Code 4

Using Mobile Features & Connecting to the Web

Learning Objectives:

In this unit, you will learn about...

- User interface components
- Media components
- Sensors
- Social components
- Web databases

Mobile Components

Congratulations on making it to the fourth coding unit! So far you have been exploring ways to talk to your phone and build apps. In this unit you will learn about some of the mobile features that your phone has that you can take advantage of as well as how to connect to other places on the web.

Use this unit as a reference to learn about what capabilities your app can offer. You don't need to try all of the components listed, but should experiment with the ones that make the most sense to include in your app.

For reference, here is the Technovation 2017 Judging Rubric and Technical Checklist.

User Interface Components

These are features that can be included in your app that a user can interact with. Check out MIT's website for more information.

Notifier

This component can send the user a message or an alert. The user can press a button to get rid of dialog or the dialog can disappear by itself after a few seconds. The notifier is good for sending information to the user and is also really useful for debugging, which you will learn about in Code 6. Here's a tutorial.

Canvas

This component allows users to use sprites, draw, move images. This can be very useful if you're building a game! Here are some tutorials:

- Paintpot 1
- Paintpot 2
- Paintpot 3
- Paintpot 4
- Ball MIT Tutorial
- MIT digital doodle

WebView

This component allows you to embed a web browser in the app. Use this if you want the user to be able to access a specific website. You can specify what URL you want the browser to show the user, but this browser isn't a full browser. For example, pressing the phone's hardware back key will exit the app, rather than move back in the browser history. Here is a tutorial.

Date and Time Picker

This component allows users to pick a date and/or time. This is especially useful for apps where users need to schedule events. Here is a tutorial.

Media Components

These are features that involve media such as photographs, audio, and video. Check out MIT’s website for more information.

Camcorder

This component allows the user to take videos. You can use it for social apps, video sharing apps, or anytime else that you would want your user to record a video. Learn how to use it here.

Camera

This component allows the user to take pictures. This can be useful for apps that allow users to set profile pictures or take pictures of. Here is a tutorial.

ImagePicker

This allows the user to pick an image from their photo library. It's a good idea to include this feature if you plan to use the Camcorder or Camera components. It will allow your user to pick photos that they have taken outside of using your app. Here is a tutorial.

VideoPlayer

Allows you to embed a video into the app that the user can click on. The video must be a .wmv, .3gp or .mp4 and not bigger than 1MB. Here is a tutorial.

YandexTranslate

This allows you to translate text into another language. It requires that your app have internet access since it relies on Yandex translate service. Here is a tutorial.

SoundRecorder

This allows a user to record a sound or noise. Here's an example of it in use from Pura Vida Apps.

Player

This audio component plays a sound. This works best for "long" sounds, such as songs, speeches, or poems. Here is a tutorial and the tutorial overview. **Video Player** is a similar component, but works for video rather than just audio!

Sound

This component is very similar to the Player component, but is best for short sounds, like notification "dings". Here's a tutorial to learn how to use it.

SpeechRecognizer

This translates the user's speech into text. This is especially useful for apps that require hands-free capabilities. Here is an example.

TexttoSpeech

This does the reverse of SpeechRecorder! With this feature, users can enter text and the app will read it out loud. Here's a tutorial.

Sensors

Since you're designing an app for a mobile device, you should take advantage of the **features** and **sensors** that mobile phones have. These **features** and **sensors** are unique to mobile devices and a computer might not necessarily have them. Using these features will be what makes your app a *mobile app* instead of a *web app* or a *website*. Check out all the cool sensors smartphones have!

App Inventor allows you to program these sensors! Your phone may not have all of the things that were talked about in the video, but it will most likely have the ones that will be discussed in this section. Here is a list of some cool components that you can program using App Inventor. Visit **MIT's website** for more information.

Accelerometer Sensor

This component can determine if the phone is shaking and if it's being held upright or upside-down. This capability is very useful when you want the screen to re-orient in response to how the phone is being held, or if you want the app to react to shaking. Here is a **tutorial**.

Pedometer

This sensor uses the accelerometer sensor to measure how many steps the user holding the phone takes, and can also estimate distance travelled. Here is a **tutorial**.

Gyroscope

This component can sense if the phone is being tilted. It is more precise than the accelerometer and can measure how much the orientation of the phone has changed. Here's a **tutorial**.

Clock

This allows your app to get the current time or use a timer. This component can be useful for setting a timed alarm, using a timer or knowing when it is daytime or nighttime.

Here are some tutorials:

- Timer Tutorial
- MIT Clock Documentation

Location Sensor

This sensor collects the latitude and longitude of the phone's location. This sensor can be useful anytime you need search for businesses or points of interests near the user. Here's **more information**.

Proximity Sensor

This sensor can tell if the phone is being held up to a person's face or not. However, not all phones can support this capability. Here's a **tutorial**.

Social Components

These are features that enable users to make phone calls, send emails, text and share things through your app. View MIT's Website for more information.

ContactPicker

This displays the user's contacts and allows users to choose someone from that list. Here's a **tutorial**.

EmailPicker

This component allows the user to input an email address. Here's a **tutorial**.

PhoneNumberPicker

This component allows the user pick a phone number from a contact list, such as friends or family. Here's a tutorial.

PhoneCall

This enables the user to make a phone call from your app. Here's a tutorial.

Texting

This component allows the user to send a text message to another user's phone through your app. Here's a tutorial.

Sharing

This allows users to share messages, images, or other content in your app with other apps in the user's phone. Here's a tutorial.

Twitter

This component allows communication between your app and Twitter. Users can search tweets, send and receive messages, get a list of followers, and more. Here is an example.

Connectivity Components

These allow your app to interact with places outside of your app, like the web and other apps. Visit MIT's Website for more information.

ActivityStarter

This allows your app to start up other apps installed the user's phone. These apps can be apps that were created in App Inventor or apps that weren't. They can also be apps like Camera and Maps that are pre-installed on the device. Here are some tutorials:

- Tutorial 1
- Tutorial 2
- Tutorial 3

Web

This component enables your app to get information from websites. It allows you to get information and use it without having a web browser in your app. Here's a tutorial.

BluetoothClient and BluetoothServer

These components enable your app to enable connection with Bluetooth. Here is an example.

"One of the secrets to staying young is to always do things you don't know how to do, to keep learning."

– Ruth Reichl

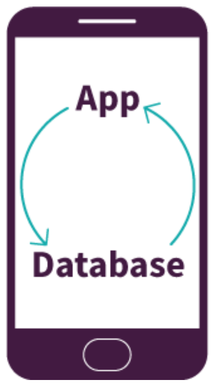
Web Databases

One of the things that you should definitely take advantage of is your phone's ability to connect to the web. This can be more than accessing the web or email through your app. You can actually store **data** on the web and access it from multiple phones running your app!

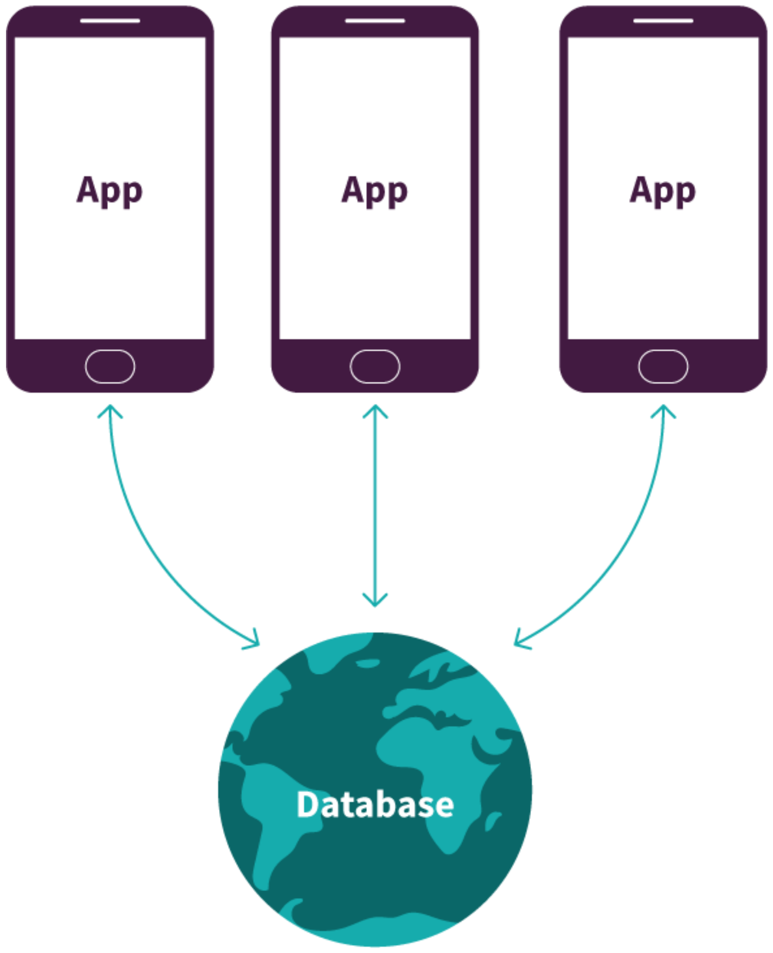
Remember when you learned about **TinyDB** in Coding Unit 2? **TinyDB** is an a *local* database that your app can store data in and get data from. When a phone uses **TinyDB**, it only stores data on that phone, so two users won't be able to see what is in each other's **TinyDB.**

A **web database** works a little bit differently. Let's say you have two phones running an app, Phone A and Phone B. When Phone A stores data into the TinyDB, Phone B can't access it since it is stored on the harddrive of Phone A. When you use a web database, Phone A can store data into a database and Phone B can access the database and retrieve the data. Instead of the database being only on one phone, now both phones using the app can share it. Another advantage of using web databases with App Inventor is that you (the programmer) can access and edit what is in it through the internet.

Local Storage



Web Database Storage



Web Databases can also be updated at any time and the phones using them will reflect the changes. Let's say you put a list of restaurants into TinyDB and put your app into the app store. Any users who download your app will have that list of restaurants. Later, you update your app to include many more restaurants. You update your code and update your app in the app store. However, the phones that downloaded your app before you updated the list won't be able to get the new restaurants. These users would have to delete the app and redownload it from the app store! If you had used a web database for the restaurants, then the app would see automatically update the list of restaurants. Each time the phone is connected to the web and accesses the database, it will get the current version of the list.

Have you ever shared a picture on Facebook or Instagram? Apps like these use web databases. When your friend uploads and shares a picture, all of her followers can see it. Web databases allow you to share and send data to everyone using the app.

So when would you use a web database? Here are some common uses:

- Sharing data from a game, like a high score list
- Allowing users to share images with each other through a feed
- Displaying a feed that updates all users see
- Requiring login and passwords for users

- Remembering everything about a user such as their transactions, financial records or medical records

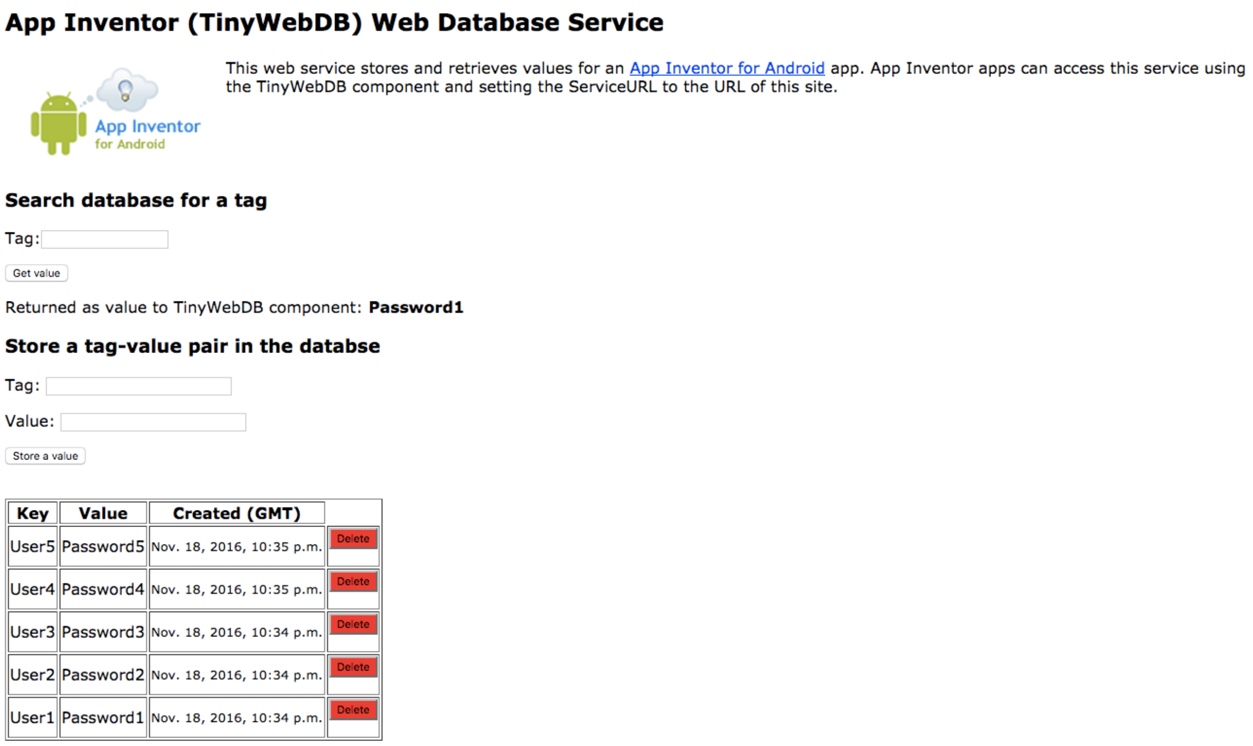
Luckily there are a few different ways to set up a web database in App Inventor! The next few sections will discuss some of the web databases that App Inventor allows you easily incorporate in your app. As you read the sections, think about which ones could potentially work with your app.

Thought Exercise

Can you think of five apps that you have used that rely on web databases?

TinyWebDB

TinyWebDB is similar to TinyDB because it uses **tags value pairs** to store and retrieve data. TinyWebDB is a great choice if you are using data where the data is mostly in pairs, like usernames and passwords. Once you get your TinyWebDB set up, you can make calls to store and get data from it. Unlike TinyDB, any phone using your app can store and get data from it. TinyWebDB also has the added bonus that you can go onto your web service and view all the tags that exist and delete ones you don't want anymore. Your web service will look like this:


App Inventor (TinyWebDB) Web Database Service

This web service stores and retrieves values for an App Inventor for Android app. App Inventor apps can access this service using the TinyWebDB component and setting the ServiceURL to the URL of this site.

Search database for a tag

Tag:

Get value

Returned as value to TinyWebDB component: **Password1**

Store a tag-value pair in the databse

Tag:

Value:

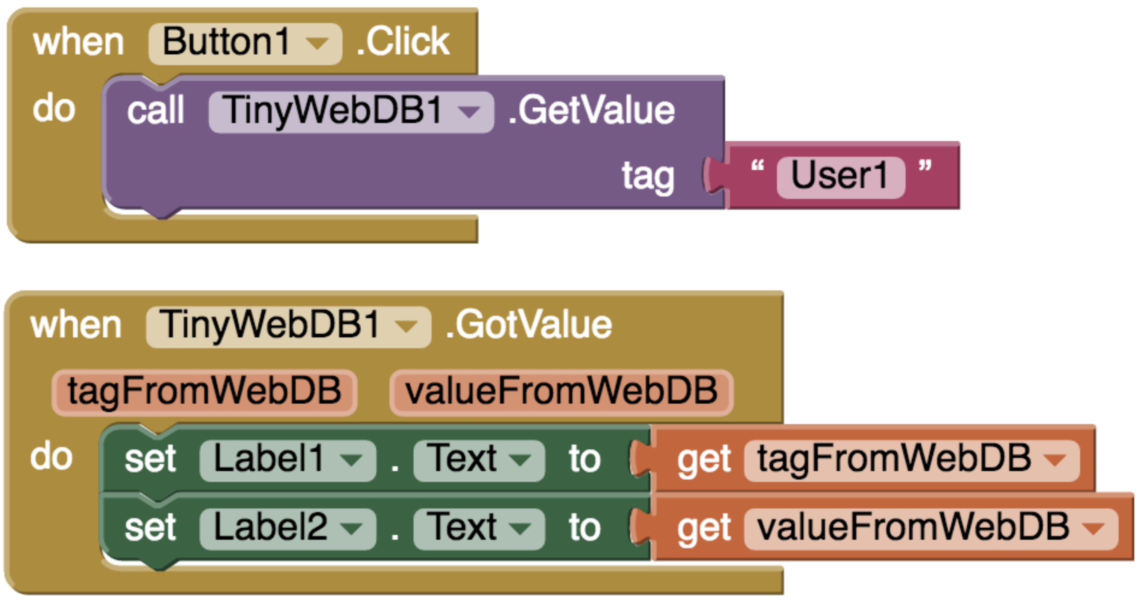
Store a value

Key	Value	Created (GMT)	
User5	Password5	Nov. 18, 2016, 10:35 p.m.	Delete
User4	Password4	Nov. 18, 2016, 10:35 p.m.	Delete
User3	Password3	Nov. 18, 2016, 10:34 p.m.	Delete
User2	Password2	Nov. 18, 2016, 10:34 p.m.	Delete
User1	Password1	Nov. 18, 2016, 10:34 p.m.	Delete

Notice how we have the option to delete some tags that are there. It's much easier to manage your data like this, so even if your app does not require data to be shared between phones, you may want to set up a **TinyWebDB** anyway!

To use **TinyWebDB**, you'll have to set up your own web service where the data will be stored and retrieved from. Here are MIT's instructions on how to do that: Custom TinyWebDB. When we set up our **TinyWebDB** found ourselves gettings you find yourself getting stuck, so here is a video we followed along with these instructions.
TinyWebDB is a non-visible component, or one that doesn't show up on the screen, and looks like this:

Programming **TinyWebDB** is slightly different than programming **TinyDB.** When you make a call to get some data you'll have to wait for your app to ask the web service for data. Once you app gets the data, then you can do stuff with it. Check out our code below:


when Button1 .Click
do call TinyWebDB1 .GetValue
tag "User1"
when TinyWebDB1 .GotValue
tagFromWebDB valueFromWebDB
do set Label1 . Text to get tagFromWebDB
set Label2 . Text to get valueFromWebDB

When the user presses Button1 the app will ask the web service for the value of the tag called "User1". Once the app has the value from **TinyWebDB,** it will then set Label1 and Label2 to the tag and value. The when TinyWebDB1.GotValue will use the last value it got from the database.

Troubleshooting tips for setting up **TinyWebDB:**

- If you are having trouble downloading Google App Engine for Python, try this.
- You do not need to put your credit card info in Google at any time, all of the cloud platform features you need are free.
- Just like with TinyDB, if you use the same tag name twice in TinyWebDB, the data will get overwritten.

Fusion Tables

Fusion tables are another type of web database that App Inventor allows you to easily use. **Fusion tables** use tables to store and retrieving data instead of a tag value pairs. This allows them to store more complex data sets. Just like **TinyWebDB,** multiple users of your app can store and retrieve values from fusion tables and you will be able to manage it through it's web service.

Fusion tables allow you to store data in a **table**. Tables can be great when you have a lot of data to keep track of, and organize data into rows and columns. To better understand **fusion tables** let's think through an example.

Let's say you were in charge of ordering new jerseys for your soccer team. Everyone gets a jersey with their name and with their number on it. You need a way to keep track of everyone's name, number, and shirt size. The easiest way to do this is by creating a table like this one:

Name	Number	Shirt Size
Marissa	2	L
Ruchi	6	M
Hadar	3	S
Stephanie	7	M
Iyonce	4	S

Instead of just using a pencil and paper, you decide to use **fusion tables** to keep track of all this information. You create an app and ask your team members to download it. From their own phones they enter their name, number, and shirt size. The information from their phones get stored on the web in a fusion table. You can then log on and view and edit everyone' information!

To get a better of idea of some things you may use fusion tables for, here's are some scenarios when past Technovation teams have incorporated fusion tables**.** Compare and see if any of the scenarios are similar to what you are trying to do with your app!

- Creating a directory of location centers that are accepting donations. The table entries include latitude and longitude, phone numbers, and items they accept.
- Storing information to create custom user profiles. Things like user location, username, and favorites
- Making maps and plotting points on maps, check out this tutorial
- Visualizing data in pie charts and graphs
- Storing users' information
- Storing posts and comments on a forum

Check out these two past apps that used fusion tables

- "CramJam" and "BCNeeds" app both used them.
  - CramJam Demo Video
  - CramJam Pitch Video
  - BCNeeds Demo Video
  - BCNeed Pitch Video
- "CramJam" used it to integrate loading of Google Docs and images for preview of online notes. "BCNeeds" used it to record donatable services and goods at Charity collection points. Both used the table to store persistent data and to integrate it with list pickers