Send a Tweet to Your Office Door

And let your coworkers know what you’re up to

By ERICO GUZZO  /  JUNE 2011

Here at IEEE Spectrum, staffers routinely put Post-It notes on their doors and cubicles to let colleagues know they're out. But you can't slap a note on your door if you decide to work at home at the last minute—which, as I learned this winter, happens quite often when you have a very pregnant wife about to go into labor any instant.

That's why I set out to replace those yellow sheets of self-adhesive stationery with something less, uh, analog—something that would allow me to post the updates electronically.

An idea then popped up in my head: Twitter for my door.

It turns out to be a simple DIY project that an experienced hobbyist can complete in a few hours. Or if you’re me and this is your first serious hardware project, it might take you a couple of months and nearly drive you insane.

It worked out in the end. Now, when I’m home—or actually anywhere with my phone—I can send a tweet to a small LCD that hangs by my door, thus keeping my coworkers informed of my whereabouts. The LCD also shows current weather conditions in New York City; I thought my colleagues would appreciate this value-added service.

The hardware setup is fairly straightforward. The main component is an Arduino, the popular microcontroller for DIY projects. I used a Duemilanove model (US $30). You’ll also need an accessory board called the Ethernet Shield ($45), which allows the Arduino to connect to the Internet. For the LCD, I used a Sparkfun black-on-green 5-volt model with four lines of 20 characters each ($30), which I connected to the Arduino with a thin ribbon wire ($5).

An Ethernet cable plugs into the shield, which attaches directly to the Arduino. Three wires connect the Arduino to the LCD—one for data bits, transmitted serially, the other two for power and ground.

It would have been great to have the device get its Internet access wirelessly, but that would have involved more complicated hardware and frequent battery replacements. So I drilled a small hole in the wall and ran a cable, hoping that the office manager wouldn’t notice (or read this article!).

The idea is, you post the tweet and the Arduino goes online periodically, fetches the message, and displays it on the LCD. Combining the Arduino and Twitter wasn’t a new idea. With a Google search I found similar projects, including one to send a tweet every time you flush your toilet.

With other people’s code as my starting point, soon my tweets were showing up on the little display. Nice. Except that two days later it stopped working. I pressed the Arduino’s reset button and it came back to life—until the next time. It would work for a day or two and then freeze. As it turned out, making my device work wasn’t very difficult. The hard part was making it work reliably.

A new Google search turned up fellow DIYers with the same problem. They were blaming the Ethernet Shield’s software stack. Something about data transmissions not closing properly and preventing new ones from closing.
An Arduino with an Ethernet Shield retrieves the updates from Twitter; the device gets a mechanical tweak; a hole in the wall is needed for Internet access.

One way around this issue was to periodically reset the shield. But the problem is that the reset pins of the Arduino and the Ethernet Shield are connected. That means if you use the Arduino to transmit a reset pulse to the shield, it will reset itself midway through the transmission, a process that can damage the chips.

The inelegant but effective fix was simple and decidedly mechanical: Bend the metal pin from the Ethernet Shield so it doesn’t connect to the Arduino reset pin and program the Arduino to reset the shield before it retrieves a tweet.

The device started working again…until it didn’t. For days I looked at my hardware and software and couldn’t figure out what had happened. I thought it might’ve been a curse from the Post-it gods, but then I read a news story that revealed the truth.

Sometime last August, Twitter changed its authentication method. The Arduino code to access my door’s Twitter account was outdated. Twitter’s new authentication method, called OAuth, is much more secure but not simple to implement, at least not for an inexperienced programmer like me.

It was back to Google, where I soon found a way to do the new Twitter authentication using PHP, a popular programming language for Web sites. That meant shifting the authentication from the Arduino to the Web.

I wrote a PHP program that retrieves the tweet and stores it in an XML text file. The PHP code runs on a Web hosting service I use (companies like Go Daddy, DreamHost, and others offer similar PHP capabilities). Using a scheduling tool called cron, I set the PHP program to run every 5 minutes.

So here’s how the system now works: After I post a tweet, the PHP program retrieves it from Twitter and puts it into an XML file on a Web server. Then the Arduino in my office fetches the XML file, parses it to extract the tweet, and sends it to the LCD. The PHP program also retrieves weather information from the Web and puts the data into the same XML.

If you’re considering replicating this project, here are some improvements you could try. First, implement the OAuth authentication in the Arduino itself, getting rid of the PHP component, the XML, and the Web server. (A public Twitter account would be even simpler, skipping the OAuth authentication entirely. But that’s not a good idea if you want to leave notes with phone numbers and other information you don’t want to be public.)

Second improvement: Implement some form of error detection that keeps garbled data from the LCD. Third, use a fancier, color LCD to format the message, weather, and other information in a better-looking way to impress your coworkers (and boss).

I created the sketch (as Arduino C language programs are called) using version 18 of the software; the Arduino maintainers have since revamped its Ethernet library, and I haven’t tested my sketch with the new versions.

By the way, you don’t have to use Twitter to convey your message. The Arduino can just as easily grab e-mail from Gmail or another POP account. I used Twitter because the office network blocks access to e-mail servers, and anyway, messaging from a phone is easy and feels cool.

As I write this, the system has been working uninterruptedly for over three months. Now excuse me, I need to tweet my door.