Scraptcha: Trash Sorting for Good Fun

Build your own trash sorter game.

By: Matt Staniszewski

Is it trash? Recycling? A new species? Whatever I'm throwing away, I am always baffled by the different trash bins at the coffee shop, my lunch hangout or at the office. So I strapped a Raspberry Pi to each bin with a camera and created an app to help people sort trash while making a game. Scraptcha was born!

This idea sprung from my continous confusion about how to properly throw something out. Having moved up and down the west coast, the rules are different everywhere, even within cities. Paper goes in the slightly darker blue bin. Break down all cardboard. Only plastics 2 – 7, please. I love the idea of recycling, I'm just really bad at keeping track of the rules. We've all been there: you've just finished your coffee at Starbucks when you walk up to throw away your cup and you're confronted with three bins: trash, recycling and compost. Is the lid recycling? What about the cup, it's soiled so does that mean it's trash? Is there a plastic lining in the cup or can I compost it? Sometimes, a helpful picture guide is there to save the day, but more often you're your own.

That's why I decided to put my embedded skills to the test and combined a Raspberry Pi with some cheap hardware and a little javascript app to create Scraptcha, the trash-sorting, captcha-identifying device that helps you figure out how to sort trash and even make a game out it.

HOW SCRAPTCHA WORKS

Scraptcha uses an off-the-shelf USB webcam combined with a Raspberry Pi and some custom software to allow you to take a picture with the click of a button. Javascript running on the Pi connects to a central server called a manager, which allows you to connect to Scraptcha via an app. More on the app interface later.

Scraptcha has two modes, Capture and Gaming Mode. In the first mode, a request is made from the app to take a picture. This sends a signal to the Pi via Wi-Fi, which sends a command to take a picture using the webcam. The picture is then analyzed with a super-intelligent guestimation system where it decides whether the item is trash, recycling or compost. To accomplish this, I used OpenCV to analyze each pixel in the image. Currently, it guesses based on color as follows:

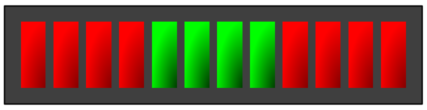
RED	= Trash
GREEN	= Compost
BLUE	= Recycling

A dumb guess for now, but users can help out with that in Gaming Mode. In the future, the software on the Pi will use large sets of verified data in combination with a support vector machine (SVM) to make a more intelligent guess. These are pretty cool algorithms, but they take a bit of time to implement. If you're interested in giving SVMs a shot, see the short tutorial at docs.opencv.org/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html along with some pre-made datasets for testing your new SVM (archive.ics.uci.edu/ml/datasets.html and cosmal.ucsd.edu/~bmcfee/data/eharmony.html)

MATERIALS AND TOOLS

- **Raspberry Pi Check out Element14, Sparkfun or Adafruit for one**
- **USB Power Supply Adafruit #501**
- **Mini USB Cable Adafruit #592**
- **SD Card 2GB or larger…a microSD card with a converter is cool too**
- **Wireless-N USB Adapter Airlink #AWLL5088 (Amazon)**
- **USB Webcam Logitech C310 or another Pi-compatible one (see elinux.org/RPi_VerifiedPeripherals)**
- **Pi T-Cobbler Adafruit #1105 (or the regular cobbler in this article, #914)**
- **GPIO Ribbon Cable Adafruit #862**
- **2x16 Character LCD Adafruit #181**
- **I2C/SPI LCD Backpack Adafruit #292**
- **LED Bargraph Adafruit #459**
- **Shift Register Adafruit #450**
- **PNP BJT Transistors 2N2907s (3) on Adafruit #522**
- **Resistors 100Ω (3)**
- **USB keyboard/mouse**
- **HDMI monitor cable and hopefully a monitor it can plug into**
- **Ethernet cable**
- **4-port USB Hub (self-powered)**
- **Breadboard (2)**
- **Soldering Iron**
- **Lead-free Solder**
- **Wire Cutters**
- **Wire Strippers**

The image and guess are stored in the manager's database and displayed in the app. An LCD connected to the Pi also displays the guess along with Scraptcha's status. The LED bar contains 12 red and green LEDs, which are used to designate which trash bin to throw the item into. Theoretically, this system would have three cameras and three sets of LEDs, one per bin. For the prototype, I only used one camera, but the LED bargraph displays the status for all three bins. The LEDs are grouped into three four-LED sets that display green for the correct bin and red for incorrect. The LED order from left to right is trash, recycling and compost. For example, if it guessed recycling, you would see the following:



Gaming Mode is where we can have a bit of fun. Remember all those really dumb guesses? Well now's your chance to tell the computer what's up. In this mode, you will be given one of the images that was taken by Scraptcha which hasn't been verified yet. You get to tell Scraptcha what's what, selecting trash, recycling or compost, certifying the item in question and storing it back into database.

Over time, once a large database of verified images is stored up, this database can be used with the SVM (if implemented) to train Scraptcha to do some more complex object detection and guessing.

Ever heard of captchas? You know, those random garbled alpha-numeric security passcodes on websites? Well, an even cooler aspect of Scraptcha is that once you build up this dataset of verified data, it can be used as a set of "Scraptchas" to authenticate people for websites. Pretty neat, huh?

1. SETUP THE HARDWARE

Construct the circuit as shown in Figure A and B below. These images were taken using the open-source tool Fritzing (frtizing.org) which is great for designing hobby projects. It includes a breadboard view along with a schematic and PCB layout tab, and any changes done on one screen will show up on the other. Most of the parts are available in libraries you can find online, including Adafruit, which can be found over at github.com/adafruit/Fritzing-Library.

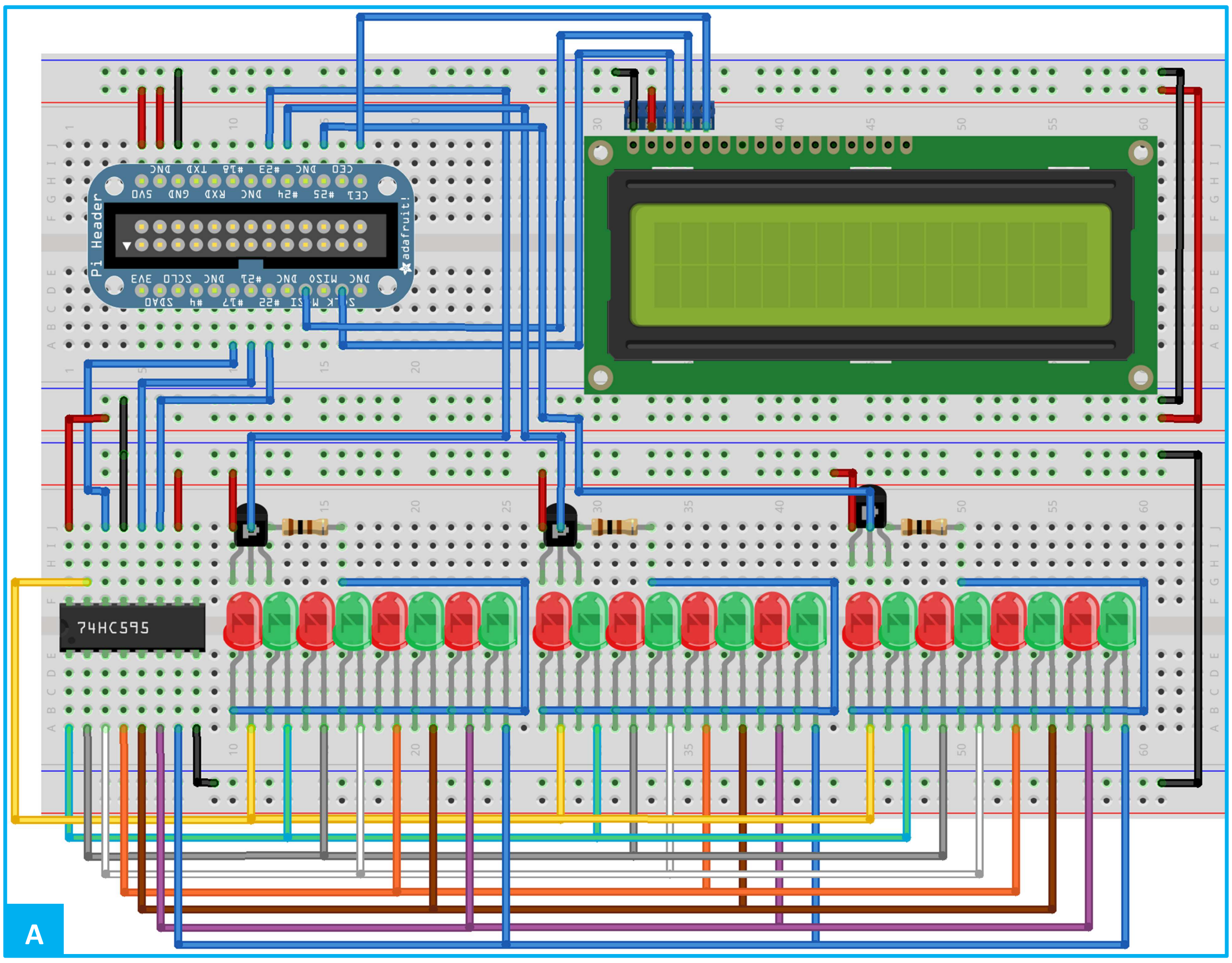
One important note, both the breadboard and schematic drawings don't show the LCD hooked up to anything; the SPI signals just go to a five-pin terminal. This is because I'm actually using Adafruit's I2C/SPI LCD Backpack (adafruit.com/products/292), which isn't in the Fritzing libraries. The LCD is actually connected through the backpack to the five-pin terminal block. For info on hooking this up properly, pop on over to learn.adafruit.com/i2c-spi-lcd-backpack.

The LED bargraph has been shown here as individual LEDs because, again, there's no Fritzing symbol for it. The transistor/shift register setup is borrowed from Adafruit (adafruit.com/products/459).

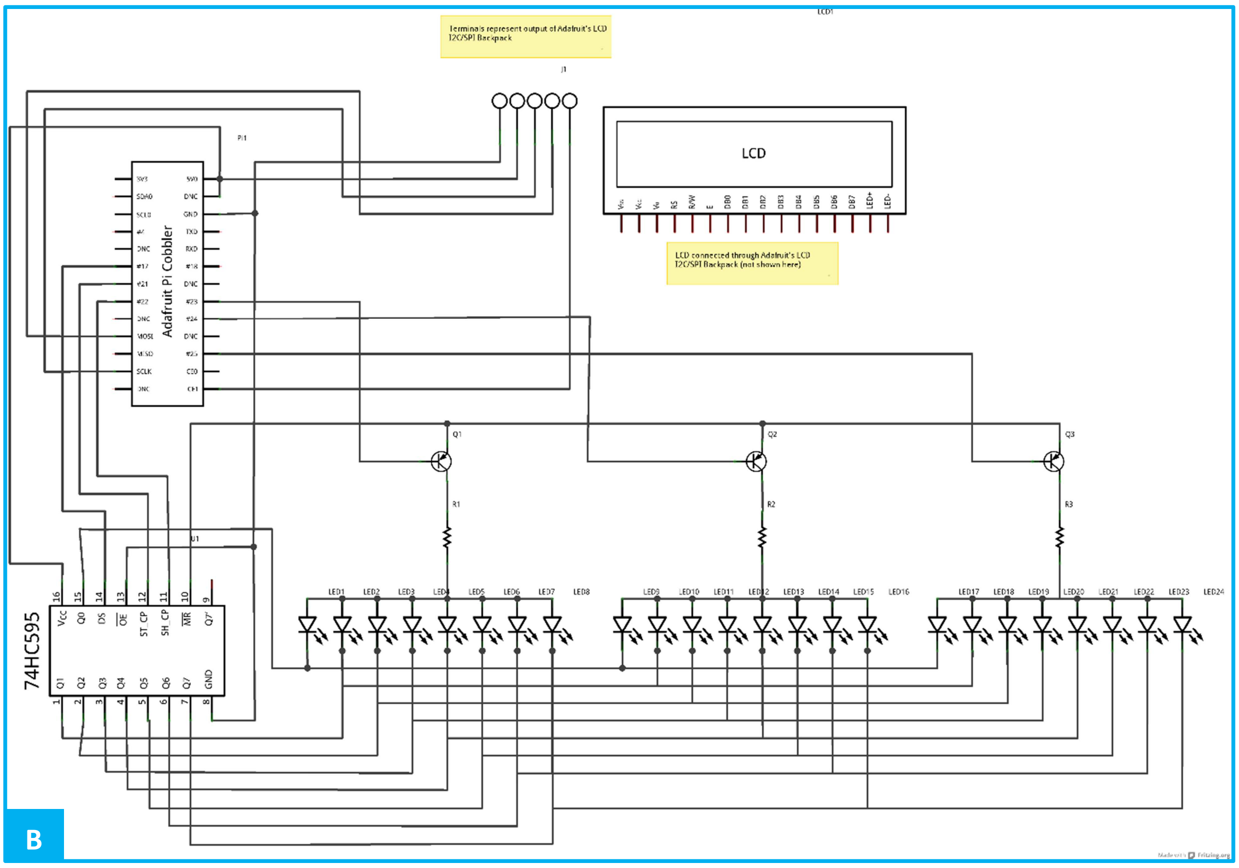
I show the regular Pi Cobbler in the breadboard view. In the actual circuit, I used a T-Cobbler. Both have the same functionality, just a different board design. The Pi Cobbler is more compact while the T-Cobbler is a bit more accessible.

2. HOOK UP YOUR PI

Now that we have the hardware set up, let's work on getting your Pi fired up. The



A



B

Raspberry Pi folks have a great quick start guide that provides all you need for hooking up the Pi and then installing Raspian. Check it out over at raspberrypi.org/quick-start-guide.

In addition to the guide, be sure to plug in your USB hub to the Pi and power it up with your wireless card in addition to the Ethernet cable. This is so that you can test Wi-Fi setup before you try to use SSH.

Speaking of SSH (openssh.org), that's how we'll connect to the Pi remotely once it's all setup. That way, we don't need to be lugging a monitor, keyboard and mouse around when we deploy Scraptcha. When you first start the Pi, you'll enter raspi-config. Be sure to enable SSH before saving your configuration and restarting. As always, eLinux has the goods (elinux.org/RPi_raspi-config).

For Wi-Fi, consult the Raspberry Pi forums for setup. There's a lot of good information about setting up and configuring wpa_supplicant and the other files needed to get you up and running. Be sure to test it before you shut down as well, since we won't have a monitor after this.

Next, we'll need to grab Node.Js. Node is an asynchronous version of Javascript, which means it can run multiple commands simultaneously. It's used along with some C code to run Scraptcha's hardware and communicate with the manager. Grab the latest precompiled version to save yourself some time (I recommend 0.8.16, which follows this guide): raspberrypi.org/phpBB3/viewtopic.php?f=34&t=24130.

Alrighty, more packages. This time we're going with apt-get, so once you're on the net, do a `sudo apt-get update`. Follow that up with a fresh install of npm, git and libopencv-dev using `sudo apt-get install`. Last but not least, install node gyp with `npm install -g node-gyp`. By default, SPI is disabled on the Pi, so let's fix that by going into

`/etc/modprobe.d/raspi-blacklist.conf` and putting a pound sign (#) next to `blacklist spi-bcm2708` and saving.

Since we're connecting via SSH to the Pi, it would be good to know just what address it has. There's a few options to do this. First, you can try the Python script located at github.com/stantheman286/Scraptcha/blob/master/ipemailnotification.py, which will email you the Pi's IP address every time you turn it on. Just add your email address and password to line 33. Another option is getting a free version of a dynamic DNS client such as No-IP (no-ip.com).

Wouldn't be nice if the Pi would just start up the Scraptcha code on boot? Install forever via `npm install –g forever` to do just that.

Both forever and the IP notification service require you to add the appropriate command to your `/etc/rc.local` file so that it starts with the Pi. Use `python <path-to-script>/ipemailnotification.py` for the notifier and see stuffaboutcode.com/2012/06/raspberry-pi-access-from-internet-using.html for setting up No-IP. We'll cover the forever command in the software section. It's a good idea to add a `sleep 1m` command before your commands in `/etc/rc.local` to make sure the network is ready when the commands are run.

Before we reboot the system, we should probably test out and understand SSH. Grab a tool like PuTTY (chiark.greenend.org.uk/~sgtatham/putty), install it onto a computer (not your Pi) and connect to port 22 of the IP address of your Pi with your username and password. If you got a command prompt, congrats, you're in! If not, check your network configuration, router and raspi-config settings.

Whew, your Pi is set! Now let's safely shutdown (because we should be safe) by typing `shutdown –h now`. Always turn your Pi off in this way; unplugging the power without shutting down may corrupt the SD card!

3. CONNECT THE HARDWARE TO THE PI

Now that we have the hardware in one corner and the Pi in the other, it's time to marry the two together.

With the Pi shutdown and unplugged (wait a good amount a time after shutting down to guarantee it's off), disconnect the keyboard, mouse and monitor. If you got Wi-Fi up and running, you can lose the Ethernet cable too.

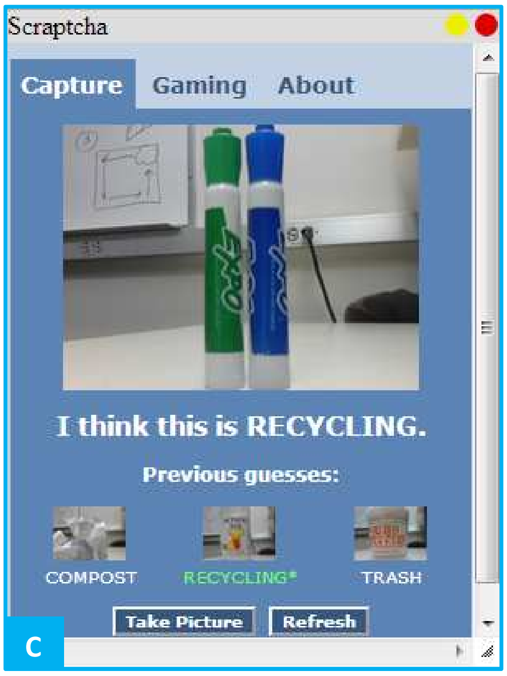
Connect the Pi to your breadboard using the GPIO ribbon cable and the Pi cobbler. Connect the webcam to the USB hub, which should be connected to the Pi along with your wireless card at this point. Make sure the hub is connected to the Pi and powered on and then turn on the Pi. Now you're ready for some code!

4. CONFIGURE THE SOFTWARE

Back to software land. With the Pi powered on, login via SSH as we did before, create a new directory where you'd like to put your code and use Git (read up at help.github.com) to clone both github.com/stantheman286/scraptcha and github.com/stantheman286/minimal542project. Be sure to grab the `v0.2_main` branch of the `minimal542project` repo and place the `minimal542project` folder within the `scraptcha` folder.

If you have a different webcam from the Logitech C310 or are using a version of Node other than 0.8.16, we need to configure the hardware to use it. Enter the `scraptcha/c` directory and run `node-gyp configure` to configure the driver. Next, edit the `Makefile` located in this directory to include CFLAGS and LDFLAGS as explained in opencv.willowgarage.com/wiki/CompileOpenCVUsingLinux. Finally, run `node-gyp build` to complete webcam setup.

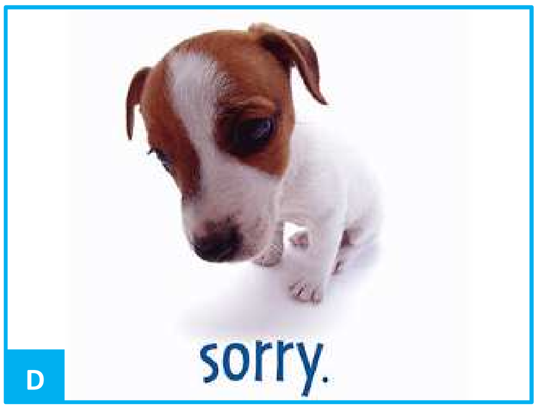
Same goes for the SPI driver all you non-Node 0.8.16 users. Enter the `scraptcha/c/nodeSPI` directory and run `node-`



gyp configure and node-gyp build. No intermediate step this time.

Now that we have all the hardware configured, let's take it for a walk. Run `node scraptcha/node/test.js` to run a quick LED and LCD test followed by taking a picture and making a guess. If all three happen, congrats, Scraptcha is ready for primetime!

Now, we just need to tell the device who it's talking to. The manager communicates with the database as well as handles the commands between the app and the device. You can edit the manager name and port settings in lines 100 – 101 of `scraptcha.js`. By default, Scraptcha is set up to use the bioturk.ee server at UW.



Give Scraptcha a try by running `node scraptcha.js` in the `minimal542project` folder and making sure it shows up in your manager's dashboard. Before we get too carried away with playing, let's set up Forever. Stop the `scraptcha.js` program with CTRL+C and then open up `/etc/rc.local` again and add the following lines to run the Scraptcha code at boot:

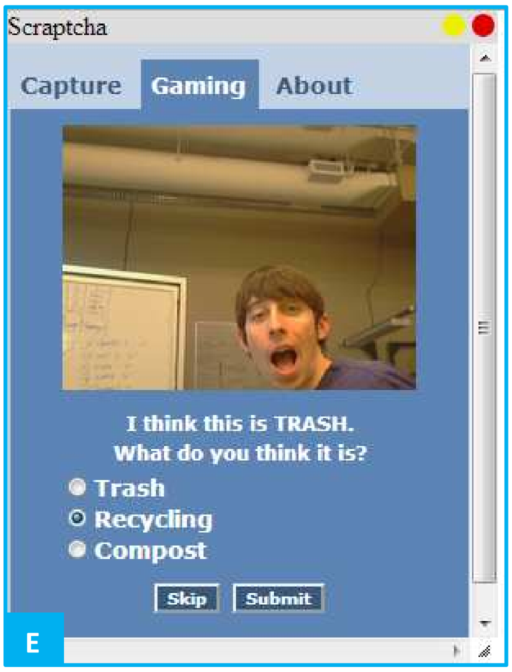
```
cd <path-to-
code>/scraptcha/minimal542project
su pi -c 'sudo forever start -a -l
forever.log -o out.log -e err.log
scraptcha.js'
```

Save and do a `sudo reboot` to restart the Pi. Congrats, you're ready for the big show!

5. TAKE PICTURES OF TRASH AND PLAY!

Fire up Google Chrome and navigate to your manager's website (bioturk if default). After your Pi has rebooted, you should see a 'Launch' button for Scraptcha. Click it start the app.

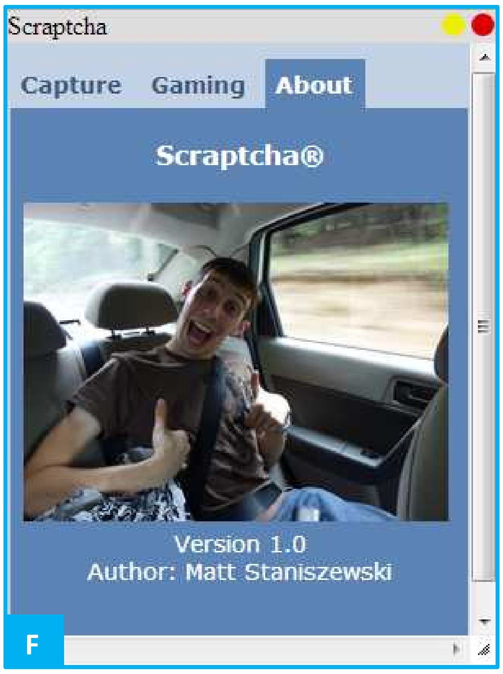
Scraptcha has two modes, Capture Mode and Gaming Mode, which are displayed along the top with About. Capture Mode (Figure C)



allows you to take pictures of trash using the Scraptcha hardware by clicking ‘Take Picture’. The latest image will appear as the large picture with its guess while the last three pictures and guesses will appear in the tray below. The screen updates every 10 seconds, but if you’re in a rush, click ‘Refresh’. Guesses in white are those made by Scraptcha; those in green with an asterisk have been verified by users in Gaming Mode. Every time you take a picture, the LCD and LEDs will also be updated with the latest status and guess.

Switching to Gaming Mode (Figure E), you’ll be presented with the oldest, unverified image in the database. You can select whether an item is trash, recycling or compost and hit ‘Submit’ to record your choice to the database and verify the image. You can also choose to skip with the ‘Skip’ button. Scraptcha will keep giving you photos to verify until you have verified or skipped them all, which will then give you the ‘Sorry Puppy’ (Figure D) and restart from the beginning. You can always switch back to Capture Mode to take more pictures.

Last but not least is the About tab (Figure F), which contains an extremely flattering photo of yours truly, the creator of Scraptcha, along with some version info. Enjoy the picture, it’s a classic.



F