

Cliente para acesso a sonda

Universidade de Aveiro

Ricardo Ermida, Rodrigo Santos



Cliente para acesso a sonda

DETI - Universidade de Aveiro

Universidade de Aveiro

Ricardo Ermida, Rodrigo Santos

(89187) ricardoermida@ua.pt, (89180) rodrigo.l.silva.santos@ua.pt

20 de Abril de 2018

Resumo

O primeiro passo para o funcionamento da aplicação é a criação de um socket, através do qual é efetuada a conexão ao servidor **xcoa.av.it.pt**, de seguida é necessário enviar o comando **CONNECT** para o servidor de modo a receber o **Token** necessário para a comunicação com o servidor. Para iniciar a recolha dos dados de temperatura, vento e humidade, enviamos o comando **READ** seguido do token recebido anteriormente. Optámos por aguardar por 2 objetos com a informação de temperatura, vento e humidade, e à medida que eles são recebidos, são guardados num ficheiro Comma-Separated Values (CSV), por fim é feita a média dos valores da temperatura e é mostrado na linha de comandos um conselho sobre o tipo de roupa a levar consoante a temperatura.

Conteúdo

1	Introdução	1
2	Descrição e Testes	2
2.1	ligação á Sonda	2
2.2	Receção dos Objetos JSON	3
2.3	Resultados e Conselho	5
3	Conclusões	6

Capítulo 1

Introdução

O tema deste relatório será o código desenvolvido para o acesso remoto a uma sonda, de modo a obter dados sobre a temperatura, vento e humidade.

Será feita uma descrição e discussão acerca dos aspectos mais importantes do funcionamento da aplicação, com recurso a imagens da execução da aplicação e de alguns testes a algumas partes do código.

Este relatório está dividido em 3 capítulos.

Depois desta introdução, no Capítulo 2 é feita a descrição do funcionamento da aplicação, este capítulo encontra-se dividido em 3 subcapítulos no primeiro é descrita como é feita a ligação à sonda, no segundo como são recebidos e guardados os objetos JavaScript Object Notation (JSON), no terceiro subcapítulo apresentamos a nossa opção para o conselho é calculado. Por fim, no Capítulo 3 são apresentadas as conclusões retiradas após terminado o trabalho.

Capítulo 2

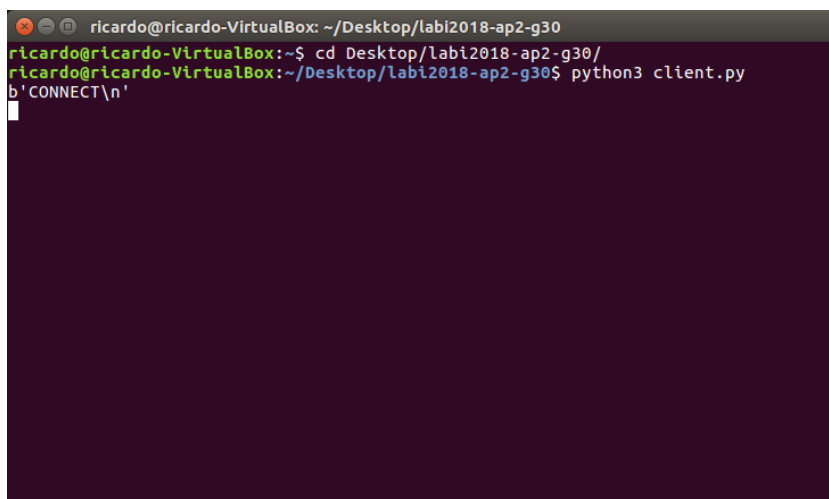
Descrição e Testes

Para que a aplicação funcionasse corretamente é necessário que o código desenvolvido esteja bem escrito, para sabermos se está ou não bem escrito é preciso realizar testes em diferentes partes do código.

2.1 ligação á Sonda

Quando executamos o programa criamos um socket através do qual entramos no server do xcoa. De seguida criamos uma variavel com o comando "CONNECT"codificamo-la e enviamos o comando para efetuar a ligação ao servidor. Nesta fase fizemos um teste de forma a garantir que a informação que queríamos enviar estava correta e consequentemente que o programa estava a funcionar corretamente, este teste não foi incluído no código final do trabalho.

A imagem a seguir, mostra o resultado desse teste, ou seja, o comando CONNECT codificado que é enviado para o servidor:



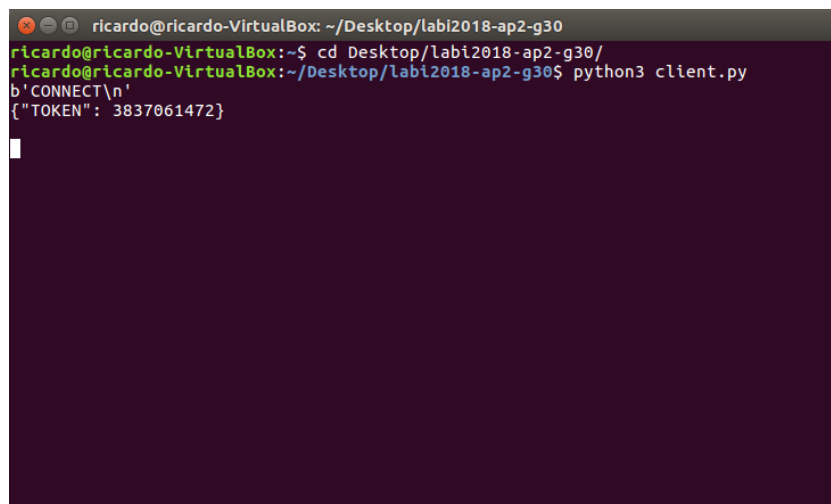
```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
```

Figura 2.1: Teste do comando CONNECT

2.2 Receção dos Objetos JSON

A comunicação entre a sonda e o cliente é feita através de comandos enviados pelo cliente e objetos do tipo JSON enviados pela sonda. Logo é necessário receber esses objetos e ler o seu conteúdo.

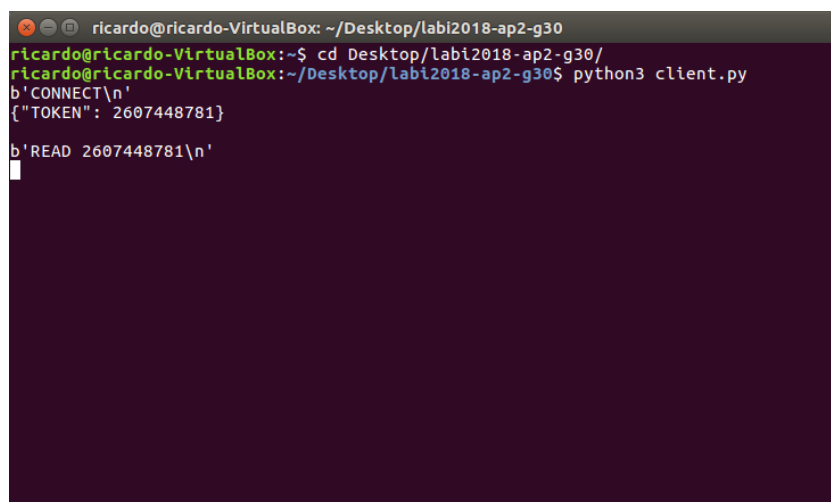
Depois da conexão á sonda estar feita é feita a recolha do objeto JSON com o token, nesta fase é testado se este objeto é recebido corretamente,

A terminal window titled 'ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30'. The user enters 'cd Desktop/labi2018-ap2-g30/' and then 'python3 client.py'. The output shows 'b'CONNECT\n'' followed by a JSON object: '{"TOKEN": 3837061472}'.

```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 3837061472}
```

Figura 2.2: Teste do token

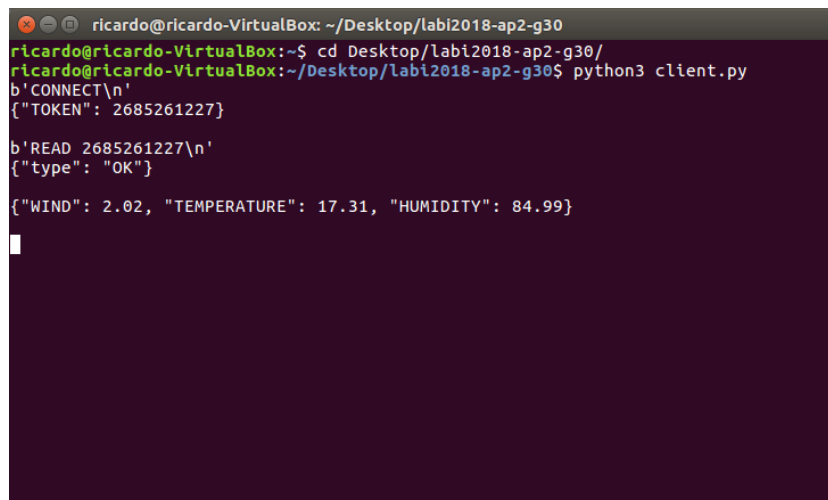
abrimos o objeto e criamos o próximo comando a enviar, comando READ seguido do token, o comando é codificado e enviado para o servidor, tal como no comando CONNECT foi testada também a codificação do comando READ,

A terminal window titled 'ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30'. The user enters 'cd Desktop/labi2018-ap2-g30/' and then 'python3 client.py'. The output shows 'b'CONNECT\n'' followed by a JSON object: '{"TOKEN": 2607448781}'. Then, the user enters 'python3 client.py' again, and the output shows 'b'READ 2607448781\n''.

```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 2607448781}
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'READ 2607448781\n'
```

Figura 2.3: Teste do comando READ

de seguida é recolhido e impresso no terminal o objeto de OK enviado pelo servidor. Após a receção do OK é recolhido o primeiro objeto com valores de vento, temperatura e humidade.

A terminal window titled 'ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30'. The user runs 'cd Desktop/labi2018-ap2-g30/' and then 'python3 client.py'. The output shows a successful connection with token 2685261227, followed by a 'READ' command and a JSON object: {"type": "OK"}, {"WIND": 2.02, "TEMPERATURE": 17.31, "HUMIDITY": 84.99}.

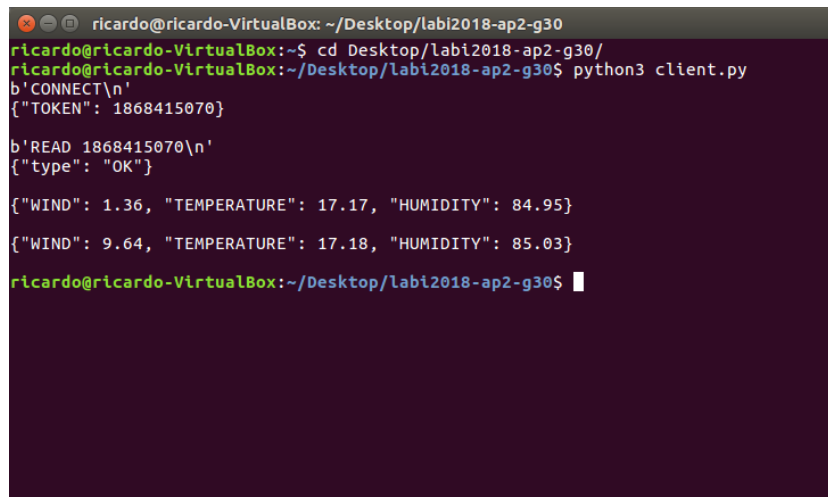
```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 2685261227}

b'READ 2685261227\n'
{"type": "OK"}

{"WIND": 2.02, "TEMPERATURE": 17.31, "HUMIDITY": 84.99}
```

Figura 2.4: Teste da receção do primeiro objeto do tipo JSON

Esses valores são guardados em variáveis, para depois serem armazenados num ficheiro CSV criado anteriormente, o mesmo acontece com o segundo objeto de valores.

A terminal window titled 'ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30'. The user runs 'cd Desktop/labi2018-ap2-g30/' and then 'python3 client.py'. The output shows a successful connection with token 1868415070, followed by a 'READ' command and two JSON objects: {"type": "OK"}, {"WIND": 1.36, "TEMPERATURE": 17.17, "HUMIDITY": 84.95}, {"WIND": 9.64, "TEMPERATURE": 17.18, "HUMIDITY": 85.03}.

```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 1868415070}

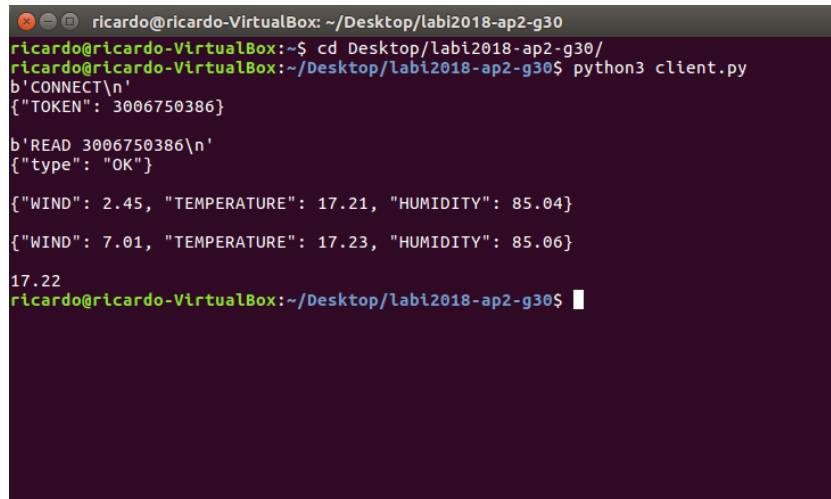
b'READ 1868415070\n'
{"type": "OK"}

{"WIND": 1.36, "TEMPERATURE": 17.17, "HUMIDITY": 84.95}
{"WIND": 9.64, "TEMPERATURE": 17.18, "HUMIDITY": 85.03}
```

Figura 2.5: Teste da receção do segundo objeto do tipo JSON

2.3 Resultados e Conselho

Nesta parte do trabalho fizemos a média das somas das temperaturas, de modo a conseguir informar o utilizador se estava calor, frio, ou temperatura amena, e o tipo de roupa que deve levar.



```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 3006750386}

b'READ 3006750386\n'
{"type": "OK"}

{"WIND": 2.45, "TEMPERATURE": 17.21, "HUMIDITY": 85.04}

{"WIND": 7.01, "TEMPERATURE": 17.23, "HUMIDITY": 85.06}

17.22
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$
```

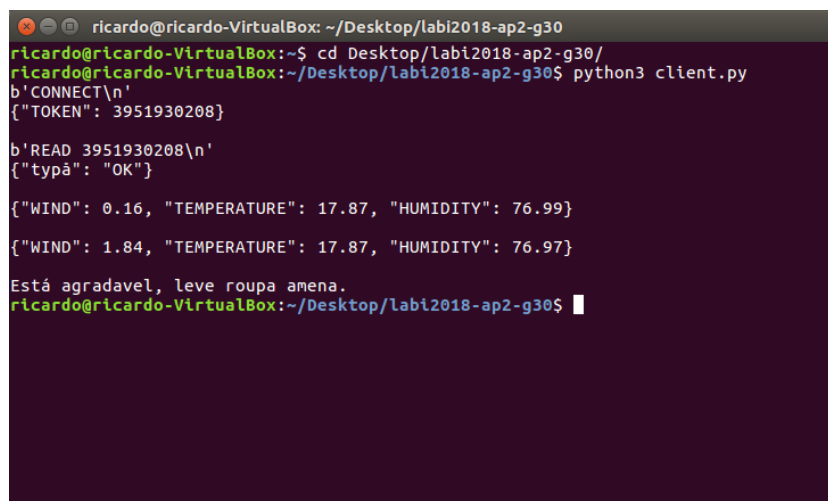
Figura 2.6: Teste do cálculo da media das temperaturas

Se a temperatura estiver abaixo dos 12 graus o utilizador recebe uma mensagem a dizer que está frio e que deve levar roupa quente, se a temperatura estiver a cima dos 12 graus e abaixo dos 20 graus o utilizador recebe uma mensagem a dizer que a temperatura está agradável e que deve levar roupa amena, se a temperatura estiver a cima dos 20 graus o utilizador recebe uma mensagem a dizer que está calor e que deve levar roupas mais leves.

Capítulo 3

Conclusões

Após todos os testes unitários executados anteriormente verificamos que o programa funciona corretamente, e tem o aspeto seguinte:



```
ricardo@ricardo-VirtualBox: ~/Desktop/labi2018-ap2-g30
ricardo@ricardo-VirtualBox:~$ cd Desktop/labi2018-ap2-g30/
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$ python3 client.py
b'CONNECT\n'
{"TOKEN": 3951930208}

b'READ 3951930208\n'
{"typâ": "OK"}

{"WIND": 0.16, "TEMPERATURE": 17.87, "HUMIDITY": 76.99}
{"WIND": 1.84, "TEMPERATURE": 17.87, "HUMIDITY": 76.97}

Está agradável, leve roupa amena.
ricardo@ricardo-VirtualBox:~/Desktop/labi2018-ap2-g30$
```

Figura 3.1: Programa executado com os testes

Para o código final do programa retiramos os testes feitos ao longo deste por questões de simplificação estética.

Com a realização deste projeto aprofundamos as nossas competências em programação na linguagem Python, aplicando esses conhecimentos numa situação real. Este trabalho de aprofundamento foi uma boa maneira para nos preparar para futuros projetos que possamos vir a estar envolvidos que envolvem Python e comunicação com servidores.

Contribuições dos autores

No desenvolvimento deste trabalho de aprofundamento, ambos os membros do grupo demonstraram o empenho necessário para a sua conclusão. O Ricardo Ermida executou os testes aos diferentes blocos de código, tendo o Rodrigo Santos desenvolvido o código necessário para a aplicação. O relatório foi feito em conjunto por ambos os membros do grupo. Ambos leram o relatório várias vezes para certificar de que não havia erros de qualquer espécie e para que fosse modificado, se necessário, algum do conteúdo. Assim o trabalho desenvolvido pelo Rodrigo Santos e pelo Ricardo Ermida foi de 50% cada um.

Acrónimos

CSV Comma-Separated Values

JSON JavaScript Object Notation