I finally started working in earnest on a new electronics project today. It was inspired by a discussion with one of my neighbors several months ago about forgetting to close the garage door and cases where the door seemed to be closing, but it reopened after driving away. Most people I've talked with have similar stories. Not only is leaving the garage door open responsible for causing increased utility usage in neighborhoods like mine with attached garages, but it also leaves your house a wide open target for theft. A few months ago I got a Raspberry Pi embedded computer system, and I figured this would be a fun project to learn about lots of new things that I only have minimal experience with: <ul> Relays and sensors Web applications and security Android application development Python script language Today I got the basic part of the project up and running. I'm able to remotely run a simple script on the Raspberry Pi that can open or close my garage door. I started with the basic Raspberry Pi board and also purchased the following pieces: EDIMax EW-7811Un USB Wireless Network Adapter: This is a very small, low powered WiFi adpater that plugs into one of the Raspberry Pi's USB sockets so I don't have to run an Ethernet wire out to my garage. It turned out to be surprisingly cheap and easy to set up. SainSmart 4-channel Relay Module: If you're not familiar with what relays are, They're basically switches that can be controlled by the outputs of the Raspberry Pi, but also serve to isolate the higher power/voltage requirements of the external devices from the lower powered and fairly delicate logic on the Raspberry Pi board. I will only need one channel for the garage door project, but I have a couple other ideas for things that can be controlled using the other channels. Wiring the relay to the Raspberry Pi board was very straightforward. I removed the jumper from the relay board that connects Vcc and JDVcc together since the relay requires 5V on JDVcc, but the Raspberry Pi only has 3.3V tolerant IO pads. Technically, Vcc is specified at 5V for the relays also, but they seem to function just fine from the 3.3V IOs. It was just a matter of wiring JDVcc to the 5V pin, Vcc to the 3.3V pin, IN1 to a GPIO pin (I chose GPIO17, pin 11, for no particular reason), and Ground to Ground. Right now I have alligator clip test leads for all these connections, but will make the wiring more durable and permanent later on in the project. &#65533; The next part was figuring out how to activate the garage door opener. It wasn't documented in my garage door opener's reference manual, but it turns out that most garage doors are operated simply by shorting to the two wires together when the button is pressed. That made wiring the relay to the garage door opener switch very easy. I just connected a wire to each of the terminals on the back of the wall-mounted garage door opener and connected them across the normally-open inputs of the relay. It's not pretty yet, but I'll clean it up once I get all the functionality implemented. Now that all the wiring was done, I needed to control the GPIO of the Raspberry Pi to turn the relay on and off to activate the garage door opener. It turns out that the Linux distributions that are available from Raspberry Pi have a nice GPIO library pre-installed and it's a piece of

cake to write a simple Python script to activate the GPIO pins. #!usr/bin/env python import RPi.GPIO as GPIO, time GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.OUT) GPIO.output(11, GPIO.HIGH)
time.sleep(1)
GPIO.output(11, GPIO.LOW) GPIO.cleanup() Really, that's it! This configures the GPIO library to address the GPIOs by pin number on the board (I've connected the relay to pin 11), sets it up as an output, then turns it on for 1 second and turns it back off. ﷺThis results in the garage door opener behaving the same as if you had pushed the button for 1 second and then released it. The 1 second wait seems to work well with my opener, but might vary for others. Right now I can SSH (secure remote terminal) to the Raspberry Pi from my phone or my computer and control the garage door from anywhere I have internet access. ﷺ This is just the starting point of the project. I still have a few goals to accomplish before I'm done: ﷺ <ul> Add a couple of sensors to the garage door so the Raspberry Pi can determine if it is open or closed Make an easy Android App front-end that can operate the door and show its current state Send an alert notification if the door has been open for longer than a pre-configured amount of time Figure out how to make all this network communication secure Any other ideas on fun features that could be added?{jcomments on}