How to set up the whole damn thing

Step 0: Prerequisites

- One coffee maker
- One relay
- One piece of hardware running Java and Python
- One remote server running Java and MySQL

Step 1: Get the hardware set up

Get a relay and hook it up to the coffee maker. We used a Mr. Coffee Coffee Maker, but it doesn't really matter. Find the power, and use a relay to determine if it gets power or not.

Picture of our coffee maker if you like those:
https://photos.app.goo.gl/Arefk2g7ZicIbPQs1

Once you've got that all set up, hook it up to a Pi. The pins don't really matter, but I used pin 18 to talk to the relay and a breakout board. If that's not your thing, change it.

Hook it up to some power, we used a 270 ohm resistor for the relay we used. Check the specs for the relay you used to see what you need to use.

Hook it all up and your hardware is good to go. Putting the coffee maker back together will probably be the most annoying part.

Here's a picture if you like pictures
https://photos.app.goo.gl/KinpAYphhubshhrh8

Step 2: Set up your RPi

This is the thing that will physically interact with your relay. There are two components to this.

The Coffee-Script https://github.com/IOTCoffee/Coffee-script and the listener server: https://github.com/IOTCoffee/Remote-coffee-listener-server

Build the remote listener with `./gradlew uploadArchives`, and it will build your remote listener into a jar file called: `Taco Weather Simple-1.0.jar` DON'T ASK WHY IT'S CALLED THAT. I renamed it to `coffee.jar`

Copy the jar file and covfefe.py to your Pi.

Run covfefe.py `./covfefe.py` then get its pid.

Run the jar file `sudo java -jar coffee.jar <COVFEFE.PY PID>`

Expose your Pi to the internet and get the URL or IP of it. You will need this later to interact with your coffee maker.

Step 3: Set up your remote server

Make sure your remote server runs Java. Also make sure port 80 is exposed to the internet.

Get the last bit of server software here:
https://github.com/IOTCoffee/IOT-Coffee-Server the IOT-Coffee-Server

Build the server with `./gradlew distTar` this will make a tar of all of the

server components you need. The tar will be located in a place that looks
something like this: `build/distributions/Pillserver-1.0-SNAPSHOT.tar`
DON'T ASK WHY IT'S CALLED PILLSERVER. I renamed it to `coffee.tar`

Time to get MySQL set up. The webserver expects a user named `root`, and a
password named `root`. Insecure? Sure. If you don't like it, change it. It's in
the file `src/main/kotlin/Main.kt`

Make these tables:

```
mysql> show tables;
+--------------------+
| Tables_in_iotcoffee |
+--------------------+
| cmaker             |
| logins             |
| users              |
+--------------------+
3 rows in set (0.00 sec)

mysql> describe cmaker;
+-------+---------------+------+-----+---------+-------+
| Field | Type          | Null | Key | Default | Extra |
+-------+---------------+------+-----+---------+-------+
| email | varchar(255)  | YES  |     | NULL    |       |
| url   | varchar(2048) | YES  |     | NULL    |       |
| name  | varchar(255)  | YES  |     | NULL    |       |
+-------+---------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> describe logins;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| ID    | int(11)      | NO   | PRI | NULL    | auto_increment |
| email | varchar(255) | YES  |     | NULL    |                |
| token | varchar(255) | YES  |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> describe users;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| ID       | int(11)      | NO   | PRI | NULL    | auto_increment |
| email    | varchar(255) | YES  |     | NULL    |                |
| password | varchar(255) | YES  |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql>
```

Move the tar to your remote server and unzip it. Run the executable HERE:
`./Pillserver-1.0-SNAPSHOT/bin/Pillserver`

Take note of the URL where your server is located. You will need it later.

Congrats, your server is good to go.

Step 4: Getting the app ready

You need to change the URL of the app to point at the server you just made.
Yes you actually need to modify the code. Get over it.

Go to: `src/main/java/com/darrienglasser/iotcoffee/Consts.kt`

You'll get something like this:

```
package com.darrienglasser.iotcoffee

val BASE_URL = "http://54.82.246.46"

```

Change the BASE_URL to whatever your server is.
DON'T EVEN THINK ABOUT ADDING A / AT THE END. DON'T I CAN SEE IT YOU WANT TO DO
IT IT'S A BAD IDEA.

Go back to the root directory of the app and type `./gradlew assembleDebug` this
will build your app. This is the directory you'll find it.

`app/build/outputs/apk/debug/app-debug.apk`

Make sure adb debugging is activated on your phone. Install with
`adb install app-debug.apk`

Register and brew and you should be good to go!