

1


UNIVERSITÉ
La Rochelle

IoT (Internet of Things)... pourquoi maintenant ?

- Puissance de traitement à faible coût
- Multitude de capteurs, prix en baisse
- Couverture 3G/4G/5G si pas de réseau local (Bande passante adéquate pour transmission de données)
- Smartphones (= Interface homme-machine)
- Engouement pour le big data (collecte de données)
- IPv6 ($3,4 \times 10^{38}$ adresses disponibles)

1	Smart Home	100%
2	Wearables	63%
3	Smart City	34%
4	Smart grid	28%
5	Industrial internet	25%
6	Connected car	19%
7	Connected Health	6%
8	Smart retail	2%
9	Smart supply chain	2%


2



La Rochelle

IOT Classification 1/2

- Most applications fall into one of 3 categories*
 - **Monitoring Space**
 - Environmental and Habitat Monitoring
 - Building automation
 - Precision Agriculture
 - Data Centers
 - Indoor Climate Control
 - Military Surveillance
 - Treaty Verification
 - Intelligent Alarms
 - **Monitoring Objects**
 - **Monitoring Interactions of Objects and Space**



* *Classification proposed by Culler, Estrin, Srivastava*

3



La Rochelle

IOT Classification 2/2

- **Monitoring Objects**
 - Structural Monitoring
 - Eco-physiology
 - Condition-based Maintenance
 - Medical Diagnostics
 - Urban terrain mapping
- **Monitoring Interactions of Objects and Space**
 - Wildlife Habitats
 - Disaster Management
 - Emergency Response
 - Ubiquitous Computing
 - Asset Tracking
 - Health Care
 - Manufacturing Process Flows

Structural monitoring (motion)



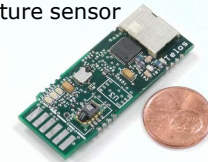


4

Les premiers modules pour IoT (vers 2005)

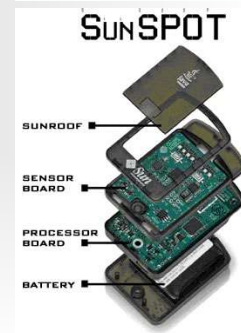
■ Telos : UC Berkeley & Intel Research

- Single board :
 - Robustness, Ease of use, Lower Cost
 - Integrated Humidity & Temperature sensor
- First platform to use 802.15.4 (Zigbee)
- Motorola HCS08 40 MHz processor, low power, 1.8V



■ Sun SPOT

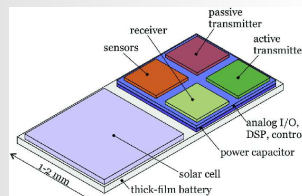
- no OS but programmable in Java
- 180 MHz, 32-bit ARM core processor, 802.15.4 radio
- 3.6V rechargeable 750mAh lithium-ion battery, consumes only 36 μ A in deep sleep mode
- Sensor Board : Accelerometer, light, temperature sensors + color LEDs, digital I/O pins, ...



5

Le concept « poussière intelligente » : smart dust

- Réduire la taille à quelques mm²
- *Everything on one chip*
- Portée radio limitée : 15m à 20m ou transmission optique
- Berkeley Motes
- MICA chip
 - 2 x AA batteries
- i-motes INTEL

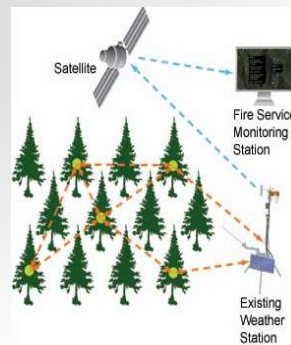
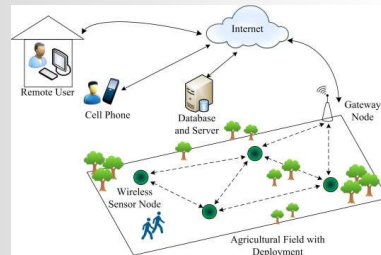


6

6

Quelques concepts en relation avec l'IoT

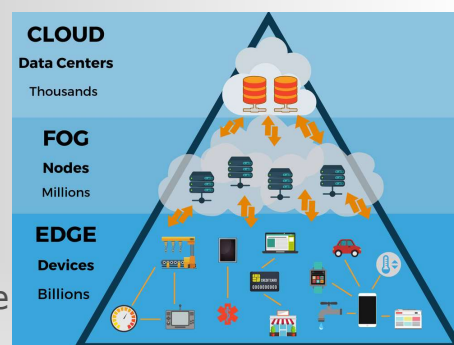
- Les réseaux de capteurs (WSN Wireless Sensor Network)
- M2M (Machine To Machine) communications temps réel de données sans intervention humaine
- Edge Computing
Traiter les données à la périphérie du réseau, près de la source des données



7

Edge Computing

- Equilibrer le traitement des données dans le « réseau »
 - Effectuer des calculs au plus près des capteurs pour réduire la quantité des données à transmettre
 - Mise à l'échelle, moyennes journalières, ...
 - Répartition dynamique et collaborative des calculs
 - Développer des algorithmes parallèles et repartis au lieu de tout traiter par le serveur de collecte



8

Retour à l'ESP32 : Wireless



- Meilleur choix économique et technique face à la solution filaire.
- Attention : lacunes concernant la confidentialité, la sécurité et de l'intégrité des données.

9

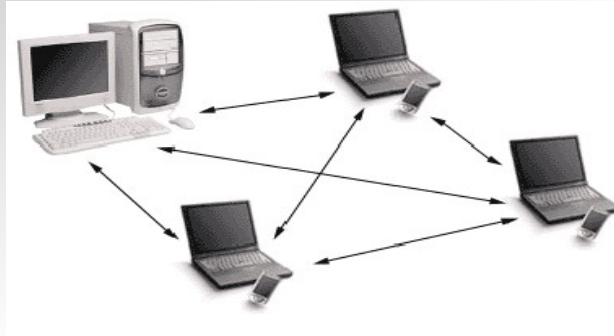
WiFi (WLAN) : Couche Physique

- Norme IEEE 802 . 11 (ISO/IEC 8802-11)
 - Débits initiaux de 1 à 2Mbits.
 - Amélioration successive de la norme 802.11a, 802.11b, 802.11g, 802.11n, 802.11ac, ...
 - Débit 11 Mbits ou plus (de l'ordre de 600Mbit/s en 2,4 GHz et plusieurs Gbit/s sur 5GHz ou agrégation sur un rayon de plusieurs dizaines de mètres en intérieur.
- La couche MAC autorise l'établissement de deux types de réseaux:
 - Mode Ad hoc.
 - Mode Infrastructure.
 - Dans les deux cas : SSID (Service Set Identifier)
 - trames balises (*Beacon Frame*), toutes les 100ms environ
 - Cryptage possible (WEP, WPA, WP2)

10

Mode Ad Hoc

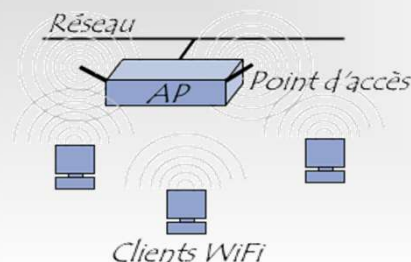
- Les clients communiquent entre eux sans passer par un équipement central.
- Aucun des participants (Émetteur / Récepteur) n'a de rôle particulier.
- peut devenir compliqué si le nombre de machines augmente
- Problème de sécurité



11

Le mode infrastructure

- Au moins un Émetteur / Récepteur Wifi joue un rôle particulier, celui de point d'accès (AP).
- La communication entre clients passe par l'AP. Le périphérique est le client, l'AP est le maître.
- Mode utilisé pour étendre un réseau câblé, genre Ethernet, avec une couverture Wifi pour les portables (ou les machines que l'on souhaite pas câbler).



12

Bluetooth

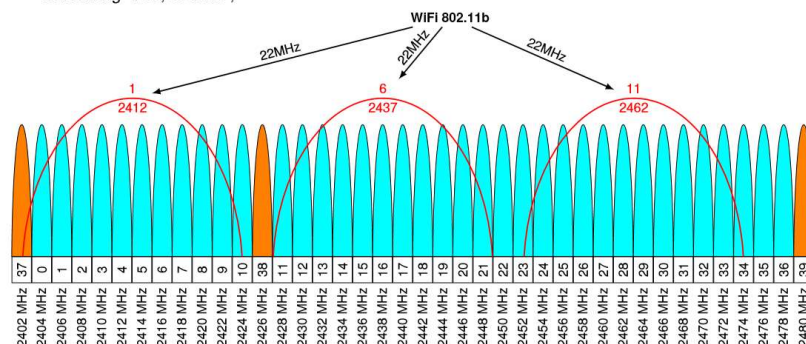
- *Bluetooth* est directement inspiré du surnom du roi viking Harald Blåtand (en anglais Harald Bluetooth), connu pour avoir réussi à unifier les tribus danoises au sein d'un même royaume
- Norme de télécommunications sans fil permettant l'échange bidirectionnel de données à courte distance. Son but est de simplifier les connexions entre les appareils électroniques à proximité.
- **Bluetooth Low Energy BLE**
- Version basse consommation. BLE reste le plus souvent en mode de veille.
- La consommation énergétique du BLE est environ 100x moindre que le Bluetooth classique.

13

Bluetooth : Couche Physique

BLE utilise 40 canaux de 2400MHz à 2480MHz regroupé en deux types :

- données :
 - ◊ de 0 à 36 ;
 - ◊ communication bidirectionnelle entre éléments connectés ;
 - ◊ saut de fréquence adaptatif : $f_{n+1} = (f_n + \text{hop}) \bmod 37$ avec *hop* allant de 5 à 16 ;
- «advertising» : 37, 38 et 39 ;




Wifi 802.11b en mode DSSS, des blocs de 22MHz, soient 3 blocs indépendants (sans chevauchement) :

- ▷ Channel 1 : centré sur 2412, de 2401 (2412-11) à 2423 (2412+11) ;
- ▷ Channel 6 : centré sur 2437, de 2426 (2437-11) à 2448 (2437+11) ;
- ▷ Channel 11 : centré sur 2462, de 2451 (2462-11) à 2473 (2462+11) ;

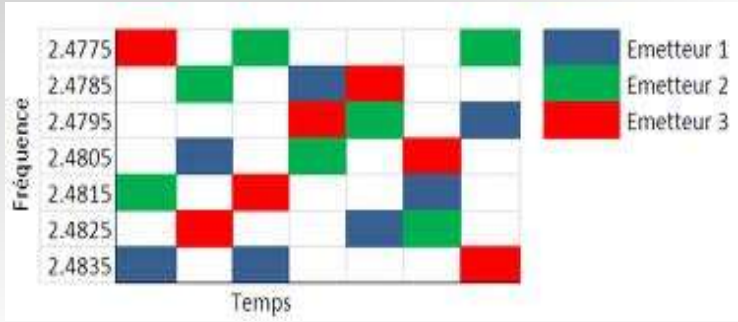
BLE, 3 canaux d'«advertisement» : 37, 38 et 39 ;

14




Bluetooth : Couche Physique

- Bluetooth emploie la gamme radio de 2,45 GHz, Cette bande est divisée en 79 canaux d'1 Mhz chacun.
- Bluetooth utilise le principe FHSS (*frequency-hopping spread spectrum*), qui consiste a permettre d'émettre à plusieurs simultanément, grâce à la commutation rapide entre plusieurs canaux de fréquence.



15



Protocoles

- De la transmission série aux protocoles très complexes, presque tout est possible

Différents protocoles

Protocol Name	Transport Protocol	Messaging Model	Security	Best-Use Cases	Architecture
AMQP	TCP	Publish/Subscribe	High-Optional	Enterprise integration	P2P
CoAP	UDP	Request/Response	Medium-Optional	Utility field	Tree
DDS	UDP	Publish/Subscribe Request/Response	High-Optional	Military	Bus
MQTT	TCP	Publish/Subscribe Request/Response	Medium-Optional	IoT messaging	Tree
UPnP	-	Publish/Subscribe Request/Response	None	Consumer	P2P
XMPP	TCP	Publish/Subscribe Request/Response	High-Compulsory	Remote management	Client/Server
ZeroMQ	UDP	Publish/Subscribe Request/Response	High-Optional	CERN	P2p

- Evidemment, on utilise aussi le protocole HTTP

16

Protocoles spécifiés Bluetooth:

- L2CAP (Logical link control and adaptation protocol) :
 - Ce protocole adapte les protocoles des couches supérieures (SDP, RFCOMM) sur la couche bande de base. Il s'appuie sur la couche inférieure afin d'assurer le contrôle du flux et des erreurs, il emploie des liaisons asynchrones seulement.
- SDP (Service Discovery Protocol):
 - Ce protocole est chargé de recueillir des informations sur les types de dispositif, les services et les caractéristiques de service de sorte qu'un raccordement entre les dispositifs puisse être installé.

17

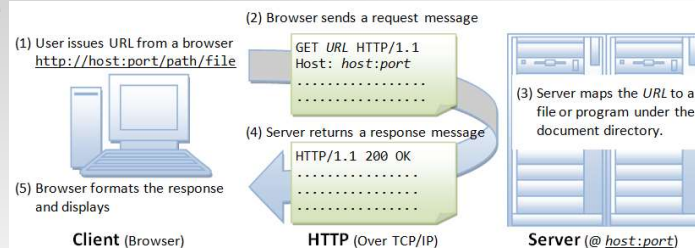
Bluetooth : Protocoles adoptés

- PPP :
 - Permet le transport de datagrammes IP sur une liaison point à point.
- TCP/UDP/IP :
 - Principaux protocoles au cœur du monde Internet.
- OBEX (Object Exchange):
 - Protocole d'échange d'objets de la couche session développé par IrDA (Infrared Data Association). Possède des fonctionnalités similaires à HTTP mais très simplifiées
- WAE et WAP :
 - Ce sont des protocoles utilisés par les téléphones portables qui ont également été intégrés dans l'architecture Bluetooth.
- Protocoles de remplacement de câbles:
 - Emulation de liaison série du type RS232
- Protocoles d'adaptation de téléphonie :
 - TCS BIN
 - AT Commands

18

Sur un réseau : serveur web ou httpd

- Serveur web ou serveur httpd (http daemon) : logiciel qui sert des pages Web aux clients
- Capable d'interpréter les requêtes http arrivant sur le port associé au protocole HTTP (port 80 par défaut) et de fournir une réponse avec ce même protocole



- Par extension, on appelle souvent la machine physique sur laquelle tourne le logiciel un « serveur web »

19

Méthodes GET et POST du Protocole HTTP

■ GET

- Une requête GET est sans effet sur la ressource, il doit être possible de répéter la requête sans effet (idempotence)

```
GET / HTTP/1.1
Host: www.perdu.com
```

- Les paramètres de la requête sont post-fixés à l'URL :

```
website.com/directory/index.php?name=YourName&bday=YourBday
```

■ POST

- Pour transmettre des données en vue d'un traitement à une ressource. cette méthode est souvent utilisée en remplacement de la requête PUT, qui devrait être utilisée pour la mise à jour de ressources.
- Les paramètres sont transmises dans le corps de la requête (non visibles dans l'URL)

```
POST /test/demo_form.php
HTTP/1.1
Host: gfs.com
SAM=451 & MAT=62
```

20

UNIVERSITÉ
La Rochelle

Méthodes GET et POST du Protocole HTTP (suite)

- La requête doit être terminée par un double retour à la ligne (CRLF CRLF)
- Si la requête est conforme (ou non), le serveur HTTP doit répondre. Exemple :

```

HTTP/1.0 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Server: Apache/0.8.4
Content-Type: text/html
Content-Length: 59
Expires: Sat, 01 Jan 2000 00:59:59 GMT
Last-modified: Fri, 09 Aug 1996 14:21:40 GMT

<TITLE>Exemple</TITLE>
<P>Ceci est une page d'exemple.</P>

```

←
CRLF CRLF

←
CRLF CRLF


21

UNIVERSITÉ
La Rochelle

Réponse à une requête

- Les **codes d'état HTTP** les plus fréquents :
 - 200 : succès de la requête ;
 - 301 et 302 : redirection, respectivement permanente et temporaire ;
 - 401 : utilisateur non authentifié ;
 - 403 : accès refusé ;
 - 404 : page non trouvée ;
 - 500 et 503 : erreur serveur ;
 - 504 : le serveur n'a pas répondu.
- Les **types de données (type MIME)** les plus fréquents
 - text/html
 - image/jpeg
 - application/pdf
 - application/rtf


22



Contenu obtenu par les requêtes

- Pages statiques
 - Contiennent du code interprétable HTML, CSS et JavaScript (typiquement extension .htm,.html,.css, .js)
 - Stockées telles quelles sur les disques durs ou cartes mémoire des serveurs
- Pages dynamiques
 - créées (entièrement ou partiellement) au moment où elles sont demandées au serveur Web. En général, une application développée côté serveur (en PHP ou NodeJS, ou Python/Django, ...) fabrique ces pages.
 - Répondent à une demande spécifique (requête avec paramètres, en général des éléments d'un formulaire HTML)
 - Si utilisation d'un *template* HTML, selon le langage côté serveur, syntaxe space-holder...

23



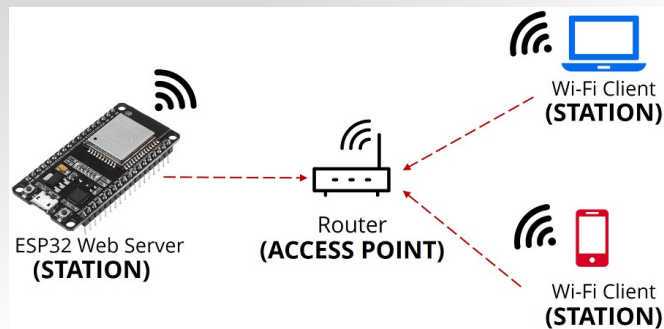
Création de contenu de façon dynamique

- Scripts CGI :
 - norme, pas langage
 - génèrent des pages complètes. Première technologie utilisée pour générer des pages dynamiques, encore très employée (grand stock de scripts CGI existants)
- Pages ASP, JSP, PHP :
 - écrites partiellement en HTML
 - comportent des scripts « serveurs » exécutés par le serveur Web au moment où la page est demandée
 - interrogent généralement une base de données
 - résultats insérés dans la page à la place des scripts.
 - La page résultante apparaît au client comme une page HTML statique (ce qu'elle n'est pas)
- Le choix de la technologie serveur dépend :
 - du système d'exploitation de la machine serveur
 - du type de bases de données à interroger
 - de la charge que doit supporter le serveur

24

ESP32 et protocole HTTP

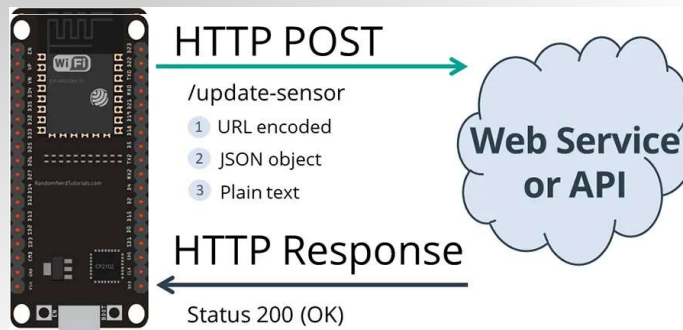
- Le *framework* Arduino contient des fonctions permettant de se connecter à un réseau WiFi en mode infrastructure (ouvert ou non), mais aussi de créer un Access Point
- Le *framework* contient des classes et méthodes de bases permettant d'envoyer des requêtes HTTP.



25

Utilisation de l'ESP32 comme un « client »

- L'ESP32 se connecte à un point d'accès, et envoie des requêtes vers un serveur
- L'ESP32 récupère la réponse. Uniquement gestion d'erreur car l'affichage d'une page Web n'est pas possible



26

Utilisation de l'ESP32 comme Access Point

- Création d'un Access Point (=Hotspot)

```
WiFi.softAP(ssid); // No password parameter, if you
                  // want the AP (Access Point) to be open
IPAddress IP = WiFi.softAPIP();
```

- Que faire avec un AP si pas connexion à internet ?
- Exécuter un serveur WEB : Difficile de deployer un framework *server-side* (comme un stack LAMP ou WAMP) sur l'ESP32.
- Classes et bibliothèques disponibles

```
WiFiServer server(80);
// in loop()
WiFiClient client = server.available(); // Listen for clients
while (client.connected()) {           // loop while the
                                        // client's connected
    if (client.available()) {           // if there's bytes
                                        // to read from the client,
        char c = client.read();         // read a byte
    }
```

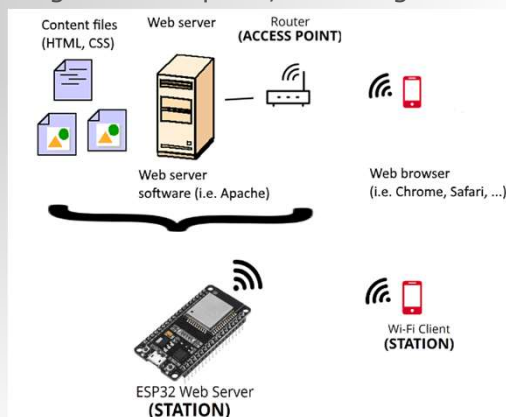
27

Utilisation de l'ESP32 comme « serveur »

- Traitement bas niveau :

- => traitement de chaine de caractere
- identification et décodage de la requête, formatage de la réponse, etc...
- OK si simple (LED on et off)

- Soit utilisation d'une bibliothèque qui simplifie la tâche. Ex : AsyncWebServer



```
server.on("/on", HTTP_GET, callback_to_call);
```

28

UNIVERSITÉ
La Rochelle

Utilisation de l'ESP32 comme « serveur »


- Interet de mettre un « serveur Web » sur un ESP32 : récupérer une page HTML décrivant une IHM qui peut alors s'afficher sur un terminal mobile



ESP 32 Web Server

Response

Request



Browser Client



ESP32 Web Server

Response

Request



Browser Client

29

UNIVERSITÉ
La Rochelle


Rappel HTML

- Balises <input> et <form>
 - Creation de champs interactifs
 - Création d'une requete

Veuillez saisir votre courriel pour recevoir nos publicités
 <input type="text" id="name">
- Si sur la page vous placer un formulaire :


```
<form>
  <input>
</form>
```
- <form> à deux attributs (optionnels) :
 - method : get ou post (get par défaut)
 - action : adresse à atteindre (même Host par défaut)

30




Méthode GET

```
<form method="get" action="nom_programme">
  Nom : <input type="text" name="nom">
  Prenom : <input type="text" name="prenom">
  Age : <input type="text" name="age">
  <input type="submit" value="OK">
</form>
```

- Au moment où l'utilisateur clique sur le bouton submit, la ligne de l'URL affiche par exemple :
http://site/nom_programme?nom=dupont&prenom=antoine&age=25
- Tout ce qui suit le ? correspond aux différents couples **variable=valeur** provenant du formulaire.
- QUERY_STRING prend comme valeur tout ce qui suit le ? dans l'URL
- Attention : caractères accentués, espaces...

31



Méthode POST

```
<form method="post" action="nom_programme">
  Nom : <input type="text" name="nom">
  Prenom : <input type="text" name="prenom">
  Age : <input type="text" name="age">
  <input type="submit" value="OK">
</form>
```

- Au moment où l'utilisateur clique sur le bouton submit, la ligne de l'URL affiche seulement :
http://site/nom_programme
- Lorsque le formulaire comporte l'instruction method="post" :
 - REQUEST_METHOD prend la valeur post
 - le programme du serveur reçoit, sur son entrée standard, d'abord la demande, ensuite le codage des paramètres
- La variable **CONTENT_LENGTH** permet de déterminer la longueur des données à lire

32

UNIVERSITÉ
La Rochelle

Les boutons sur les pages HTML

- **Restriction:** If your buttons are not for submitting form data to a server, be sure to set their type attribute to button. Otherwise they will try to submit form data and to load the (nonexistent) response, possibly destroying the current state of the document.
- **Autre possibilité (pour une IHM simple)**

Un lien peut avoir l'apparence d'un bouton. Pas de paramètres, **c'est la navigation qui effectue l'action** : navigation vers page /on → on allume, navigation vers page /off → on éteint

33

UNIVERSITÉ
La Rochelle

Reponse de l'ESP32 à une requete

- N'oubliez pas : l'ESP32 doit répondre de façon standard :
 - Code : 200 si OK, 404 si la « page » n'existe pas
 - S'il faut fournir une « page » d'interface, indiquer le type MIME de la réponse (text/html)
- En l'absence d'un système de fichier, la reponse se fait par envoi d'une chaine de caractère.
 - Attention aux caractères " ou ` qui peuvent exister dans le HTML, idem caractère \
 - Definition de la chaine en mode brut (RAW). Syntaxe :

```

R"delimiter( raw_characters )delimiter"
Char montexte[] = R"(" (
var button = document.createElement("button");
) " ";

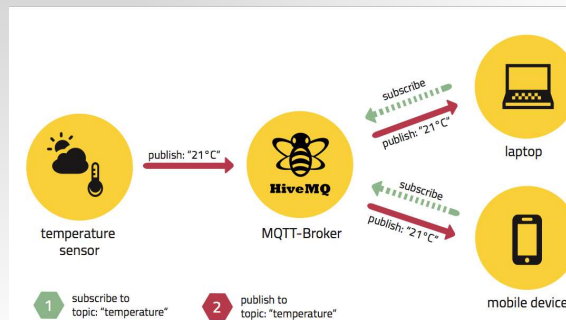
```

- Mini filesystem possible en mémoire Flash

34

MQTT (Message Queuing Telemetry Transport)

- MQTT est un protocole de messagerie publish-subscribe basé sur le protocole TCP/IP.
- MQTT v3.1.1 est maintenant un standard OASIS
 - Un Publisher envoie au broker des messages relatifs à des topics
 - Le Broker centralise l'information, accepte ou non les messages des Publishers pour les topics et envoie des notifications aux abonnés (Subscriber) à ces topics



35

Broker

- Broker local, dans le cloud, etc...
- Par ex. Adafruit/IO -> usage gratuit mais limité
- Le broker organise les données par arborescence. France
 - > Paris
 - > temperatureTank1
 - > temperatureTank2
 - > Poitiers
 - > temperatureTank1
 - > temperatureTank2
- Il est possible de s'abonner à un ensemble de données en renseignant un nœud parent, par exemple France/# ou France/Poitiers/#.

36

MQTT : Quality of Service (QoS)

- Charge utile (payload) :
texte, JSON

```
{
  "pointId": "point1",
  "value": "666.666",
  "timestamp": "2020-02-02 02:02:02",
  "quality": "true"
}
```

- The Quality of Service (QoS) level is an agreement between sender and receiver of a message regarding the guarantees of delivering a message.
- QoS : At most once (0) *fire and forget*
- At least once (1)
- Exactly once (2).

