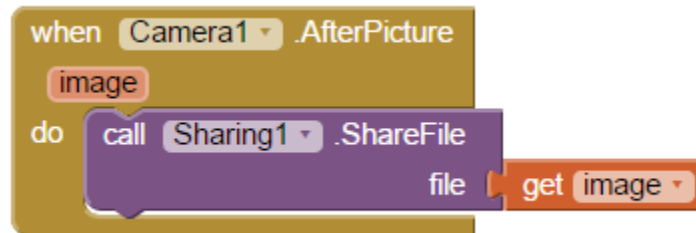
Now let's test the **Share Photo** button. After pressing this button, your device's camera interface should be visible. Take a picture and select Save. The list of available sharing apps should open. Select Gmail again and your screen should look like this:



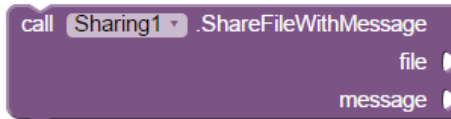
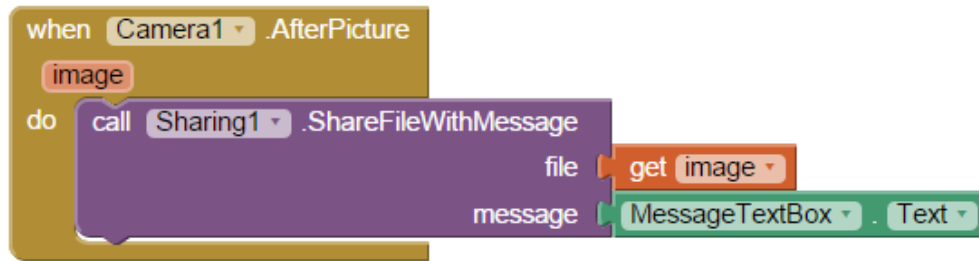
Notice that the picture you took is now attached to the e-mail. Again, discard the e-mail message and you will return to your app.

Part 4 - Share a Picture and a Message

Let's modify the blocks that control the  button and have the app share a picture and a message. You need to modify this block:




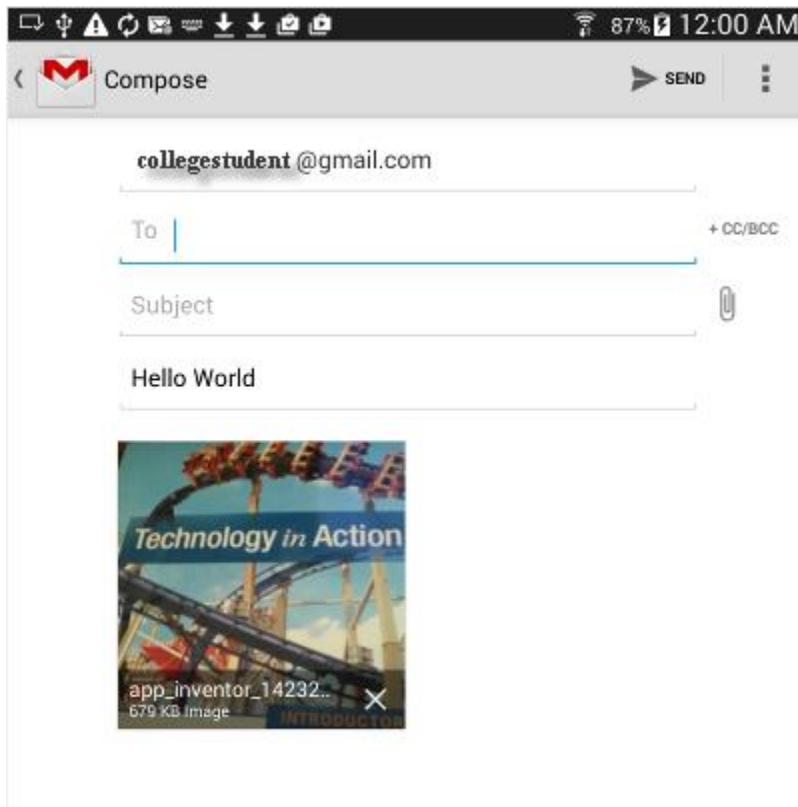
...to change it into this one:



The block is similar to the `call Sharing1 .ShareFile` block except that it will pass a file and a message to the chosen sharing app. It is found in the **Sharing1** drawer.

Alert: The Share Photo button will not produce the expected results when using the emulator.

Once you have modified the block, enter *Hello World* in the text box and test the  button again. This time, if you chose Gmail, both the picture you took and the message (Hello World) appear in the email.



Extensions to This Project

1. For sharing a file stored on your SD card, you need to reference the card as a URL...which requires knowing the file path. You need a file manager app to explore what's stored on your device. Many Android devices come with file manager apps installed (for instance Samsung devices have the My Files app). However, some devices don't have file management apps. You can obtain free file management apps (like ES File Manager or AntTek Explorer Ex) from the Google Play Store.

Once you can see what's on your device, you can reference a file name directly. For instance, say you have a folder called Music on your SD card with subfolders organized by band and then album. You can address a file on the SD card by the file path as follows: `file:///sdcard/Music/Beatles/The%20Beatles/Rocky%20Racoon.mp3`

You will need to use "URL encoding" for special characters...which includes spaces. In this case, a space is represented by "%20".

So explore your device and find a file that you want to share that is stored on your device. Determine the path to the file and the name of the file. Then modify your app to reference the file name for sharing.

Resources

- [AI2 Social Components: Sharing](#)
- [User Guide for App Inventor 2](#)
- [Guide to Understanding Blocks](#)

MIT App Inventor is a blocks-based programming tool that allows everyone, even novices, to start programming and build fully functional apps for Android devices. Google's Mark Friedman and MIT Professor Hal Abelson co-led the development of App Inventor while Hal was on sabbatical at Google. App Inventor runs as a Web service administered by staff at MIT's Center for Mobile Learning - a collaboration of MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL) and the MIT Media Lab. MIT App Inventor supports a worldwide community of nearly 3 million users representing 195 countries worldwide. App Inventor is an open-source tool that seeks to make both programming and app creation accessible to a wide range of audiences. App Inventor is the property of the Massachusetts Institute of Technology (MIT) and the work licensed under a [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). For more information on App Inventor, go to [MIT App Inventor About Us page](#).