

Building new IoT Services Easily with Open Hardware and Lhings

José Antonio Lorenzo – Lyncos Technologies S. L.
José Pereda – Universidad de Valladolid

05th February 2014



Jfokus 2014, Stockholm

Who the heck are these guys??!!

José Antonio Lorenzo

- PhD in Physics
- Lyncos Technologies, Spain
- JavaEE, embedded
- <http://blog.lhings.com>
- @joanlofe

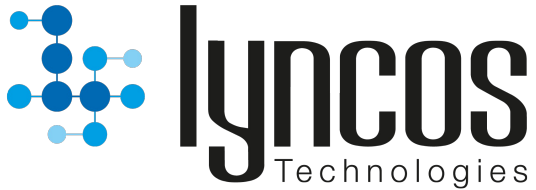


Who the heck are these guys??!!

José Pereda

- PhD in Industrial Engineering
- University of Valladolid, Spain
- JavaFX, Embedded
- <http://jperedadnr.blogspot.com>
- @JPeredaDnr





A startup based in Barcelona whose aim is to add value to products by connecting them to the Internet.



Lhings, a cloud platform to provide connectivity through the Internet to all kind of devices, is the main product of Lyncos.

www.lhings.com



Tasty Recipes

Recipe 12: Internet of things project

INGREDIENTS:

Some devices:

- 1 Arduino board
- 2 Raspberry Pi

Sensors:

- 1 thermometer
- 1 gas sensor
- 2 presence sensors

Actuators:

- 1 motor
- 3 relays

PREPARATION:

1. Design your circuits and build them.
2. Program your devices so they work as expected while taking care of low level programming issues.
3. Design your communications.
4. Program all your networking code.



- ✓ Security: mainly confidentiality and authentication
- ✓ Networking issues: firewalls, residential routers, push
- ✓ Scalability: how to manage hundreds of devices → message routing, connection management
- ✓ Reliability: quality of service, availability of the system



- ✓ Security: mainly confidentiality and authentication
- ✓ Networking issues: firewalls, residential routers, push
- ✓ Scalability: how to manage hundreds of devices → message routing, connection management
- ✓ Reliability: quality of service, availability of the system

A lot of effort in writing boilerplate code to build good connectivity,
and not focusing on what your devices really should do



```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile for currently active connection
    ConnectionProfile profile = NetworkInformation.GetInternetConnectionProfile();
    if (profile != null)
    {
        NetworkConnectivityLevel connectivityLevel = profile.GetNetworkConnectivityLevel();
    }
}
```




```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile for currently active connection
    ConnectionProfile profile = NetworkInformation.GetInternetConnectionProfile();
    if (profile != null)
    {
        NetworkConnectivityLevel connectivityLevel = profile.GetNetworkConnecti
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg)
{
    perror(msg);
    exit(0);
}

int main(int argc, char *argv[])
{
    int sockfd, portno, n;
    struct sockaddr_in serv_addr;
    struct hostent *server;

    char buffer[256];
    if (argc < 3) {
        fprintf(stderr, "usage %s hostname port\n", argv[0]);
        exit(0);
    }
    portno = atoi(argv[2]);
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0)
        error("ERROR opening socket");
```



```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile for currently active connection
    ConnectionProfile profile = NetworkInformation.GetInternetConnectionProfile();
    if (profile != null)
    {
        NetworkConnectivityLevel connectivityLevel = profile.GetNetworkConnectivityLevel();
    }
}
```

```
URLConnection connection = new URL(url + "?" + query).openConnection();
connection.setRequestProperty("Accept-Charset", charset);
InputStream response = connection.getInputStream();
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg)
{
    perror(msg);
    exit(0);
}

int main(int argc, char *argv[])
{
    int sockfd, portno;
    struct sockaddr_in serv_addr;
```

```
portno = atoi(argv[2]);
sockfd = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd < 0)
    error("ERROR opening socket");
else
    printf("socket created\n");
memset(&serv_addr, 0, sizeof(serv_addr));
serv_addr.sin_family = AF_INET;
```



```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile
    ConnectionProfile profile = NetworkConnectivityOptions.DefaultConnectionProfile;
    if (profile != null)
    {
        NetworkConnectivityOptions.DefaultConnectionProfile = profile;
    }
}
```

URLConnectivityOptions
connectivityOptions
InputStream

```
#include <Ethernet.h>
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
byte ip[] = { 10, 0, 0, 177 };

Server server(80);

void setup()
{
    Ethernet.begin(mac, ip);
    server.begin();
}

void loop()
{
    Client client = server.available();
    if (client) {
        // una solicitud http finaliza con una linea en blanco
        boolean current_line_is_blank = true;
        while (client.connected()) {
            if (client.available()) {
                char c = client.read();
            }
        }
    }
}
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg)
{
    perror(msg);
    exit(0);
}

int main(int argc, char *argv[])
{
    int sockfd, portno;
    struct sockaddr_in serv_addr;
    struct hostent *he;

    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0) {
        perror("opening socket");
    }
    portno = 80;
    serv_addr.sin_family = AF_INET;
    he = gethostbyname(argv[0]);
    if (he == NULL) {
        fprintf(stderr, "usage %s hostname port\n", argv[0]);
        return 1;
    }
    serv_addr.sin_addr = *(struct in_addr *)he->h_addr;
    serv_addr.sin_port = htons(portno);
    if (connect(sockfd, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        perror("connect");
        return 1;
    }
    bzero(buffer, 1024);
    int n = read(sockfd, buffer, 1024);
    if (n < 0) {
        perror("read");
        return 1;
    }
    printf("read %d bytes\n", n);
    return 0;
}
```



```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile
    ConnectionProfile profile;
    if (profile != null)
    {
        NetworkConnecti
    }
}
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
```

```
void error()
{
    perror("Connection timed out");
    exit(1);
}
```

URLConnection
connect
InputS

java.net.ConnectException: Connection timed out
Caused by: java.net.ConnectException: Connection timed out
at java.net.PlainSocketImpl.socketConnect(Native Method)
at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:333)
at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:182)
at java.net.Socket.connect(Socket.java:516)
at java.net.NetworkClient.connect(NetworkClient.java:157)
at sun.net.www.http.HttpClient.openServer(HttpClient.java:365)

```
usage %s hostname port\n", argv[0]);
t(AF_INET, SOCK_STREAM, 0);
ROR opening socket");
```



```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}

void NetworkInformation_NetworkStatusChanged(object sender)
{
    // Set behavior based on connectivity status
    SetDataConnectivityOptions();
}

void SetDataConnectivityOptions()
{
    // Get connection profile
    ConnectionProfile profile;
    if (profile != null)
    {
        NetworkConnectivityOptions options = new NetworkConnectivityOptions(profile)
    }
}
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
```



URLConnection
connect
InputS

```
#include <string.h>
byte mac[] = {0, 8, 0, 0, 0, 0};
byte ip[] = {192, 168, 1, 1};

void setup()
{
    Ethernet.begin(mac, ip);
    server.begin(80);
}

void loop()
{
    Client client;
    if (client)
    {
        // una s
        boolean
        while
        if
    }
}

java.net.ConnectException
Caused by: java.net.Pl
at java.net.Pl
at java.net.Pl
at java.net.Pl
at java.net.Pl
at java.net.Pl
at sun.net.www
```

```
ava:333)
etImpl.java:195)
ava:182)
```

```
ava:157)
ent.java:365)
dr;
```

```
hostname port\n", argv[0]);
STREAM, 0);
");
```




```
public App()
{
    // Set initial behavior based on connectivity
    SetDataConnectivityOptions();
    // Set up handler for network status change
    NetworkInformation.NetworkStatusChanged += NetworkInformation_NetworkStatusChanged;
}
void NetworkInformation_NetworkSta
{
    // Set behavior based on connecti
    SetDataConnectivityOptions();
}
void SetDataConnectivityOptions()
{
    // Get connection profile
    ConnectionProfile profile;
    if (profile != null)
    {
        NetworkConnecti
    }
}
```

```
#include <string.h>
#include <sys/types.h>
#include <sys/
```

```
333)
pl.java:195)
182)
```

```
.157)
.java:365)
```

```
URLConnection
connect
InputS
```

```
void loop()
{
    Client client;
    if (client)
    {
        // una s
        boolean
        while
        if
    }
}
```

```
java.net.C
Caused by
```

```
at java.
at java.net.
at java.net.
at sun.net.www.h
at sun.net.www.h
nea en blanco
```

```
name port\n", argv[0]);
M, 0);
```





Jfokus 2014, Stockholm





Lhings is a cloud service that allows you to connect, manage and control your devices easily without worrying about the nitty gritty details of networking

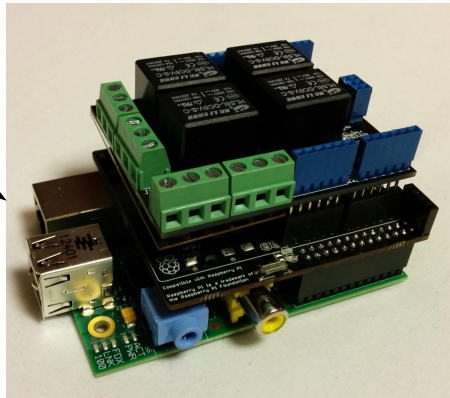


The three main concepts behind Lhings...



Actions

Device



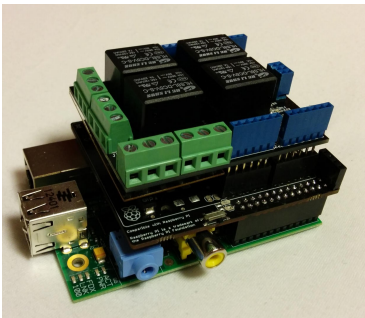
Events



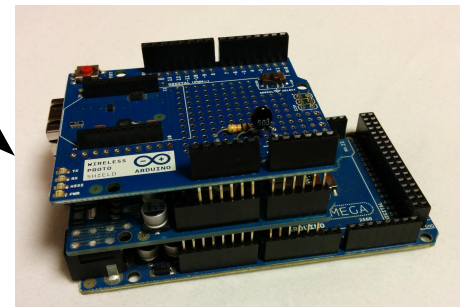
Status



... and the fourth: the rule concept

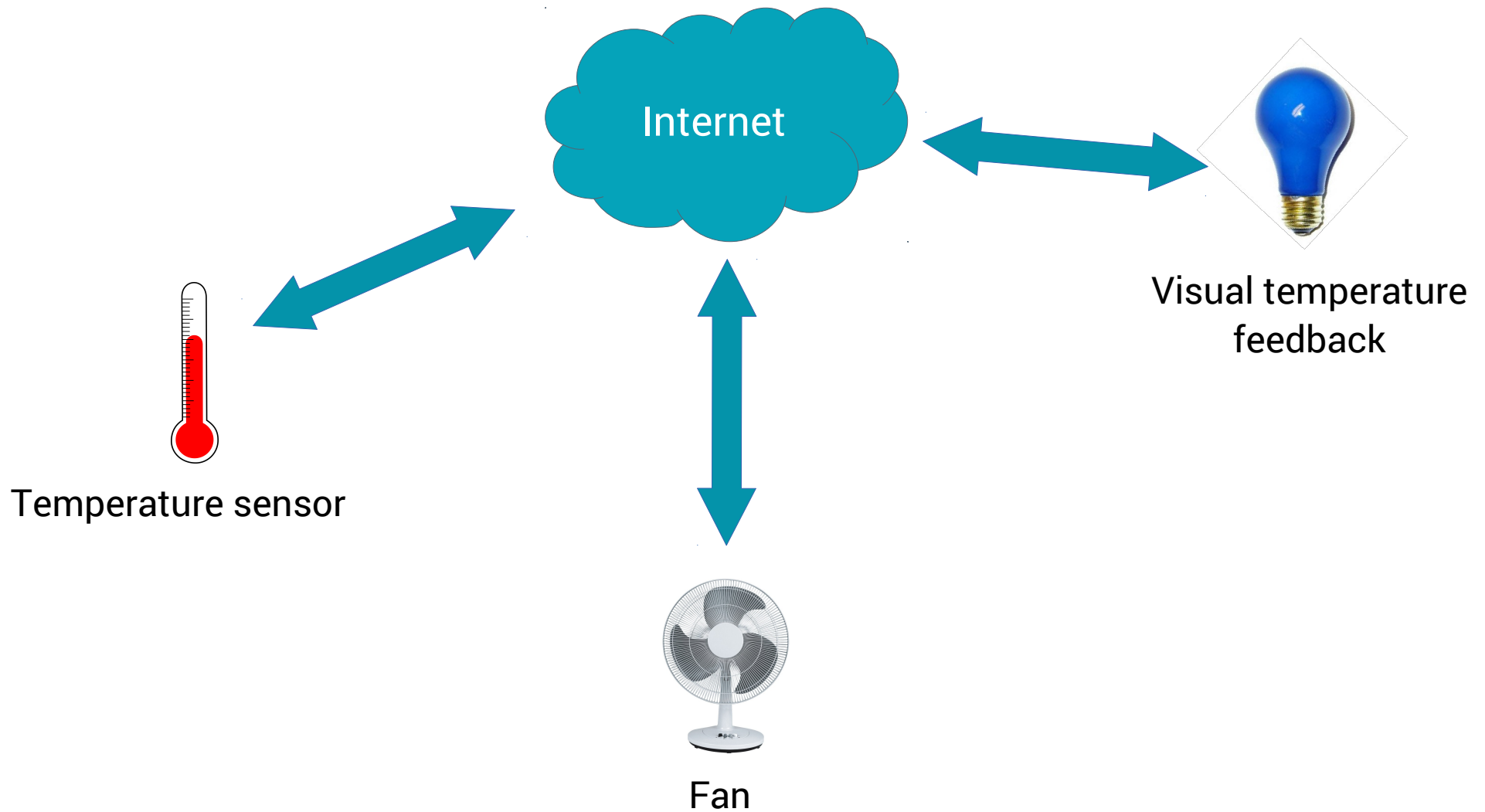


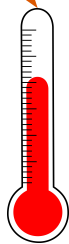
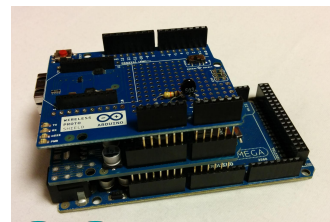
Device X



Device Y



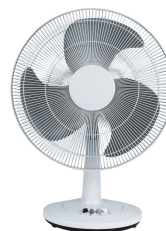
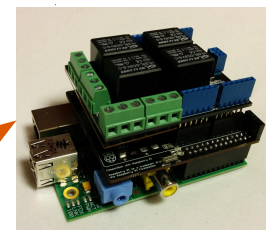




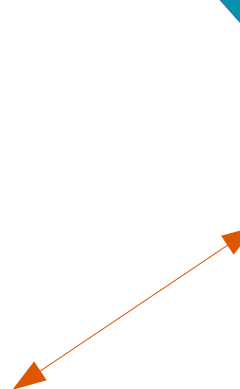
Temperature sensor

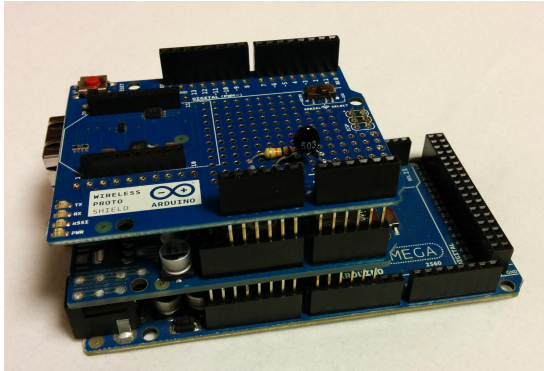


Visual temperature
feedback



Fan



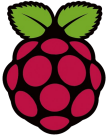
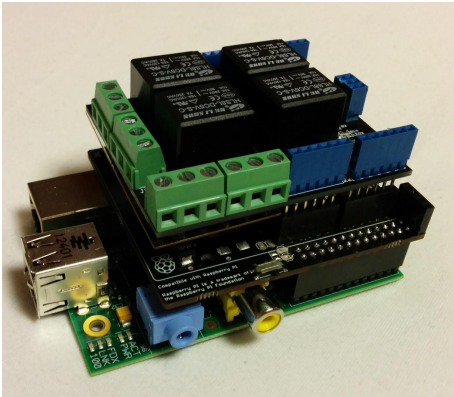


Arduino board with temperature sensor:

- **Events:**
 - Temperature
 - Above threshold
 - Below threshold
- **Status:**
 - Temperature.



Raspberry Pi with relay shield:



- **Actions:**
 - Hue on
 - Hue off
 - Set color
 - Fan on
 - Fan off
- **Status:**
 - Fan status



And now let's go with the live demo!



What we get?

- We forgot completely about networking code.
- All messages transmitted are secure.
- We don't need to worry about firewalls, no need to open ports.
- Ability to push is provided out of the box.
- You get a web control panel for your devices automatically.
- Easy to scale.
- Reliability.



Do you want to try it? Our gift to you for coming to this talk is a free invitation to test our beta version.



The Lhings libraries and the code for this demo is available on Github:

<https://github.com/lhings>

