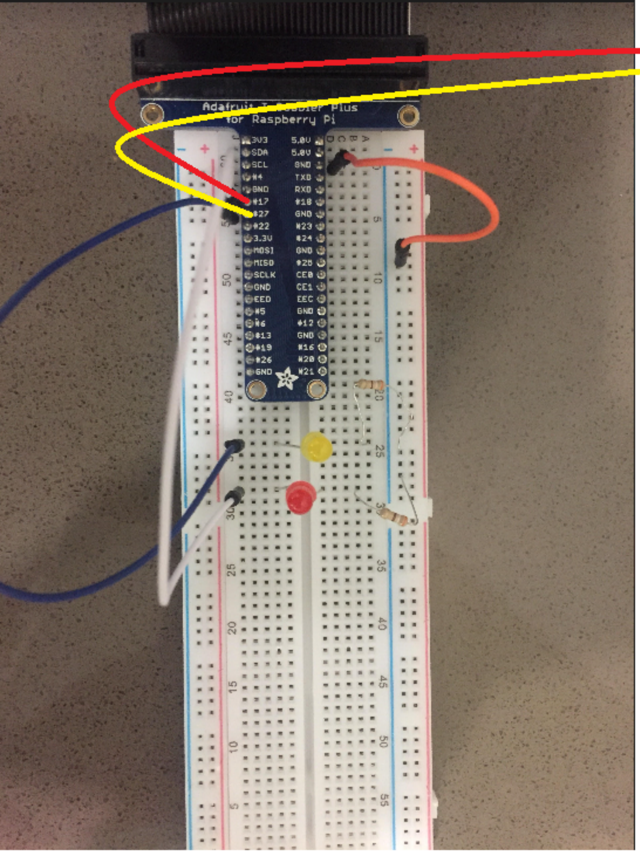
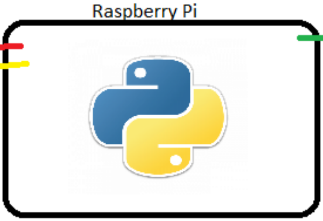
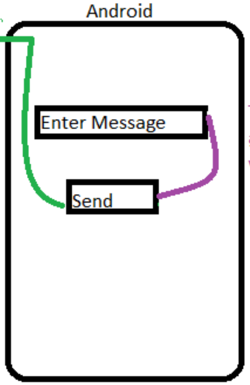




User sends message to pi to be translated to morse code.



Text is stored in message and sent over bluetooth when button is pressed

Pi uses a dictionary of pre set letter-morse code translations and replaces each letter in list with the dots and dashes the cooresspond

```python
codes = {'A': '.-', 'B': '-...', 'C': '-.-.',
         'D': '-..', 'E': '.', 'F': '..-.',
         'G': '--.', 'H': '....', 'I': '..',
         'J': '.---', 'K': '-.-', 'L': '.-..',
         'M': '--', 'N': '-.', 'O': '---',
         'P': '.--.', 'Q': '--.-', 'R': '.-.',
         'S': '...', 'T': '-', 'U': '..-',
         'V': '...-', 'W': '.--', 'X': '-..-',
         'Y': '-.--', 'Z': '--..', '0': '-----',
         '1': '.----', '2': '..---', '3': '...--',
         '4': '....-', '5': '.....', '6': '-....',
         '7': '--...', '8': '---..', '9': '----.',
         ' ': '='}
```

Pi sends gpio high command on a timer through pin 11, making the red led either blink or light up longer to simulate a dot or dash

```python
for char in range(len(data)):
    time.sleep(1)
    if data[char] == '.':
        print "Dot"
        GPIO.output(11,GPIO.HIGH)
        time.sleep(.2)
        GPIO.output(11,GPIO.LOW)

    elif data[char] == '-':
        print "Dash"
        GPIO.output(11,GPIO.HIGH)
        time.sleep(1.5)
        GPIO.output(11,GPIO.LOW)
```

Pi sends a gpio high command to the yellow led whenevr there is a space in the text to make translation easier

```python
elif data[char] == '=':
    print "Space"
    GPIO.output(13,GPIO.HIGH)
    time.sleep(.5)
    GPIO.output(13,GPIO.LOW)
```

Phone connects to raspberry pi via bluetooth to send data. Program will not run until python accepts the socket from the android application and they are synchonized on the same UUID

```python
server_sock = BluetoothSocket(RFCOMM)
server_sock.bind(("",PORT_ANY))
server_sock.listen(1)
uuid = "00001101-0000-1000-8000-00805f9b34fc"
advertise_service(server_sock, "ChristinesPiServer",
                  service_id = uuid,
                  service_classes = [uuid, SERIAL_PORT_CLASS],
                  profiles = [SERIAL_PORT_PROFILE],
                 )

while True:
    print "Waiting for connection on RFCOMM"
    client_sock, client_info = server_sock.accept()

    print "Accepted connection from: ", client_info
```