ElectroTutor:
Test-Driven Physical Computing Tutorials

Jeremy Warner, Ben Lafreniere, George Fitzmaurice, Tovi Grossman
electronics tutorials
motivation

instructables

[Image of a robot and the TinkerCAD logo]
What if we could embed checkpoints to improve the tutorial experience?
ideality
progress

configuration

code bug

miswiring

reality

time

progress
challenges

- no incremental testing
- large cost or steep learning curve
- debuggers are general purpose, content-agnostic

difficulty scoping errors
debugging tools often novice-inaccessible
weaker ability to guide novice users
Trigger-Action-Circuits
Anderson et al. UIST 2017

Smart Makerspace
Knibbe et al. ITS 2015
related

Test Driven Development
Kent Beck

incremental testing with specific test cases

mitigating fear, inspiring confidence
ElectroTutor
ElectroTutor
tutorial content

<YAML>

system

<Markdown>
test taxonomy

- setup
  - types: automatic, manual, none
- verification
  - automatic, manual
- domain
  - hardware, code, knowledge
ElectroTutor

Test Examples
project: light-sensitive alarm clock
runtime hardware test

```cpp
void setup() {}
void loop() {} ...
```

Measure the voltage across the button using the tester. Connect the black wire to GND (ground) and the white wire to the Arduino-connected side. The voltage should drop as you press the button down — hold it down for AT LEAST 2 seconds.
```c
#define thresh_to 100
#define thresh_hi 900
#define button 2
#define outLED 13
#define buzzer 4
#define freq 220

int color = 0;
Adafruit_NeoPixel strip = Adafruit_NeoPixel(PIXELS, PIN, NEO_GRB + NEO_KHZ800);
uint32_t white = strip.Color(16, 16, 16); // dim white color
void setup() {
  pinMode(light, INPUT);
  pinMode(button, INPUT_PULLUP);
}
```
wiring test
Step 9: Tone Generation Code

To stop the piezo's beeping, press the **Reset** button in the Arduino IDE.

Next, in software, add code to set up the system to buzz whenever the button is pressed.

Add these statements at the top of the file to define the buzzer pin and note frequency:

```c
#define pin buzzer 5
#define freq 220
```

Then, inside the body of `loop()` function, set the case to play when the button is pressed:

```c
if (press) {
  digitalWrite(buzzer, HIGH);
  delay(10);
  digitalWrite(buzzer, LOW);
}
```

Pass the tests to continue.

Pass all the tests in the right-hand pane to continue.
// Starter code - this is a comment.
#define button 2
#define outLED 13
#define buzzer 4
#define freq 220

void setup()
{
  pinMode(button, INPUT_PULLUP);
  pinMode(outLED, OUTPUT);
}

void loop()
{
  int press = digitalRead(button);
  digitalWrite(outLED, press);
  if (press) {
    tone(buzzer, freq, 10);
    delay(10);
  }
}

Testing

Unpassed Tests
Pass the tests to continue.

Code Selection Test

Highlight the part of your code which turns on the buzzer.

Make sure to include the buzzer code line as described in the left pane.

int press = digitalRead(button);

Check Code Selection
Insert the button onto the breadboard, as pictured.

Connect the one side of the button to GND (ground) with a jumper wire.
Connect the other side to digital pin 2 on the Arduino board.
evaluation
We were interested in seeing how ElectroTutor would shape participants’ usage patterns, knowledge retention, and subjective experiences while working with the tutorial.
<table>
<thead>
<tr>
<th></th>
<th>conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ElectroTutor</td>
<td>tests enabled, pass to progress</td>
</tr>
<tr>
<td>Control</td>
<td>no tests, no progress restrictions</td>
</tr>
<tr>
<td>evaluation</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>participants</td>
<td></td>
</tr>
<tr>
<td>12 total (10 male, 2 female, ages 20-54)</td>
<td></td>
</tr>
<tr>
<td>study design</td>
<td></td>
</tr>
<tr>
<td>between-subjects (6 ElectroTutor, 6 control)</td>
<td></td>
</tr>
<tr>
<td>skill</td>
<td></td>
</tr>
<tr>
<td>no electronics work, varied software skill</td>
<td></td>
</tr>
</tbody>
</table>
light-sensitive alarm clock
tutorial has 16 steps, 24 tests
evaluation
project
<table>
<thead>
<tr>
<th>results</th>
</tr>
</thead>
<tbody>
<tr>
<td>completion</td>
</tr>
<tr>
<td>timing</td>
</tr>
<tr>
<td>knowledge</td>
</tr>
<tr>
<td>back steps</td>
</tr>
<tr>
<td>completion rate</td>
</tr>
<tr>
<td>time to completion</td>
</tr>
<tr>
<td>post-tutorial quiz</td>
</tr>
<tr>
<td>progression patterns</td>
</tr>
</tbody>
</table>
results

$\max = 6$

completion

![Bar chart showing completion rates for different groups.](chart.png)
results

Completion Time

- ElectroTutor: 39.7 minutes
- Control: 41.5 minutes

timing
results

max score = 4

knowledge

Knowledge Test

ElectroTutor
Control

2.8

2.0
results

progress

back steps

16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
0 5 10 15 20 25 30 35 40 45

time (minutes)

step
results

back steps

progress

time
results

ElectroTutor

Control

back steps

less back steps 0.2 steps average

18.8 steps average
It’s fun making things happen. I was never good at electronics in university, so... it’s fun and breaks it down really nicely to make sure that you feel very comfortable where you are before you go onto the next thing.

The tests were definitely helpful. I especially liked the fact that tests were done in a granular manner at each step along the tutorial, so that I felt confident throughout the tutorial.
I struggled with the hardware part, I was not that familiar with it. It’s kind of like, I am following the instructions but *I am just not sure if I am doing it the right way.*

Did a good job of guiding me to successful completion, but *did not really educate me on what I was doing.*
future work

1. tutorial authorship interfaces
2. activity review dashboards
3. skill-aware dynamic tutorials
4. supporting formal learning
acknowledgments

Thanks to my collaborators, the Autodesk User Interface Research Group, and David Mellis.
ElectroTutor is a novel system that integrates tests into physical computing tutorials. Novices can benefit from incremental tests in physical computing tutorials.

Incremental tests aid novices in problem scoping, and preliminary work points towards the potential for improved tutorial completion rates.

Jeremy Warner  
Ben Lafreniere  
George Fitzmaurice  
Tovi Grossman

Thanks!


@jeremywrnr  
jwrnr@berkeley.edu