# Laboratoire 3 – Le client Web

## Objectifs :

- Comment connecter un microcontrôleur ESP32 à un réseau Ethernet?
- Comment fonctionne le protocole applicatif HTTP, à la base de l'Internet?
- Comment est constitué une requête du protocole HTTP?
- Comment programmer un microcontrôleur ESP32 pour obtenir et traiter de l'information de l'Internet?
- Qu'est-ce que le format JSON?
- Comment programmer un microcontrôleur ESP32 pour fournir de l'information vers un service infonuagique?

## Matériel requis :

- Carte protoTPhys
- Blindage Ethernet et son adaptateur pour protoTPhys
- Câble ethernet
- Interface de programmation Arduino
- Capteur DHT11

## Théorie associée :

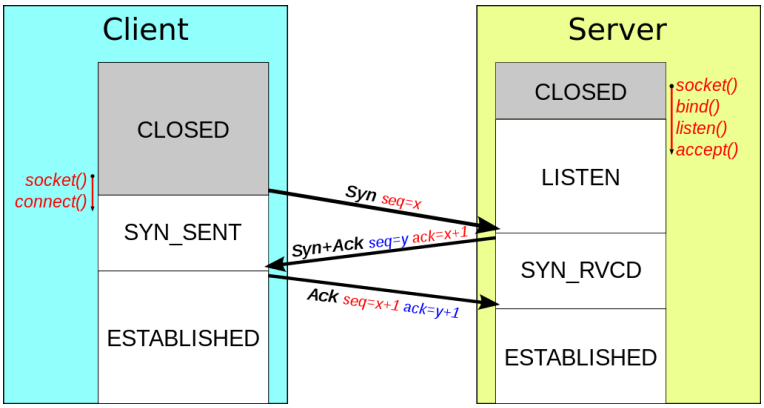
### Théorie : Le modèle client-serveur Web

« Le protocole ou environnement **client–serveur** désigne un mode de transaction (souvent à travers un réseau) entre plusieurs programmes ou processus : l'un, qualifié de **client**, envoie des requêtes ; l'autre, qualifié de **serveur**, attend les requêtes des clients et y répond. Le serveur offre ici un **service** au client. Par extension, le client désigne souvent l'ordinateur sur lequel est exécuté le logiciel client, et le serveur, l'ordinateur sur lequel est exécuté le logiciel serveur. »


Exemple :

« La consultation de pages sur un site Web fonctionne sur une architecture client–serveur. Un internaute connecté au réseau via son ordinateur et un navigateur Web est le client, le serveur est constitué par le ou les ordinateurs contenant les applications qui fournissent les pages demandées. C'est le protocole de communication HTTP […] qui est utilisé. » Source : https://fr.wikipedia.org/wiki/Client-serveur

Voici la façon dont le client se connecte à un serveur pour établir un canal de communication peut être illustré de la façon suivante :




L'image représente la machine d'état de la connexion réseau TCP/IP. L'axe vertical allant du haut vers le bas représente le temps. Le Client est votre téléphone, une tablette, Chrome… ou votre ESP32!

Source image :
https://fr.wikipedia.org/wiki/Transmission_Control_Protocol#%C3%89tablissement_d'une_connexion

Une fois le canal de communication établi, le protocole de transfert http peut se mettre en action.

« L'***HyperText Transfer Protocol*** (**HTTP**, littéralement « protocole de transfert hypertexte ») est un protocole de communication client-serveur développé pour le *World Wide Web*. »

« HTTP est un protocole de la couche application. Il peut fonctionner sur n'importe quelle connexion fiable, dans les faits on utilise le protocole TCP comme couche de transport. Un serveur HTTP utilise alors par défaut le port 80 »… de nos jours, question de sécurité, les serveurs opèrent sur le port# 443.

« Les clients HTTP les plus connus sont les navigateurs Web permettant à un utilisateur d'accéder à un serveur contenant les données [et des documents, images, vidéos, etc]. »

« Dans le protocole HTTP, une méthode est une **commande** spécifiant un type de requête, c'est-à-dire qu'elle demande au serveur d'effectuer une action. En général l'action concerne une ressource identifiée par l'URL qui suit le nom de la méthode. »

« Il existe de nombreuses méthodes, les plus courantes étant **GET**, HEAD et **POST** »

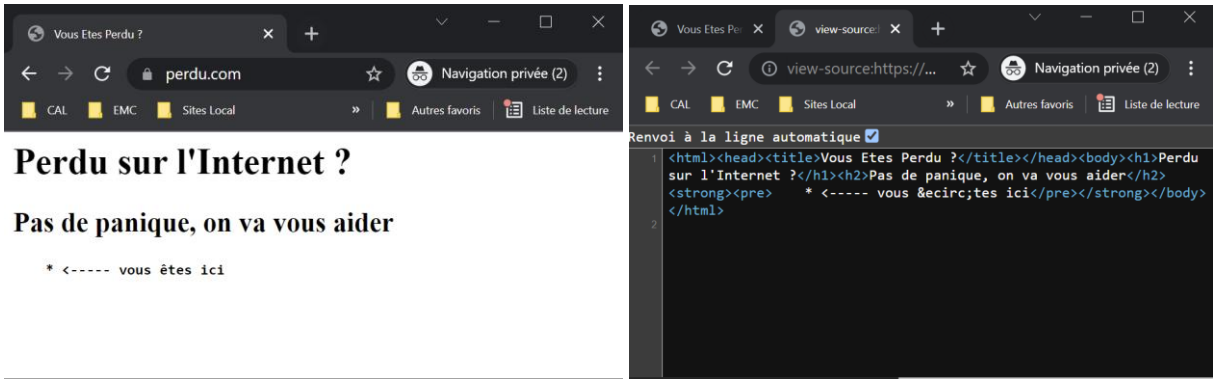
Source : https://fr.wikipedia.org/wiki/Hypertext_Transfer_Protocol

Exemple : http://www.perdu.com, communication textuelle à l'aide de l'invite de commande de Windows et de l'utilitaire « **curl** » (*command-line URL*) :

```
Invite de commandes
C:\Users\yheyn>
C:\Users\yheyn>curl -vvv -G http://www.perdu.com
*   Trying 208.97.177.124:80...
* Connected to www.perdu.com (208.97.177.124) port 80 (#0)
> GET / HTTP/1.1
> Host: www.perdu.com
> User-Agent: curl/7.79.1
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Thu, 24 Feb 2022 01:45:24 GMT
< Server: Apache
< Upgrade: h2
< Connection: Upgrade
< Last-Modified: Thu, 02 Jun 2016 06:01:08 GMT
< ETag: "cc-5344555136fe9"
< Accept-Ranges: bytes
< Content-Length: 204
< Cache-Control: max-age=600
< Expires: Thu, 24 Feb 2022 01:55:24 GMT
< Vary: Accept-Encoding,User-Agent
< Content-Type: text/html
<
<html><head><title>Vous Etes Perdu ?</title></head><body><h1>Perdu sur l'Internet ?</h1><h2>Pas de panique, on va vous a
ider</h2><strong><pre>    * <----- vous &ecirc;tes ici</pre></strong></body></html>
* Connection #0 to host www.perdu.com left intact

C:\Users\yheyn>
```

Si nous tentons la même opération mais à l'aide d'un navigateur (p.ex. Chrome), nous obtenons le rendu de la page (à gauche). Le code source de la page (code HTML) est présenté sur l'image de droite à l'aide de l'option « Afficher le code source de la page » du menu contextuel (bouton droit de la souris) :




## Théorie : Une API (*Application Programming Interface*)

« En informatique, une **interface de programmation d'applications** ou **interface de programmation applicative** […] est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels. »

« La description de l'interface de programmation spécifie comment des [logiciels] clients peuvent interagir avec un logiciel [serveur] en mettant l'accent sur les fonctionnalités offertes par le logiciel [serveur] et en cachant les détails de son fonctionnement »

Source : https://fr.wikipedia.org/wiki/Interface_de_programmation

## Manipulation 1 – Le simple client Web, méthode GET – Les données de la ISS

**Objectif** : Programmer une application pour le ESP32 permettant d'obtenir et afficher la position, à intervalle régulière, de la station spatiale internationale (ISS).

La page open-notify (http://open-notify.org/) nous renseigne comment utiliser son service. Nous n'avons alors qu'à formuler la requête, par exemple: http://api.open-notify.org/iss-now.json (via un navigateur web ou tout autre outil, tout comme le ESP32 pourra le faire) :


⇨ Testez votre requête http GET à l'aide de l'utilitaire « curl » (invite de commande Windows 11).

**Conserver le résultat cette commande pour la remise**.

Cette partie du laboratoire vous permettra d'en apprendre sur la communication client Web utilisant le protocole http et la méthode GET ainsi que le traitement du message retourné par le serveur.

Un code de base vous est fourni en guise de référence et réalisant (presque) le travail d'obtenir l'information d'un serveur distant sur internet.

Code de base Arduino : **CommObj_ESP32_ClientWEB_SimpleGET_Ethernet_Vetudiant**

Vous devrez :

- Finaliser le code de base en indiquant le serveur et la ressource (path) à obtenir
- Mettre en place les librairies et parvenir à compiler le code de base
- Transformer le message reçu en un objet JSON (via la librairie ArduinoJson)
- Modifier le code de base pour décoder le contenu et l'afficher à l'écran OLED l'information pertinente
- Transcrire le code de base dans le format du modèle de référence (AVEC commentaires).

Librairies requises :

- ArduinoJson v7.3.x : https://arduinojson.org/
- TGP_Ecran : https://github.com/TechnoPhysCAL
- Arduino Ethernet v2.0.2 https://www.arduino.cc/reference/en/libraries/ethernet/
- Arduino HTTPClient v0.6.1 https://github.com/arduino-libraries/ArduinoHttpClient

À vous donc d'aller chercher l'information, sur une base régulière (à chaque 2 minutes, environ), de la position de la station spatiale et de l'afficher sur l'écran OLED.

**Présentez à l'enseignant le résultat de votre application.**

Comme résumé de cette partie, vous devrez fournir le code logiciel transformé, dans le format spécifié, dument commenté.

## Manipulation 2 – Obtenir des données météo

Il existe une myriade d'API ouverts sur le Web. Une page Github en répertorie une sélection: https://github.com/public-apis/public-apis/blob/master/README.md

Parmi cette liste, une source d'information d'intérêt concerne les données météorologiques, OpenWeatherMap (OWM) : https://openweathermap.org/

Afin de réaliser cette partie, vous allez devoir vous créer un compte personnel et attendre d'obtenir votre clé d'API. Conservez cette **clé privée pour vous seulement**. Ne la partagez pas, cela compromettrait votre compte.

Consulter la page de RNT sur comment faire communiquer le ESP32 avec le service OWM : https://randomnerdtutorials.com/esp32-http-get-open-weather-map-thingspeak-arduino/ Mais **ATTENTION** : vous devez passer par la carte Ethernet et non pas via le Wifi.

Ensuite, visitez le site web d'OWM et consultez la page « Current Weather Data » sous l'onglet **API**.

Lisez bien les instructions et l'exemple de l'API à propos de ce que vous devez fournir comme information lors de la requête http GET.

⇨ Testez votre requête http GET à l'aide de l'utilitaire « curl » (invite de commande Windows 11).

**Conserver le résultat cette commande pour la remise**.

Une fois la requête complétée avec succès à l'aide de curl, transférer ce que vous avez compris au niveau du ESP32. Ce sera ce dernier qui forgera, fera la requête et traitera le résultat. Tentez l'expérience d'obtenir les données météo d'une localité et de les présenter via le moniteur série de l'IDE Arduino par votre plaque protoTPhys. Vous aurez ainsi une application dédiée à la météo.

Lorsque vous serez parvenu à obtenir les données d'une réponse du serveur sur le moniteur série, copier et coller le résultat dans un outil JSON prettifyer tel que: https://jsonformatter.org/json-pretty-print

Allez ensuite un peu plus loin, en décodant ce contenu à l'aide des outils de la librairie ArduinoJson. Les données à considérer, conditions actuelles : température, humidité, pression barométrique, vitesse et direction des vents. Faites afficher ces informations au moniteur série et de façon plus pratique, à l'écran OLED de votre plaquette.

Code de base : **CommObj_ESP32_ClientWEB_GET_OWM_Yh_V3_Ethernet_Vetudiant**

**Présentez à l'enseignant le résultat de votre application.**

## Manipulation 3 – Envoyer des données sur une plateforme infonuagique

ThingSpeak est une plate-forme de stockage et traitement des données pour l'Internet des objets. Elle vous permet de collecter des données dans ce qui est appelé un « canal » et d'obtenir des données d'autres canaux à l'aide de l'API (au besoin).

Le but de cette partie du laboratoire est d'envoyer les données du capteur DHT11 (température et humidité) au serveur ThingSpeak. Vous configurez ensuite votre page ThingSpeak en présentant 2 graphiques, 1 pour chaque mesure.

Pour ce faire, vous aurez besoin de vous enregistrer sur le site ThingSpeak afin d'y configurer un récepteur (Channel) et d'obtenir **VOTRE clé d'API (PRIVÉ!)**. Suivre les instructions sur la page « Get Started for Free » du site https://thingspeak.com

Consulter l'inspiration du site suivant en ce qui a trait à cette section du laboratoire: https://randomnerdtutorials.com/esp32-thingspeak-publish-arduino/

Pour l'envoi de données multiples, voyez comment faire en consultant la section « Sending Multiple Fields » du lien fourni ci-haut.

ATTENTION : vous devez adapter les instructions de RNT à ce laboratoire. En effet, dans votre cas, les données capturées sont température, humidité à l'aide du DHT11. VOUS N'AVEZ PAS DE BME280. De plus, vous devez utiliser la carte Ethernet, et **NON PAS LE WIFI!!!**

**Présentez à l'enseignant le résultat de votre application.**

## Remise

Consignes pour le résumé de laboratoire.

- Le résumé de laboratoire représente avant tout un recueil de vos notes, observations et résultats de vos travaux. Il a pour but de forger en vous une bonne pratique professionnelle, en tant que futur technologue, de consigner et de décrire vos travaux sur des mandats qui vous seront confiés.
- 1 remise de travail par équipe.
- Je veux avoir, sur peu de pages, sections bien identifiées :

  Une **page couverture** vous identifiant, titre du travail, présenté à, la date de remise, lieu de remise (CÉGEP A-L).

  **Introduction** :

  - Date des manipulations accompagné d'une brève description des travaux réalisés pour chaque période.
  - Rappel des objectifs du laboratoire (pourquoi faire ce labo?).

  **Développement** :

  - Veuillez développer/expliquer (sommairement) le déroulement général de votre code. Chaque manipulation, séparément, bien identifiées, justifiez ce que font les parties pertinentes ajoutées ou modifiées par rapport au code de base fourni. À titre d'exemple, prenez comment le site RandomNerdsTutorial présente sa section « How the Code Works ».
  - Les résultats de la commande curl (SANS la clef d'API).
  - Les 2 graphiques des données ThingSpeak.

  **Conclusion** :

  - Discussion sur les résultats obtenus, notes et observations pertinentes.
  - Ce que vous avez appris ou découvert (ou peut-être pas).
  - Une critique constructive de cet exercice de laboratoire. Votre appréciation de ce laboratoire. Soyez honnête.

- Fournir tous les codes logiciels complets des parties 1, 2 et 3, bien identifiées. N'oubliez pas d'appliquer correctement et entièrement le modèle de code (entête, section, commentaires).

- Consigne pour la remise
  - Rapport format Word
  - Remettre le programme Arduino de chacune des manipulations
  - **SVP: Aucun fichier archive (zip)**