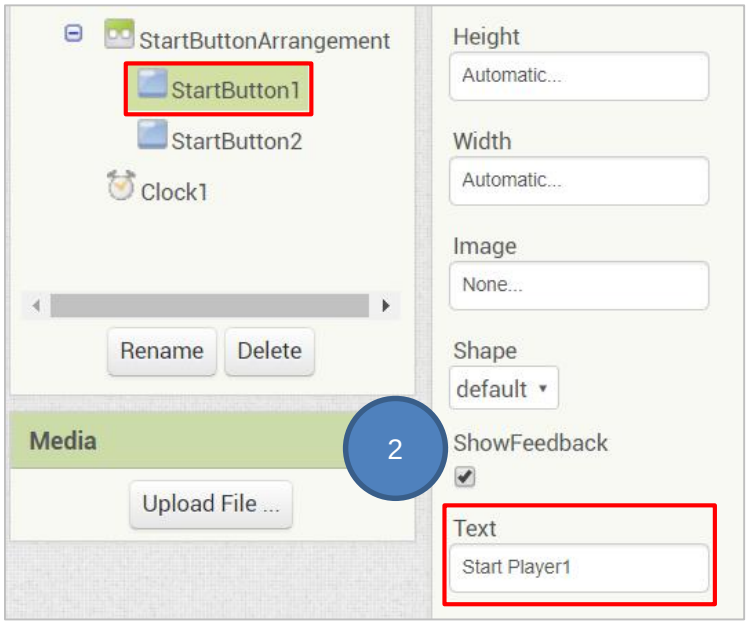
TWO-BUTTON GAME

In this lesson, we will be adding multiplayer functionality so you can play with your friend while both of you are on your own devices.

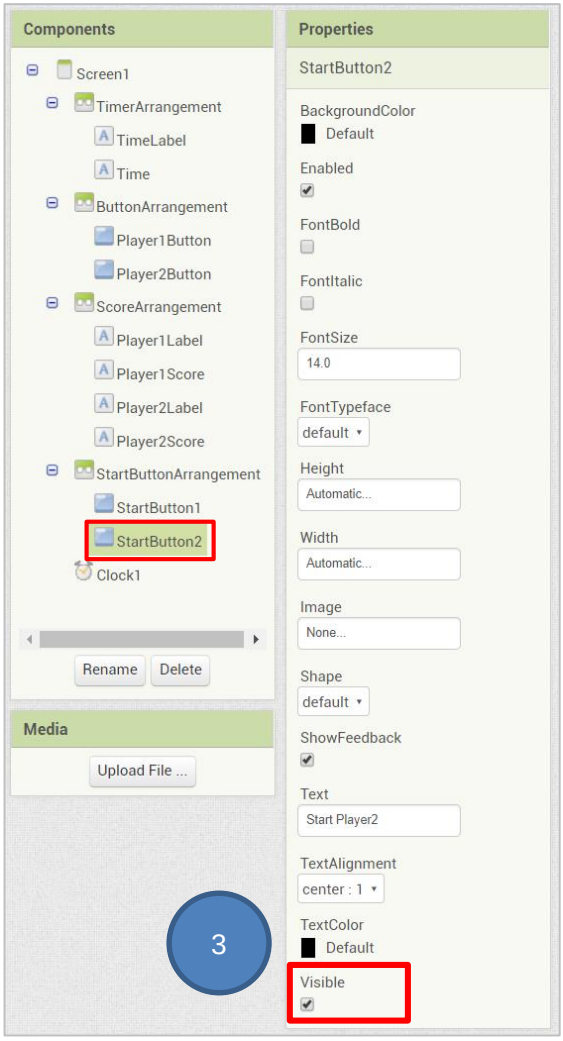
START HERE

❑ Open your TwoButtonGame app from Lesson 1, and switch to the Designer.

❑ Click on StartButton1, and change its Text property to “Start Player1”.



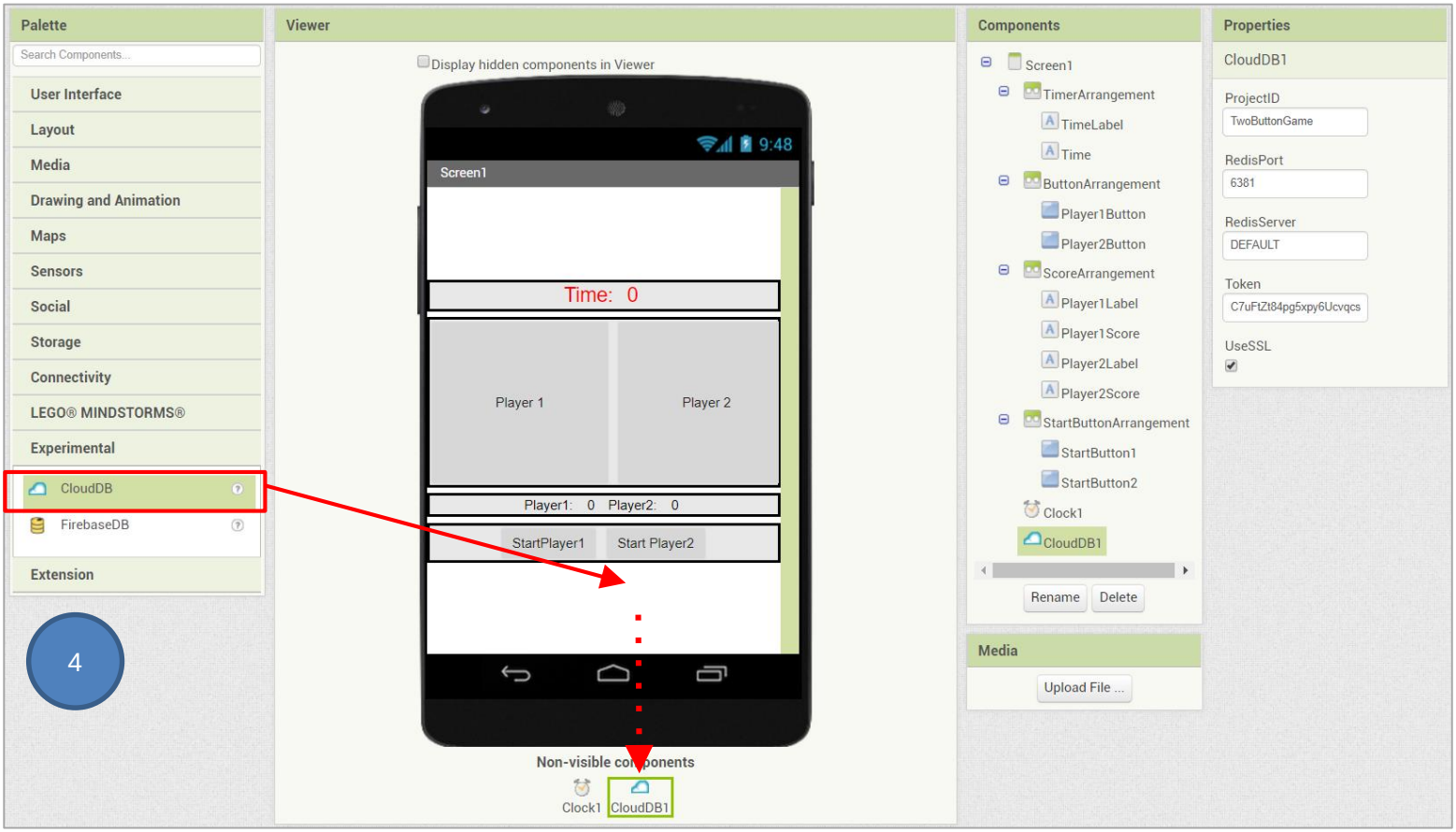
❑ Click on StartButton2, and check its Visible property, so the button appears.



TWO-BUTTON GAME

LET'S CONTINUE

❑ Drag out a CloudDB component from the Experimental drawer.
All the Properties should be filled in already.



TWO-BUTTON GAME

BLOCKS

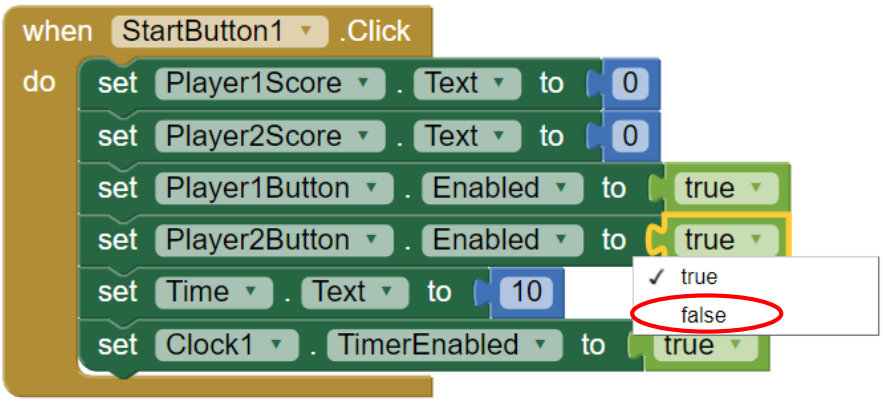
5

❑ Switch to the Blocks Editor.

The first thing we have to do is update the Start buttons. **StartButton1** should be mostly correct already.

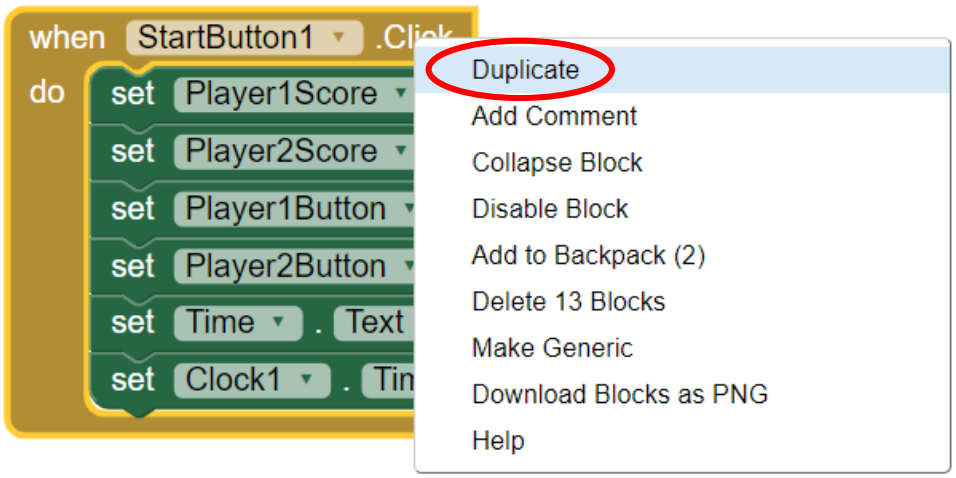
❑ **Player2Button** should **not** be enabled for Player 1. Change **Player2Button.Enabled** to **false** by clicking on **true** and selecting **false**.

6



❑ Rather than rewrite all the code, we can **duplicate StartButton1** and change it. Right click on **StartButton1.Click**, and click **Duplicate**.

7

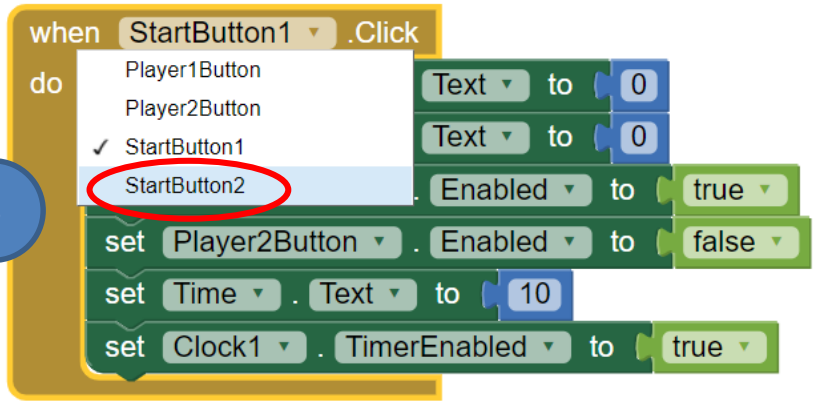


TWO-BUTTON GAME

LET'S CONTINUE

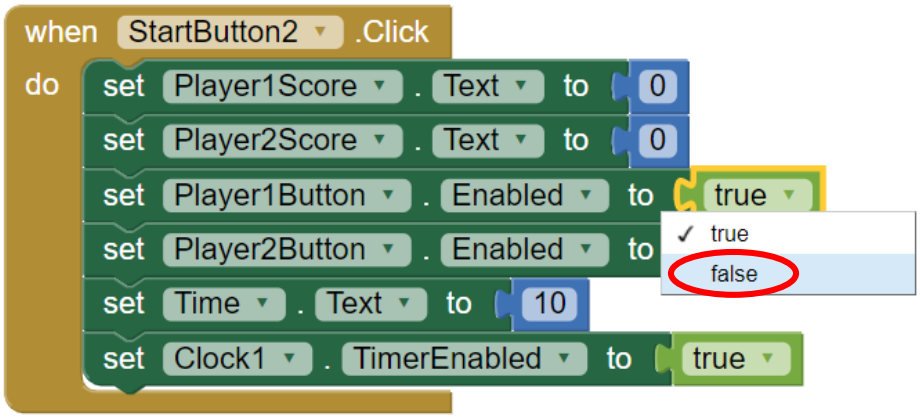
❑ Click the drop down menu of **StartButton1** and change it to **StartButton2**.

8



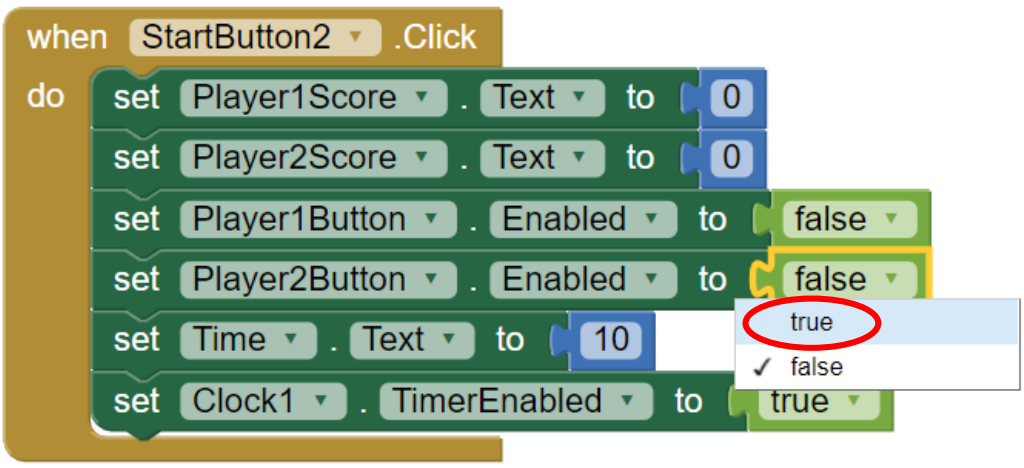
❑ Set **Player1Button.Enabled** to **false**.

9



❑ Set **Player2Button.Enabled** to **true**.

10

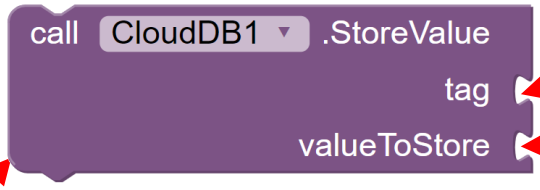


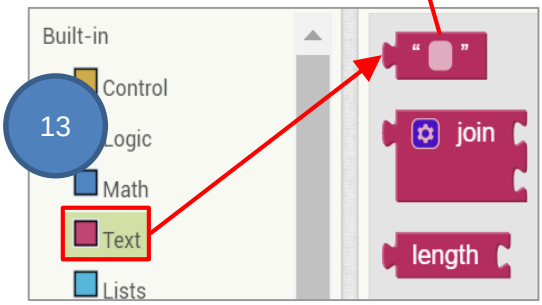
TWO-BUTTON GAME

CT Tips
CloudDB stores data on a cloud server, so you can share data between devices.

CLOUDDB STORE VALUE

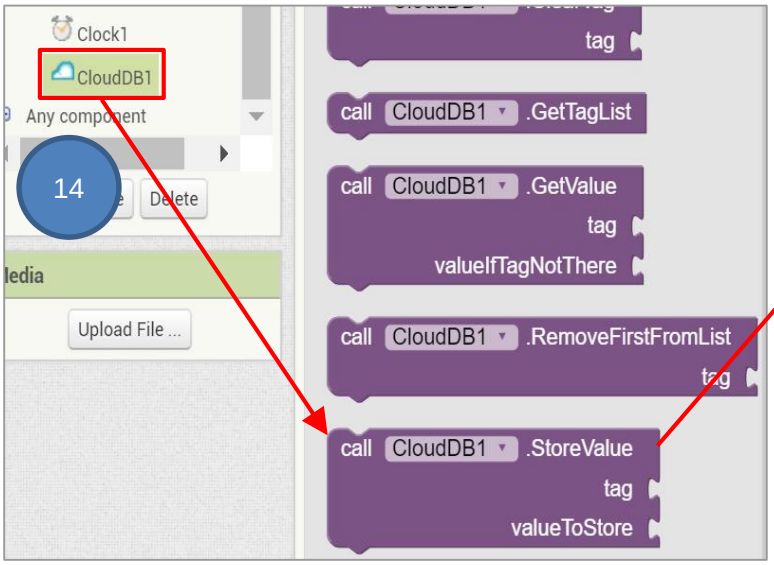
Every time a player scores a point, we need to save the new score in CloudDB.

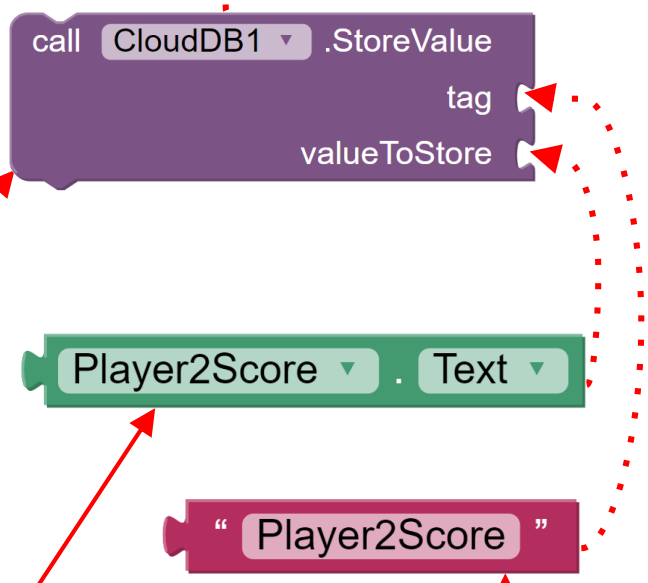


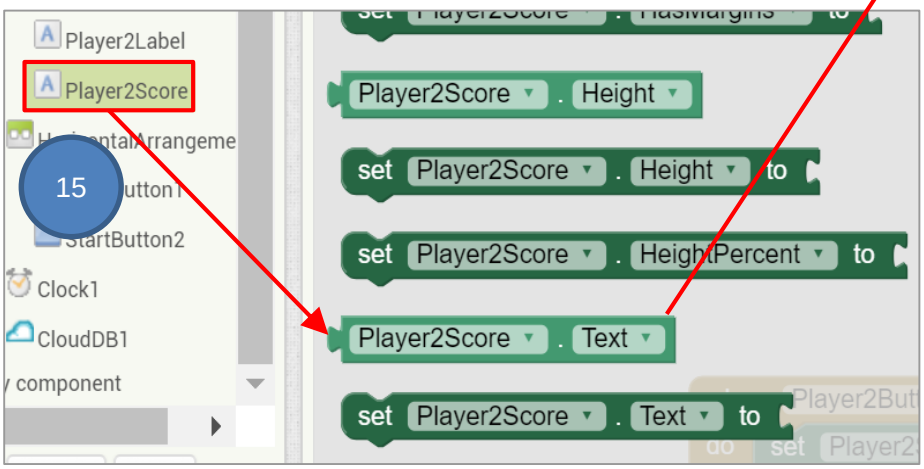


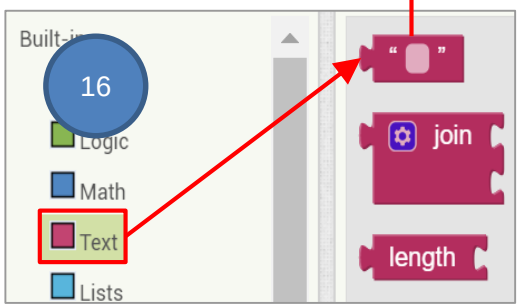
TWO-BUTTON GAME

LET'S CONTINUE







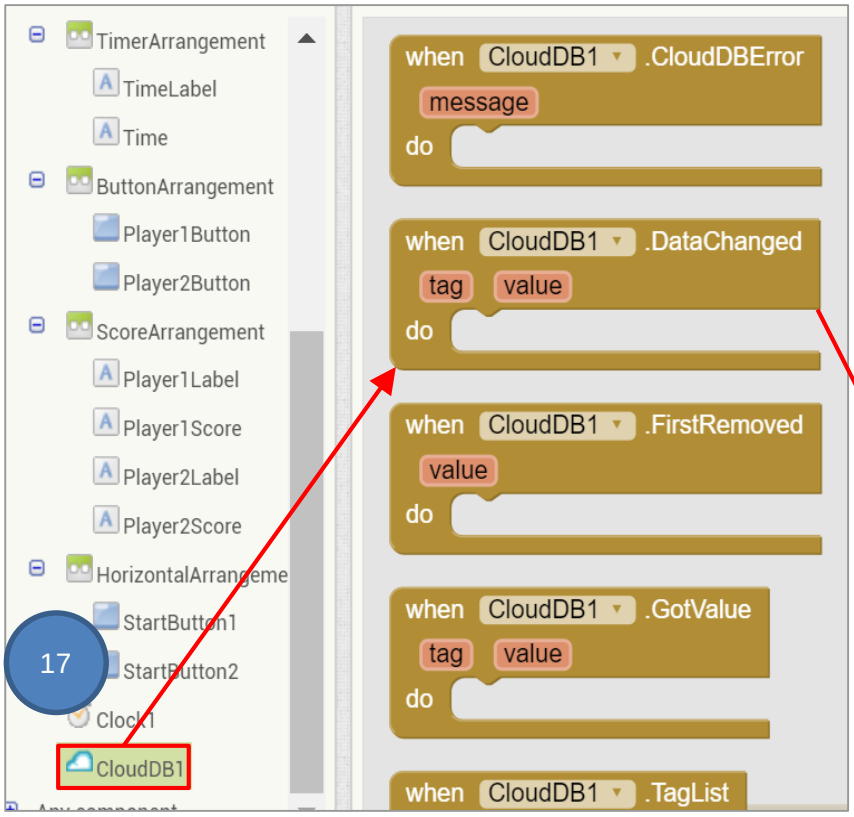


TWO-BUTTON GAME

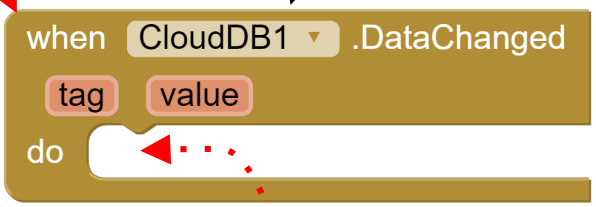
DATA CHANGED

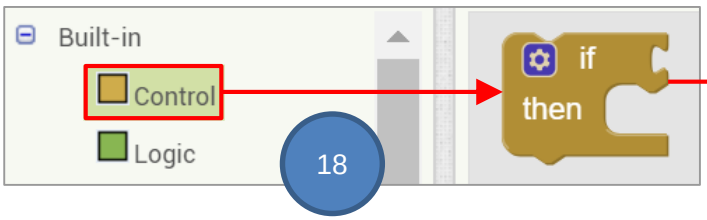
Every time we update a player's score in CloudDB, it generates a **CloudDB.DataChanged** event.

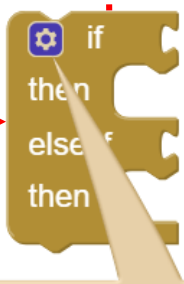
When we receive new scores, we should update them accordingly.



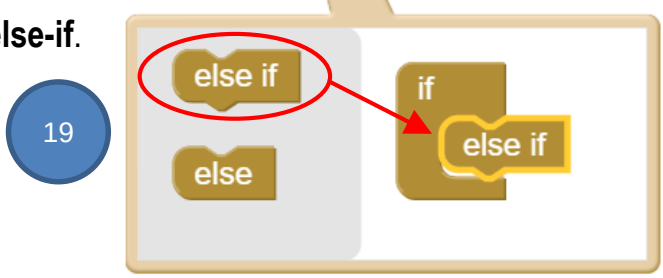
Tag: There are two possible tags based on what we stored before, **Player1Score** and **Player2Score**.
Value: The value is the **ValueToStore** we saved in the **CloudDB.StoreValue** call before.







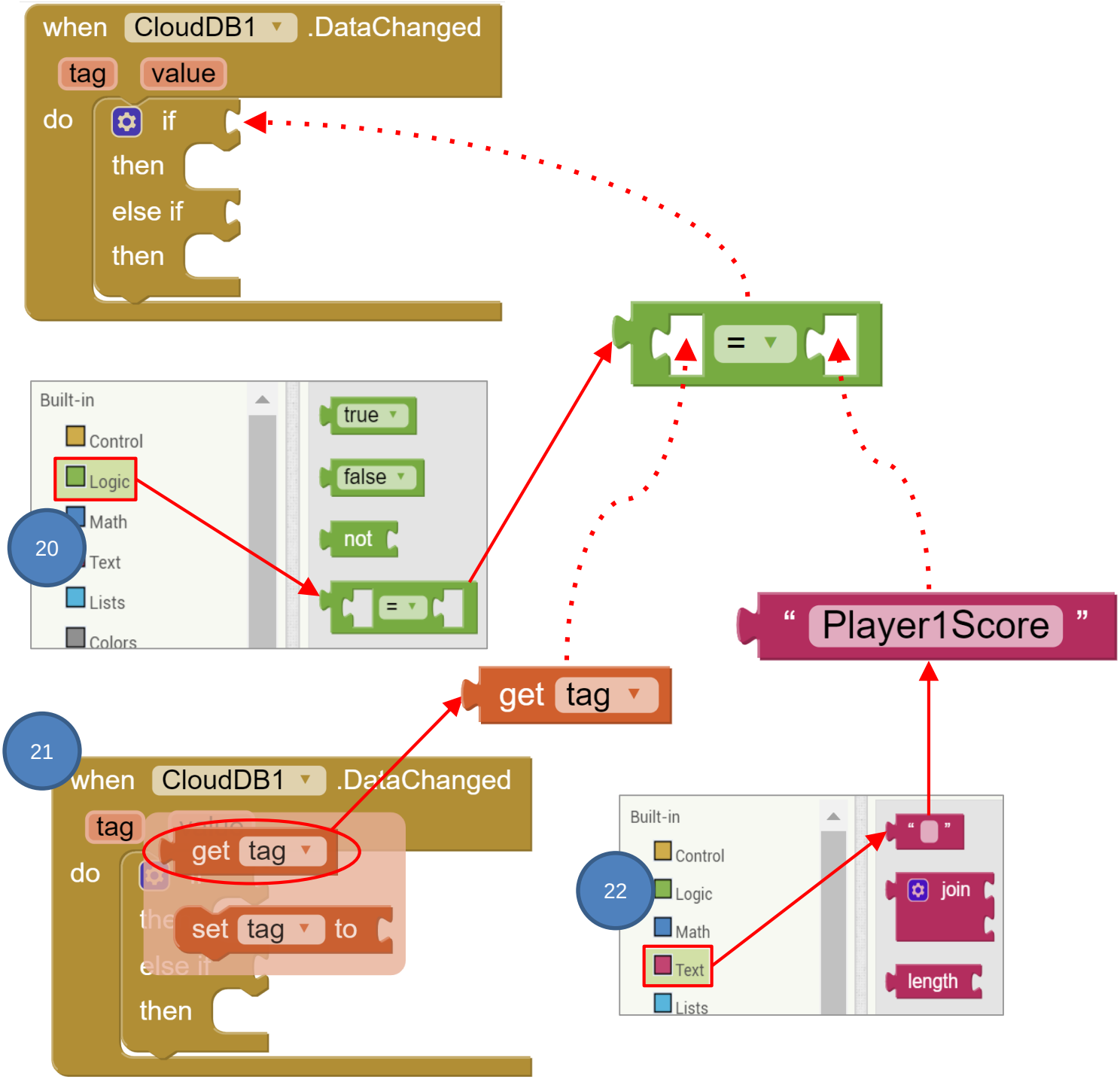
❑ Use the **mutator** of the if/then block to make an **if-else-if**.



TWO-BUTTON GAME

LET'S CONTINUE

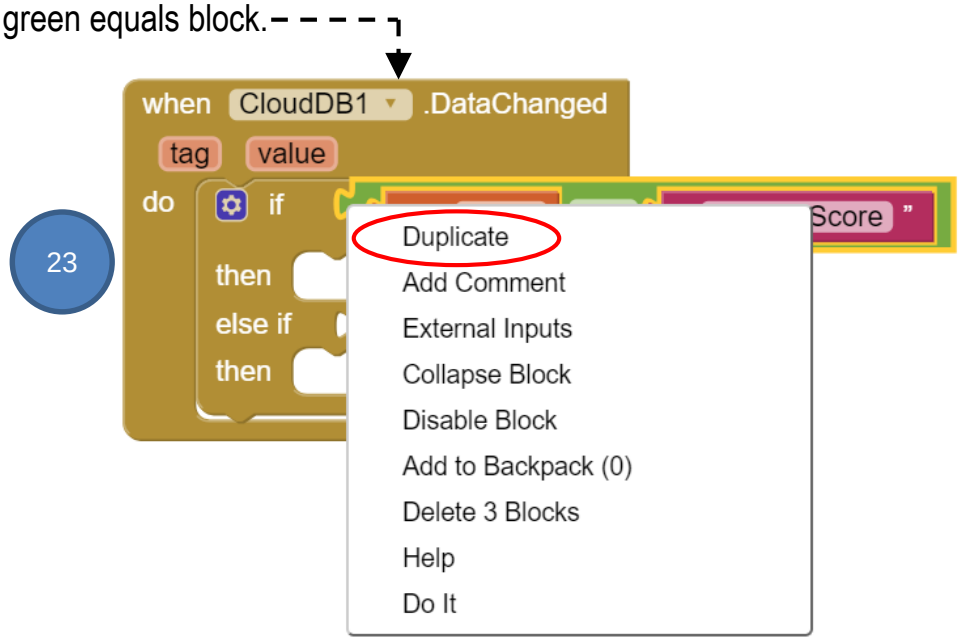
We need to check if the tag is **Player1Score** or **Player2Score**.



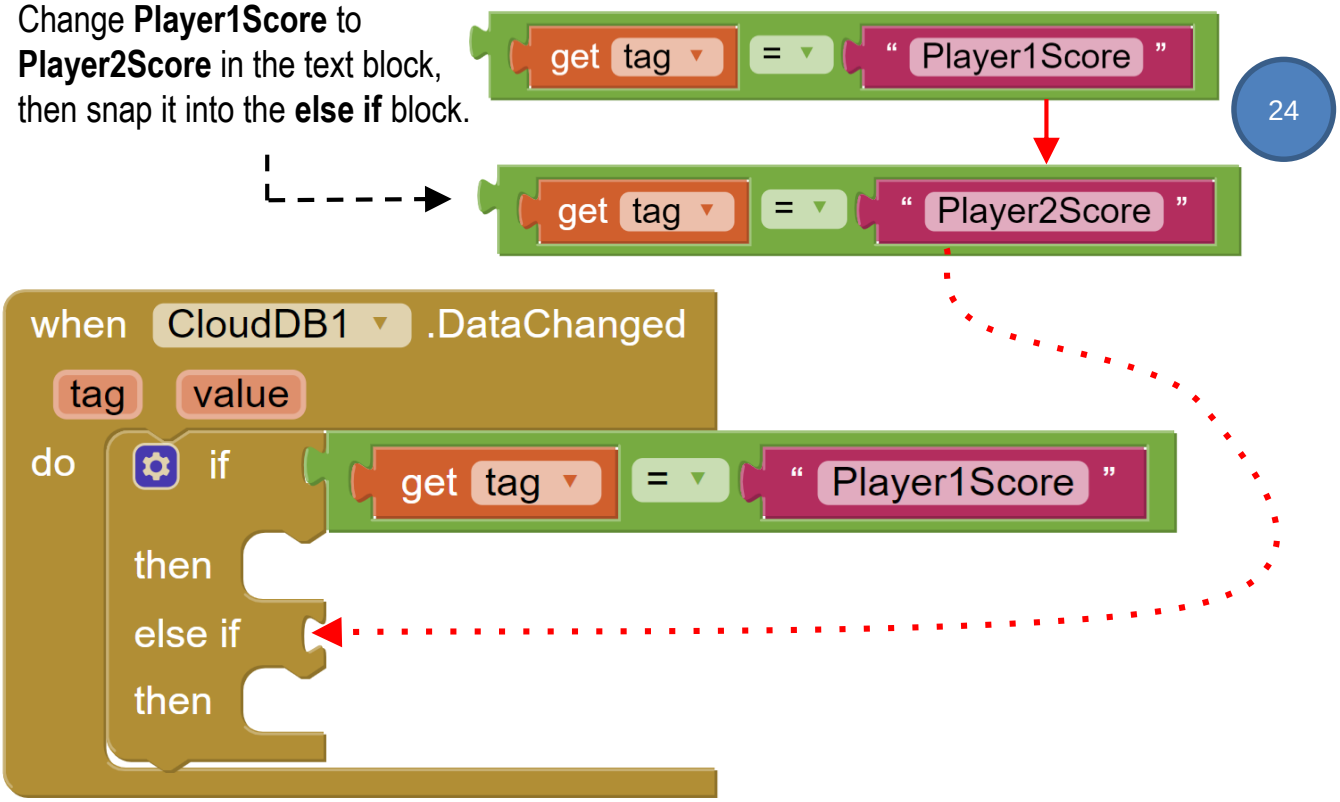
TWO-BUTTON GAME

LET'S CONTINUE

❑ **Duplicate** the green equals block.

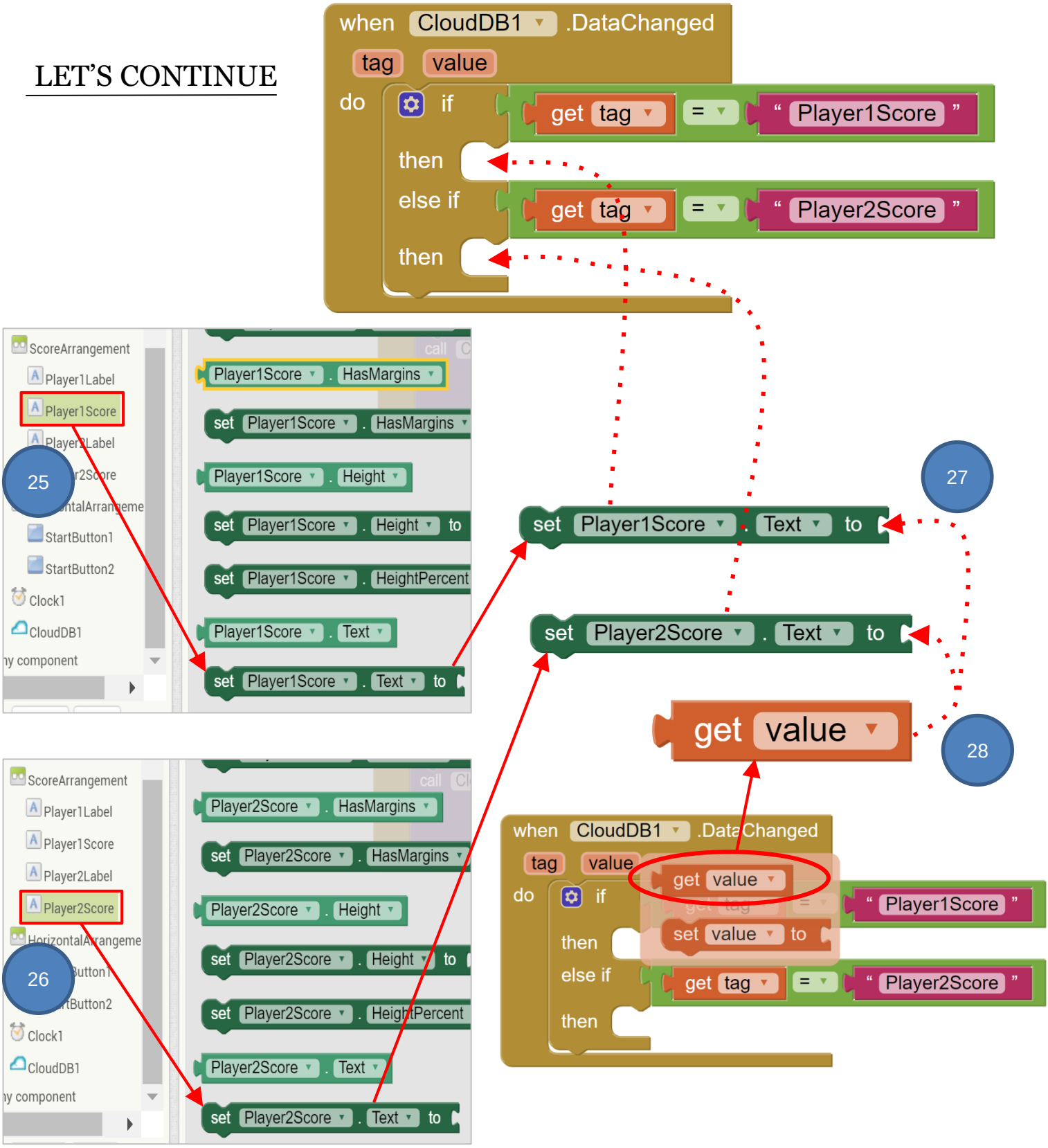


❑ Change **Player1Score** to **Player2Score** in the text block, then snap it into the **else if** block.



TWO-BUTTON GAME

LET'S CONTINUE



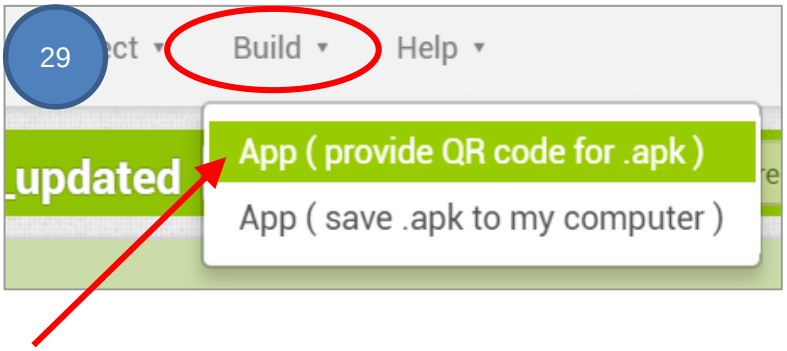
TWO-BUTTON GAME

TESTING

Time to test! Because this game app sends scores to the cloud, we need to test it using two devices. We can only test with one MIT AI2 Companion, so we'll make our project in an apk file and install the app on two separate tablets.

❑ Go to the **Build** menu at the top of the screen, and click on "App (provide QR code for .apk)". This will start the process of building the app so that it can be installed on any tablet.

29



❑ A QR code will appear in a pop-up window once the app is built. When it appears, both you and your partner should scan it using your tablets. Follow the prompts to download and install the apk file on your tablets.

30

OK

Note: this barcode is only valid for 2 hours. See the FAQ for info on how to share your app with others.

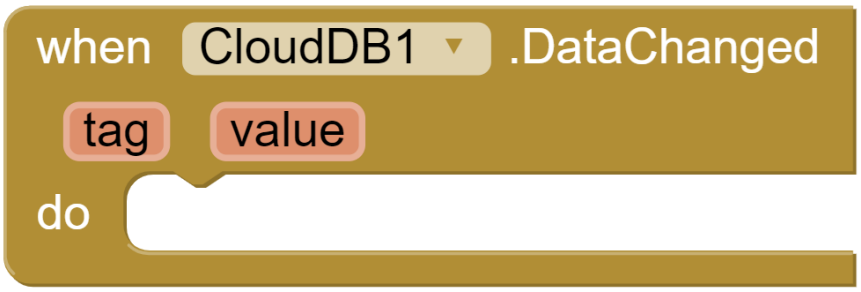
❑ Play the game against your partner on your own tablet.
Does it work correctly?
Do the scores update correctly?

31

TWO-BUTTON GAME

COMPUTATIONAL THINKING CONCEPTS

The following are the computational thinking concepts learnt in Lesson 2.

L3U8.3 Two-Button Game
1. Events: 
2. Conditionals: 
3. Data Manipulation: 

TWO-BUTTON GAME

COMPUTATIONAL THINKING PRACTICES

The following are the computational thinking practices used in this unit.

L3U8.3 Two-button Game
1. Reusing and Remixing a. Students reuse and remix an app from Level 2
2. Being incremental and iterative. a. As the second lesson builds on the first one, students will build this app incrementally
3. Testing and Debugging a. Testing and debugging is required in each lesson. In Lesson 2, students use a new method to test and debug, by installing apk files on two tablets to test for correct data transfer