FICHE DE SYNTHESE
Internet des Objet « IdO »

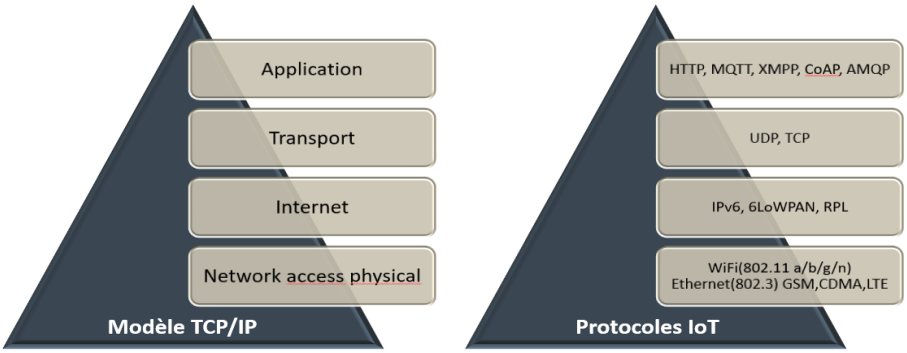
Internet des Objet « IdO »

En anglais : Internet of Things « IoT »

A. Activité 1 : envoi de données d'un capteur sur un serveur web et affichage dans une IHM

Que faut-il retenir ?

Fondamental : Les couches...



Pour communiquer l'IdO fonctionne en couches, dans notre exemple d'utilisation en WiFi :

- la couche WiFi est la couche support de l'information
- la couche IP permet d'obtenir une adresse IP
- TCP est la couche de transport
- dans la couche application se trouve HTTP le protocole de communication

Complément : Remarque concernant le programme WiFiScan

Connexion au WiFi :

```
1   // We start by connecting to a WiFi network
2   WiFi.mode(WIFI_STA);
3   WiFiMulti.addAP("SSID", "passpasspass");
```

Dans le **setup :**

- On configure le module WiFi en **mode Station** (il existe aussi le mode point d'accès AP). Il s'agit d'une commande spécifique au module ESP.
- On ajoute un point d'accès AP avec la **méthode addAP.**
- **SSID** est l'**identifiant de la box**
- passpasspass est la **clé WiFi** associée à la box.

Envoi des données par la méthode HTTP POST :

```
1
//*************************************************************************************
2
//-------------------- 3. envoi des donnees sur le serveur web----------------------------
3
//*************************************************************************************
4
5
6     // on envoie les données
7     if (client.connected())
8     {
9           Serial.print(F("Envoi de la requete..."));
10           // On l'envoie au serveur sur plusieurs lignes
11            // POST  HTTP/1.1
12            // Host: dweet.io
13            // Connection: close
14            //
15            // La première ligne indique la version du protocole HTTP
16            // La deuxième le nom du serveur
17            // important car on peut trouver différents serveurs à une même adresse IP
18            // La troisième ligne indique que le serveur doit fermer la
19            // connexion apres la réponse et ne pas attendre d'autres requêtes.
20
21            // affichage dans le moniteur serie afin de controler ce qui sera envoyé au serveur web :
22            Serial.println(String("POST ")+ url + monObjet +String("?temperature=")+String(t)+String("&humidite=")+String(h)+ " HTTP/1.1\r\n" +
23                  "Host: "+host + "\r\n" +
24                  "Connection: close\r\n\r\n");
25
26            // envoi de la requete
27            client.print(String("POST ")+ url + monObjet +String("?temperature=")+String(t)+String("&humidite=")+String(h)+ " HTTP/1.1\r\n" +
28                  "Host: "+host + "\r\n" +
29                  "Connection: close\r\n\r\n");
30
31            // On attend 1 seconde
32            delay(1000);
33    }
```

B. Activité 2 : parser des données extraites d'une API sur un serveur Web

Définition : Parser ?

Parser = ...

Attention : Droits d'utilisation des données

Lorsqu'on utilise les données d'une API, ...

Par exemple, ici pour les données du site infoclimat :

- la diffusion des données est **autorisée à la condition d'ajouter quelques lignes pour citer la source :**

Méthode personnalisée : données de prévisions météo à 7 jours en JSON, XML, CSV

Vous pouvez faire appel à l'API Infoclimat, qui vous retournera les prévisions détaillées pour cette ville.
Nous vous demandons juste de spécifier la source, sous la forme d'un lien vers www.infoclimat.fr, dans vos applications ou pages utilisant ces données.

Complément : Analyse du json de infoclimat

Que renvoie infoclimat ? Quelles sont les clés et les valeurs renvoyées dans le json ?

```
// la chaine envoyee par infoclimat est du type :
// {"request_state":200,
// "request_key":"fd543c77e33d6c8a5e218e948a19e487",
// "message":"OK",
// "model_run":"20",
// "source":"internal:GFS:1",
// "2019-01-22 23:00:00":
// {"temperature":{"2m":282.7,"sol":285,"500hPa":-0.1,"850hPa":-0.1},
// "pression":{"niveau_de_la_mer":103080},
// "pluie":0,
// "pluie_convective":0,
// "humidite":{"2m":91.6},
// "vent_moyen":{"10m":12.7},
// "vent_rafales":{"10m":40.4},
// "vent_direction":{"10m":401},
// "iso_zero":3519,
// "risque_neige":"non",
// "cape":0,
// "nebulosite":{"haute":0,"moyenne":0,"basse":5,"totale":5}},

```

Méthode : Parser les données...

On souhaite extraire la date, l'heure, et la température du json :

1. lire le json ligne par ligne en se basant sur le **séparateur virgule ','** :

Remarque : ici on choisit de lire 15 lignes soit 15 relevés sur une date et/ou heure différentes.

```
// lecture des donnees JSON
        String ligne = http.getString();
```

2. Utilisation de la méthode..... appliquée à ligne et d'un tableau pos pour stocker la position du **mot clé** recherché :

- **"sol"** pour la température au niveau du sol
- **"2019-"** pour l'année ce qui permettra de trouver la date et l'heure.

```
// pos est l'index de position du mot clé "sol"
  // indexOf donne l'index de position du premier caractere du mot clé "sol:",
     pos[i] = ligne.indexOf(keyword);
   // pos1 est l'index de position du mot clé "2019-"
   pos1[i]=ligne.indexOf(keyword2);
```

La méthode indexOf renvoie -1 si elle ne trouve pas le mot clé sinon elle renvoie elle renvoie la valeur de l'index de position du premier caractère du mot clé :
donc **si pos[i] est positif on a trouvé le mot clé !**

Pour la date le json renvoie : "2019-mm-jj hh:mm:ss"

- le mois est en position 6
- le jour est en position 9
- et l'heure en position 12

3. Les **variables déclarées** sont :

```
int pos[15] ;
int pos1[15] ;
float temperature[15];

// position des valeurs recherchées mois date heure dans la donnée : "2019-mm-jj hh:mm:ss"
int posmois=6;
int posdate=9;
int posheure=12;
// tableau de stokage des numéros de mois
String mois[15];
//  tableau de stokage des dates
String jour[15];
String nommois[]={"Janvier", "Fevrier", "Mars", "Avril", "Mai", "Juin", "Juillet", "Aout", "Septembre", "Octobre", "Novembre", "Decembre"};
//  tableau de stokage des heures
String heure[15];

```

4. Parsing des données pour trouver l'heure, la date et le mois :

```
 if(pos1[i]>=0) { /* 2019- trouve dans la ligne reçue*/

          // longueur en nombre de caracteres du JSON
               longueur=ligne.length();

          // extraction de la date jusqu'à la fin de ligne
               lignedate=ligne.substring(pos1[i], pos1[i]+20);
               Serial.print("lignedate = ");
               Serial.println(lignedate);

          // compléter les tableaux mois, date et heure
               heure[i]=&lignedate[posheure];

               jour[i]=lignedate.substring(posjour,posheure);
               mois[i]= lignedate.substring(posmois,posjour-1);

     }
```

Remarque :
La méthode substring(debut, fin) renvoie la chaîne comprise entre l'index début et l'index fin
Pour le mois il faut retirer 1 à posjour pour enlever le symbole "-"

5. Même principe pour la **température au niveau du sol** :

```
if (pos[i] >= 0) { /* mot "sol" trouve dans les données reçues */
       // indexOf donne la position du début du mot clé,
       // en ajoutant sa longueur on se place à la fin.
       pos[i] += keyword.length();

       Serial.print("pos ");  Serial.print(i);Serial.print(" = ");
       Serial.println(pos[i]);

 if (pos[i] >= 0) { /* mot "sol" trouve dans les données reçues */
      // indexOf donne la position du début du mot clé,
      // en ajoutant sa longueur on se place à la fin.
      pos[i] += keyword.length();

      Serial.print("pos ");  Serial.print(i);Serial.print(" = ");
      Serial.println(pos[i]);

     // extraction de valeur à la fin de ligne
            ligne=ligne.substring(pos[i], longueur);
            Serial.print("extrait = ");
            Serial.println(ligne);

         // extraction de la valeur située jusqu'à la première virgule
            String valeur_sol=ligne.substring(0,ligne.indexOf(","));
         // affichage de la donnée brute extraite
            Serial.print("temperature_sol = ");
            Serial.println(valeur_sol);


         // conversion de la valeur du format texte au format flottant
            temperature[i] = atof(&valeur_sol[0])-273;
            Serial.print("temperature_sol = ");
            Serial.println(temperature[i]);
```

MINI-PROJET

M et Mme Givais ont installé un **panneau solaire sur le toit terrasse** de leur habitation.

- L'inclinaison de celui-ci est automatisée et basée sur les valeurs permettant d'avoir l'angle optimal d'incidence et ainsi le meilleur rendement.
- Cependant ce couple voudrait **sécuriser ce panneau en automatisant la mise en sécurité** en fonction des valeurs des rafales de vent.
- Le laboratoire de STI2D leur propose de réaliser une **mise en sécurité automatique** en se basant sur les **données météo de la station la plus proche** de chez eux.
- Ils pourront aussi disposer d'une **girouette miniature** qui indique automatiquement la direction du vent et d'une **IHM consultable sur internet** depuis leur smartphone ou leur PC avec les valeurs suivantes :
  - **la direction du vent,**
  - **sa valeur moyenne en km/h**
  - **sa valeur en rafales en km/h**
- Les données seront prises sur l'API du serveur de données météo du site www.infoclimat.fr
- L'IHM sera disponible sur freeboard.io
- La mise à jour des valeurs se fera toutes les heures (rythme de modification des données sur le site internet)

Notre étude portera sur le **pilotage de la girouette et sur la création de l'IHM**
Votre travail se décomposera en 4 étapes :

- Analyse du besoin : réalisation des documents SysML (compléter le fichier Magic Draw fourni) :
  - Expression du besoin
  - Mission du système
  - Utilisation du système -phase d'exploitation
- récupération des données sur le serveur du site infoclimat.fr
- pilotage d'un servomoteur sur lequel sera fixée la girouette
- création de l'IHM pour la visualisation des données à distance

- Chaque étape devra être contrôlée et validée.

Rédiger un compte rendu pour le Mini-projet.