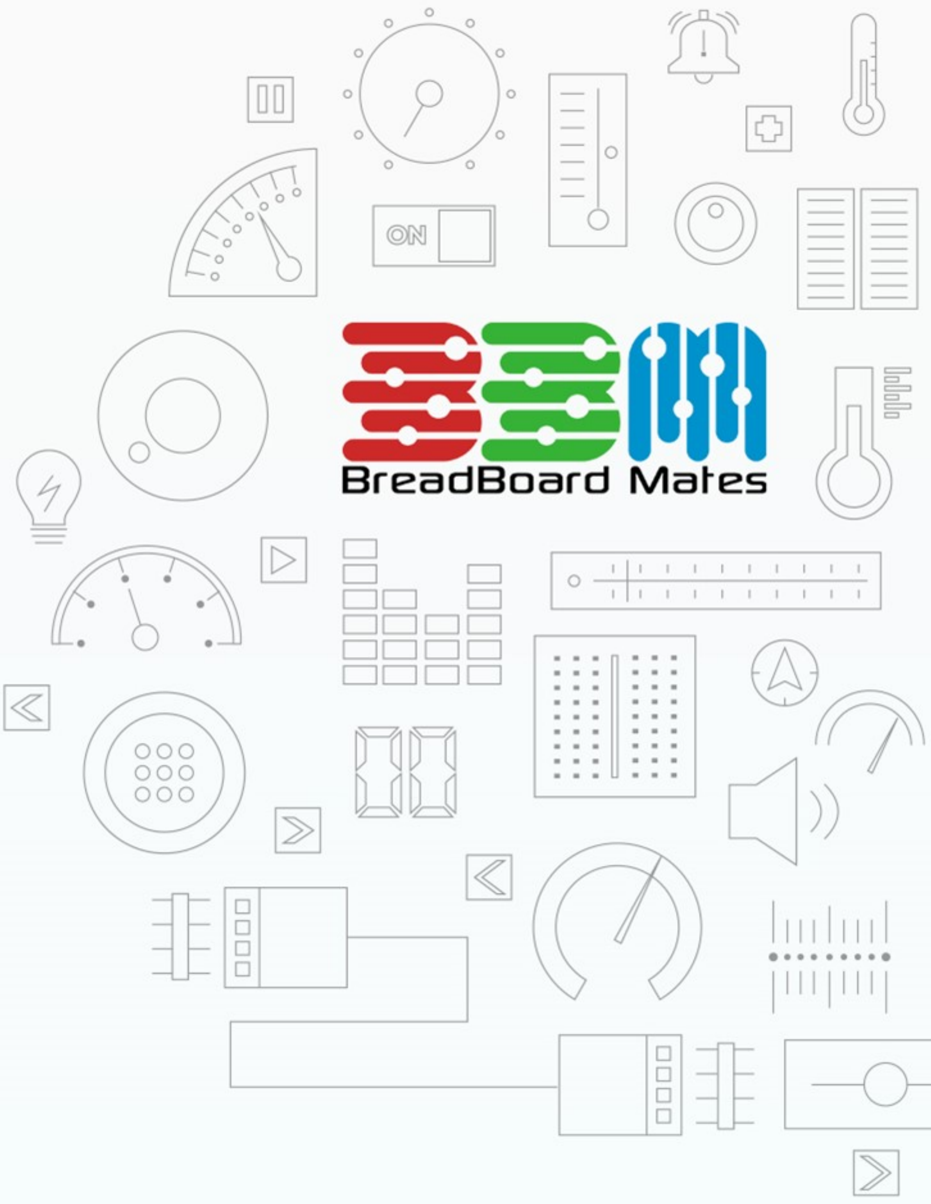


PROJECT

Simulated Time and Day of Week from Arduino

Contents

Introduction 3

Requirements 3

Hardware 3

Software 3

Graphics Design 4

Programming the Arduino 7

Running the Project 10

Downloadable Resources 10

Introduction

Arduino's are very common microcontroller boards used to study and design programmable electronics. It is often used with multiple peripherals such as buttons, sliders, sensors and motors.

Together with a TIMI acting as a small fancy display, Arduino boards become a lot more powerful and interesting to use in prototyping.

This project showcases a TIMI-96 module controlled by an Arduino Uno to output simulated day of week and time of day values.

Requirements

To proceed with the project, the following are required.

Hardware

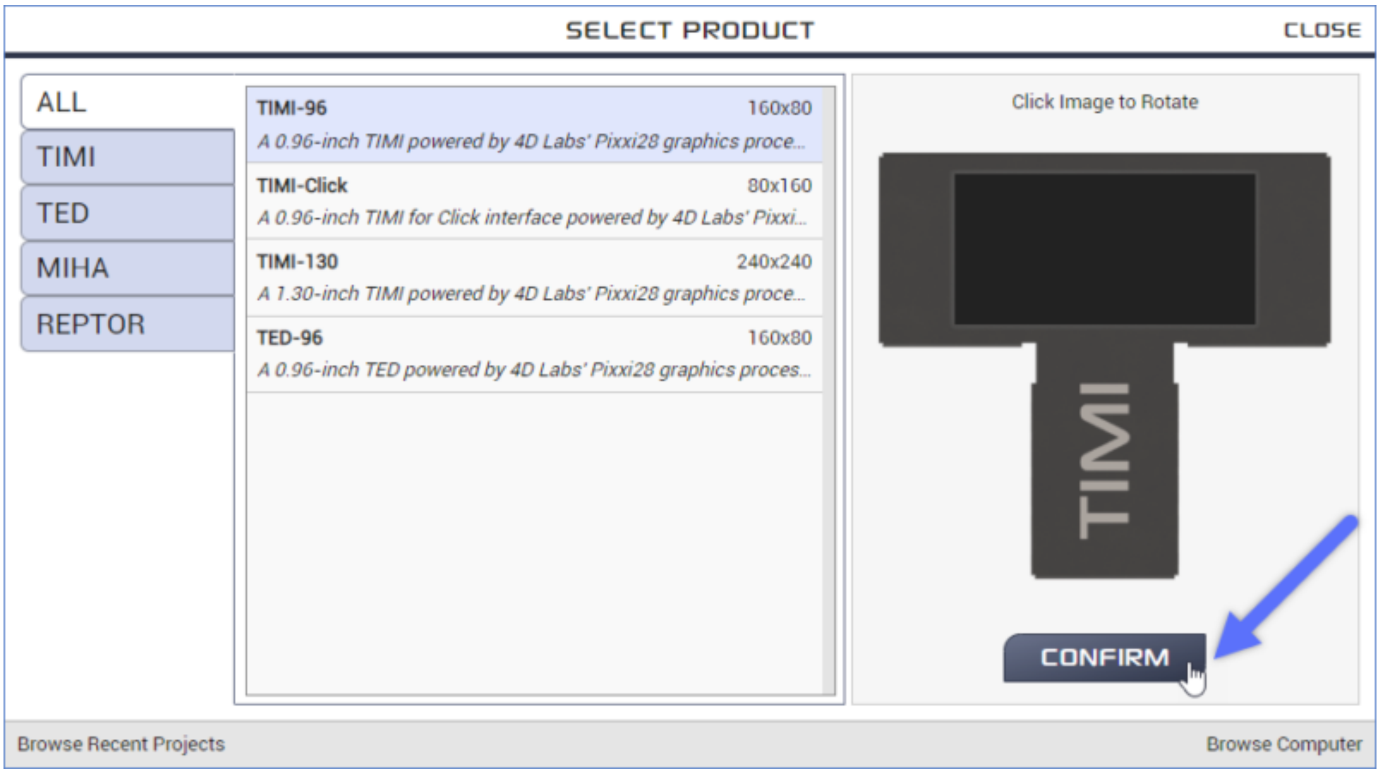
- TIMI-96
- Mates Programmer
- USB Type A to microUSB cable (for the Mates Programmer)
- USB Type A to Type B cable (for the Arduino, replace as necessary)
- Connecting Wires
- Arduino Uno
- Breadboard

Software

- Mates Studio
- Arduino IDE

Graphics Design

Step 1: Open *Mates Studio* and create a *Commander* project for *TIMI-96* with *Reversed Landscape* orientation

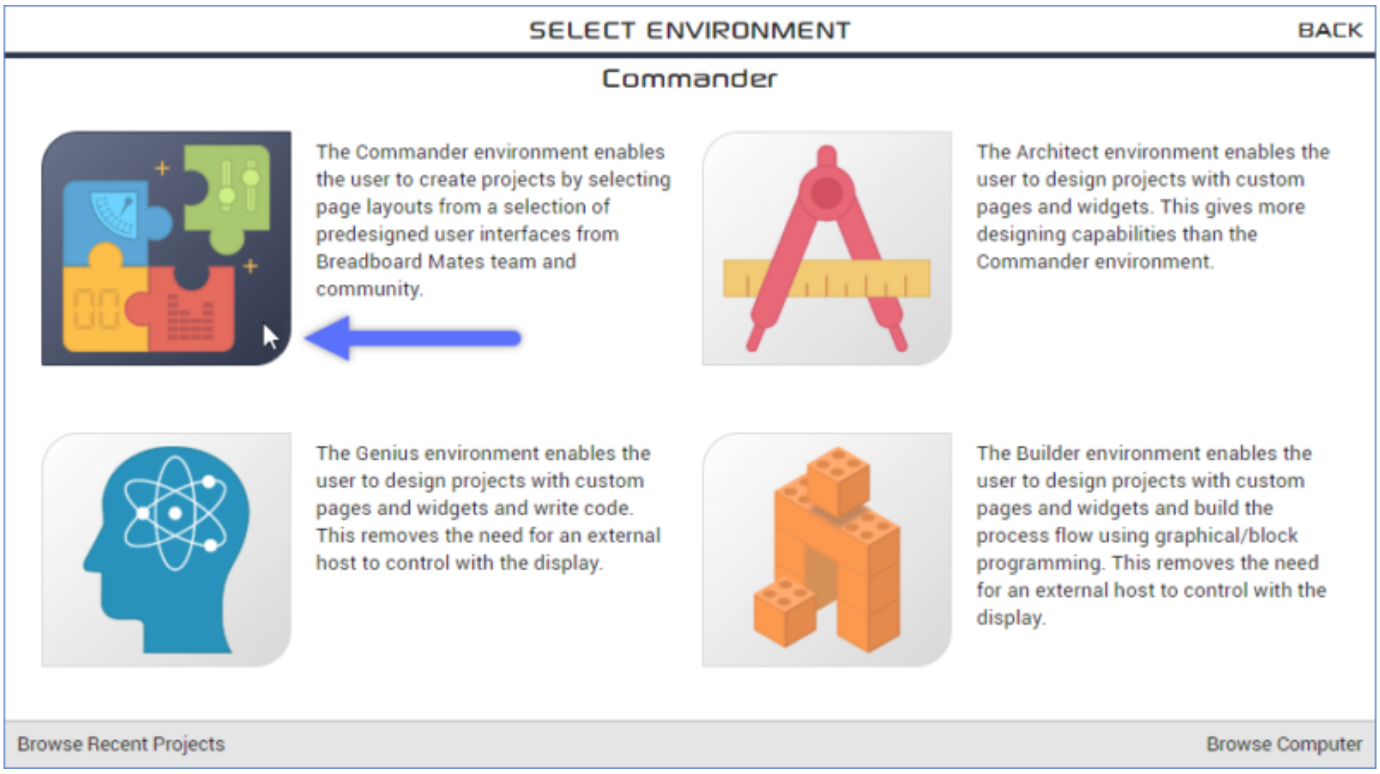


TIMI-96 160x80
A 0.96-inch TIMI powered by 4D Labs' Pixxi28 graphics proce...

TIMI-Click 80x160
A 0.96-inch TIMI for Click interface powered by 4D Labs' Pixxi...

TIMI-130 240x240
A 1.30-inch TIMI powered by 4D Labs' Pixxi28 graphics proce...

TED-96 160x80
A 0.96-inch TED powered by 4D Labs' Pixxi28 graphics proces...



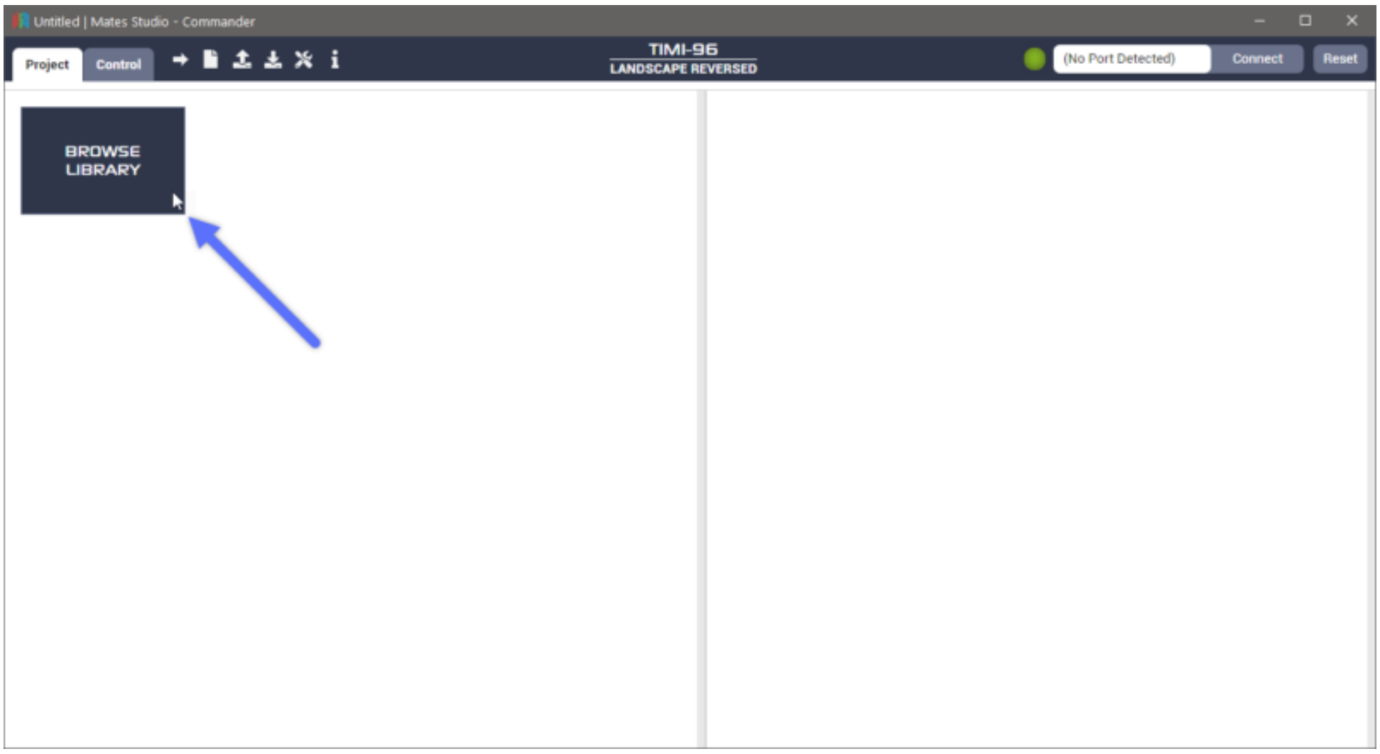
The Commander environment enables the user to create projects by selecting page layouts from a selection of predesigned user interfaces from Breadboard Mates team and community.

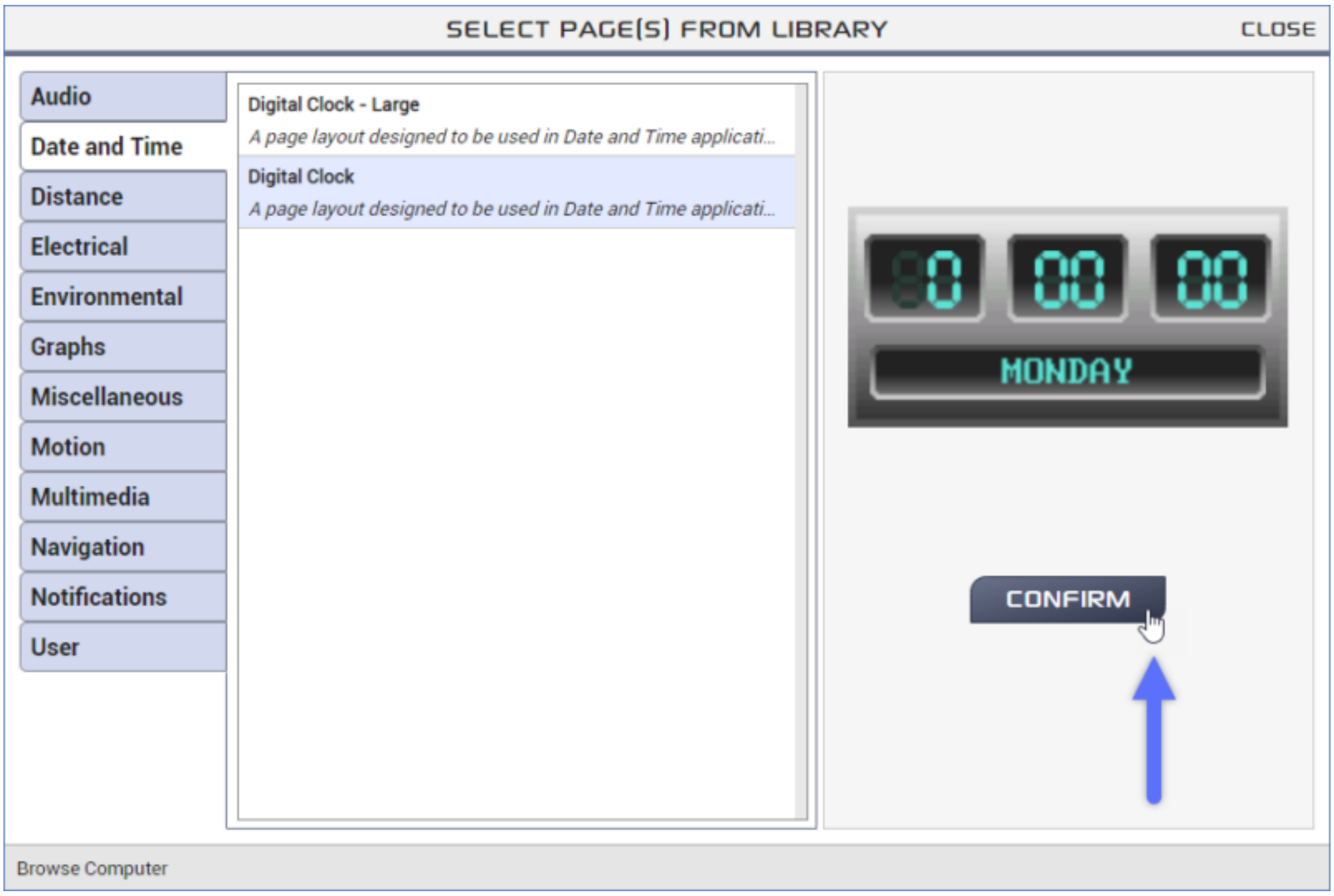
The Architect environment enables the user to design projects with custom pages and widgets. This gives more designing capabilities than the Commander environment.

The Genius environment enables the user to design projects with custom pages and widgets and write code. This removes the need for an external host to control with the display.

The Builder environment enables the user to design projects with custom pages and widgets and build the process flow using graphical/block programming. This removes the need for an external host to control with the display.

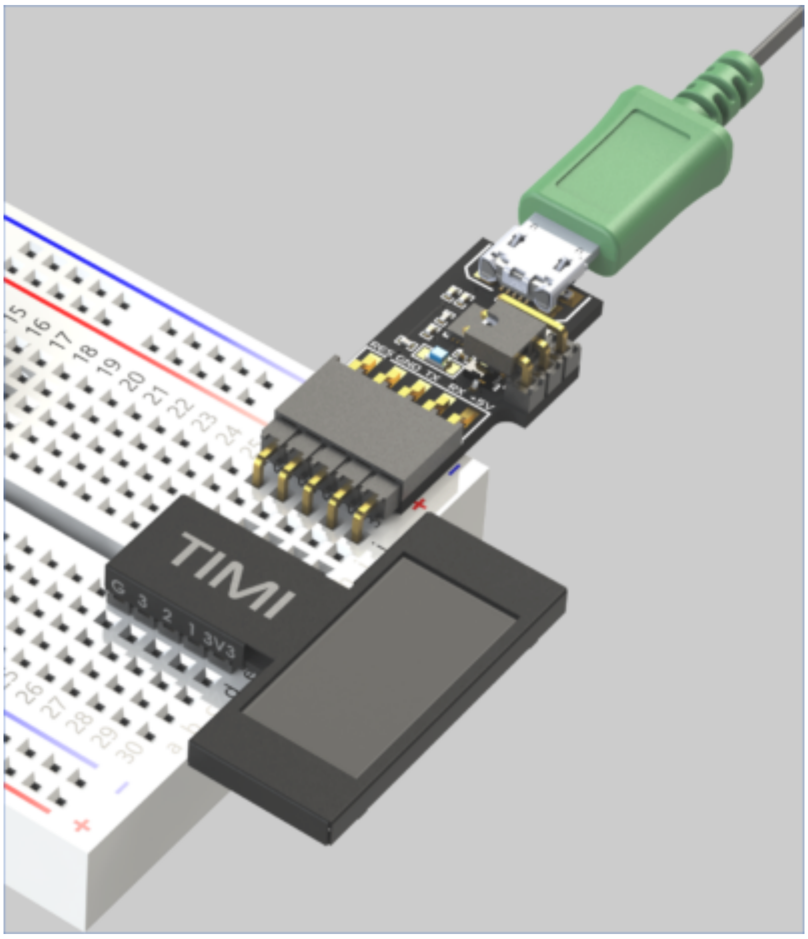
Step 2: Browse the library for appropriate page designs. For this project, *Digital Clock* page under *Date and Time* category was used.



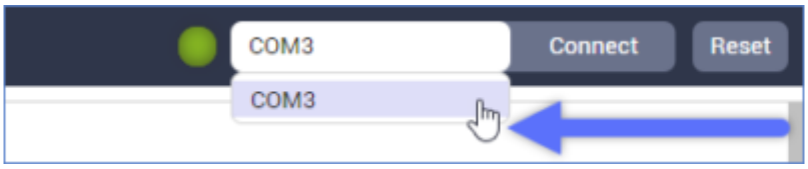


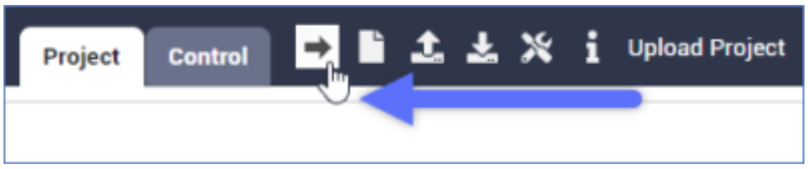
Digital Clock - Large
A page layout designed to be used in Date and Time applicati...
Digital Clock
A page layout designed to be used in Date and Time applicati...

Step 3: After finalizing the design, connect TIMI-96 to your computer

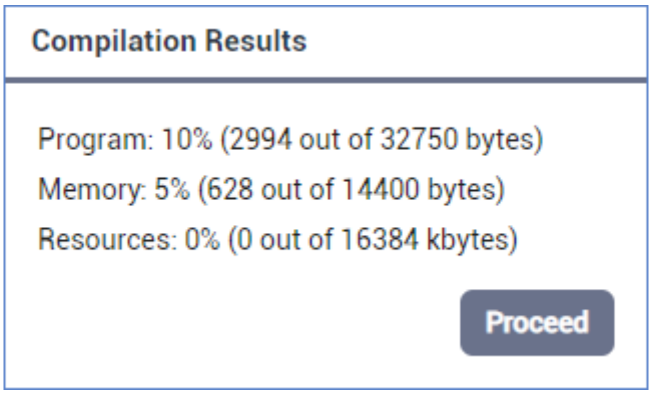


Step 4: Upload the project to the appropriate COM port





Step 5: When prompted, click *Proceed* to continue with upload.



Compilation Results

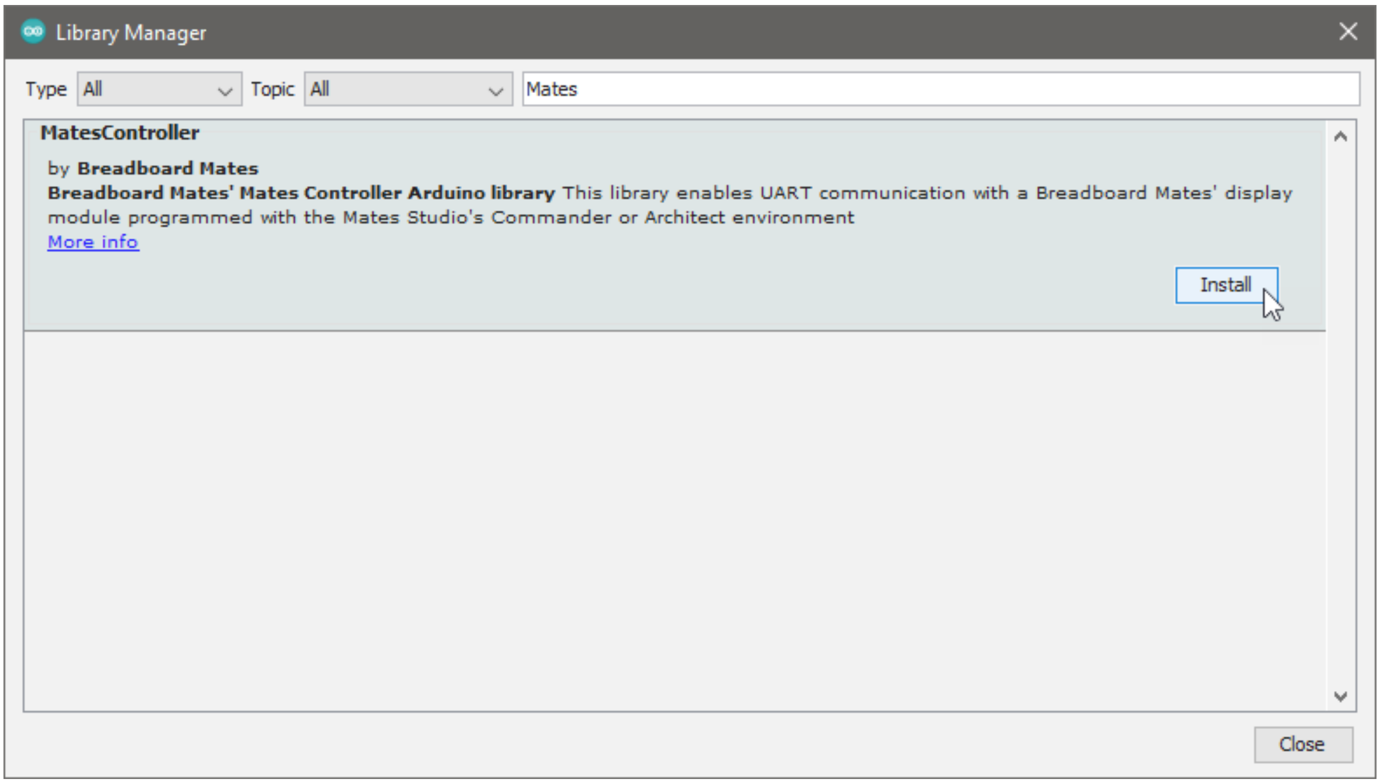
Program: 10% (2994 out of 32750 bytes)
Memory: 5% (628 out of 14400 bytes)
Resources: 0% (0 out of 16384 kbytes)

Note

It is recommended that the graphics design is finalized before moving to the next steps when working on a project

Programming the Arduino

Step 1: Install the *MatesController* library using Arduino's *Library Manager*.



MatesController

by **Breadboard Mates**
Breadboard Mates' Mates Controller Arduino library This library enables UART communication with a Breadboard Mates' display module programmed with the Mates Studio's Commander or Architect environment
More info

Step 2: Include *MatesController.h* to your project

```
#include "MatesController.h"
```

Step 3: Create a *MatesController* instance named *mates*.

```
MatesController mates = MatesController(Serial);
```

This will initialize the *MatesController* instance to the default reset pin 4 using a LOW pulse.

Step 4: (Optional) Create a function for toggling the built-in LED of the Arduino board. This can be used for debugging or showing errors if the Serial monitor can't be used.

```
int errLedStatus = LOW;
void ErrorLed_Toggle() {
  errLedStatus = ~errLedStatus;
  digitalWrite(LED_BUILTIN, errLedStatus);
}
```

Step 5: (Optional) At the beginning of the setup function, set the built-in LED pin to OUTPUT and set it to LOW.

```
pinMode(LED_BUILTIN, OUTPUT);
digitalWrite(LED_BUILTIN, errLedStatus);
```

Step 6: To start using the MatesController instance, use the `begin` function

```
mates.begin();
```

Step 7: (Optional) The `begin` function can be enclosed in an if condition to handle initialization errors.

```
if (!mates.begin()) {
  // Display didn't send ready signal in time
  while (1) {
    ErrorLed_Toggle();
    delay(100);
  }
}
```

Step 8: To print the day of week, an array of strings needs to be prepared.

```
// Days of Week Strings
const char * daysOfWeek[] = {
  "SUNDAY",
  "MONDAY",
  "TUESDAY",
  "WEDNESDAY",
  "THURSDAY",
  "FRIDAY",
  "SATURDAY"
};
```

Step 9: Integer variables for day of week, time and a few other variables are also required to store and simulate values.

```
// Simulated Timer Variable
unsigned long lastUpdate;

// Test Start Values: Monday, 23:59:47 (will actually start at 48 seconds)
uint8_t lastDay = 1;
uint8_t dy = 1;
int16_t hr = 23;
int16_t mn = 59;
int16_t sc = 47;
```

Step 10: In the loop function, the values are simulated and sent to TIMI as necessary.

```
void loop() {
  if (millis() - lastUpdate >= 1000) {

    lastUpdate = millis();

    sc++;
    if (sc == 60) {
      sc = 0;
      mn++;
    }
    if (mn == 60) {
      mn = 0;
      hr++;
    }
    if (hr == 24) {
      hr = 0;
      dy++;
    }
    dy %= 7;

    // mates.setWidgetValue(MATES_LED_DIGITS, 0, hr);
    // mates.setWidgetValue(MATES_LED_DIGITS, 1, mn);
    // mates.setWidgetValue(MATES_LED_DIGITS, 2, sc);
    mates.setLedDigitsValue(0, hr);
    mates.setLedDigitsValue(1, mn);
    mates.setLedDigitsValue(2, sc);

    if (lastDay != dy) {
      mates.updateTextArea(0, daysOfWeek[dy]);
      lastDay = dy; // prevents writing the same text to TextArea
    }

  }
}
```

As shown, the simulated values utilizes a second non-blocking delay to simulate a second increment in time. The minutes, hours and day of week are updated accordingly.

Step 11: Lastly, to ensure that the day and time values are written to TIMI for the first loop, the lastUpdate variable should be set to 1s behind at the end of the setup function.

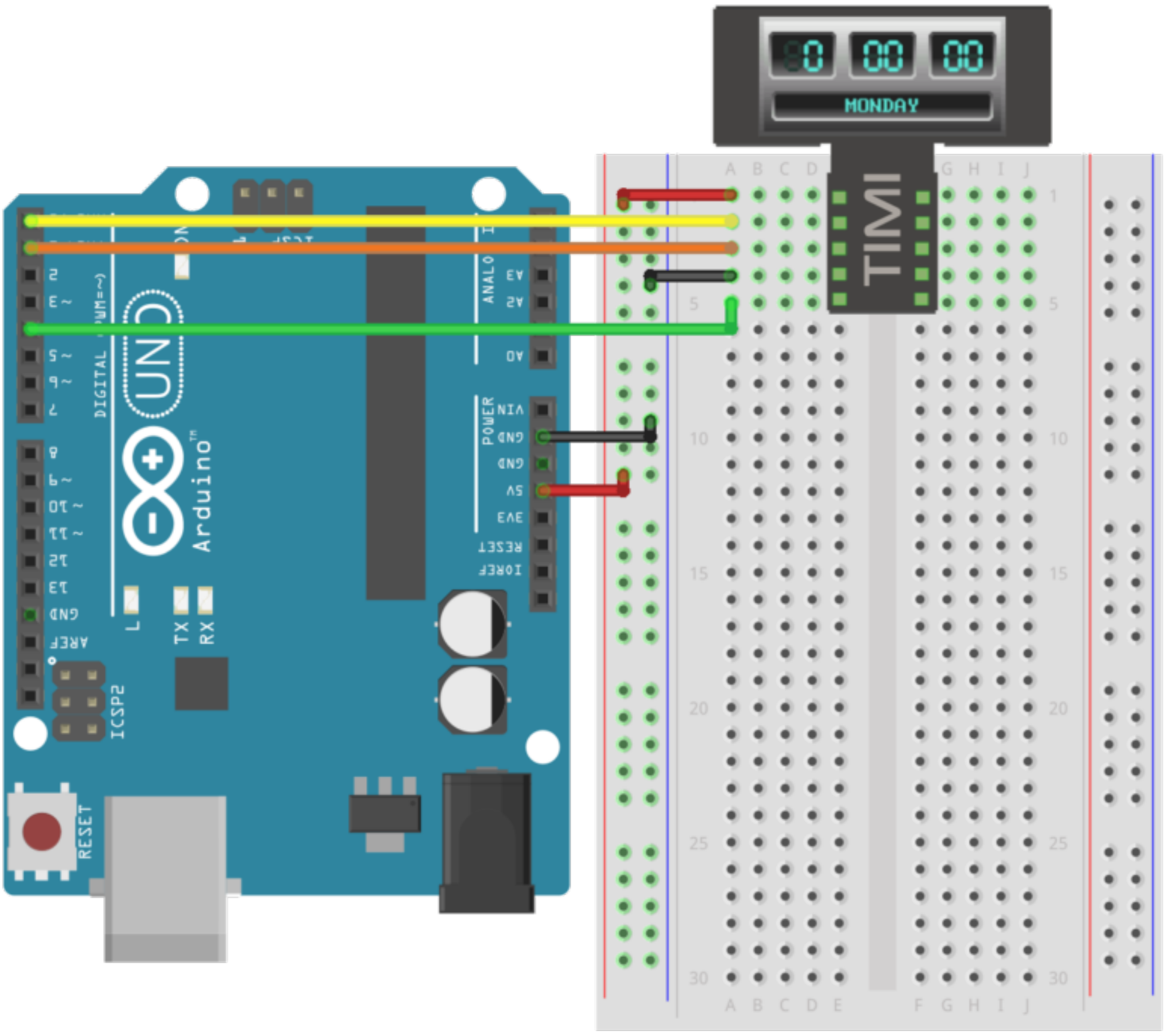
```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, errLedStatus);

  if (!mates.begin()) {
    // Display didn't send ready signal in time
    while (1) {
      ErrorLed_Toggle();
      delay(100);
    }
  }

  lastUpdate = millis() - 1000; // Ensures first write
}
```

Running the Project

After designing the user interface for TIMI and writing code for the Arduino and programming them, it is time to connect the devices together. Follow the diagram below for the connection between TIMI and Arduino.



Finally, supply power to the Arduino and observe the behavior of the project.

Downloadable Resources

The Mates Studio – Commander project and Arduino sketch are included in the MatesController library.

The Commander project is available under the extras folder of the library. You can find it in (if the library was installed using Arduino Library Manager):

```
C:\Users\%USERNAME%\Documents\Arduino\libraries\MatesController\extras\Day and Time.mates
```

Here are the links to the software applications, libraries and completed project files.

- Mates Studio
- Arduino IDE

- Arduino Mates Controller Library
- Project Files